

OPENUP/MMU-ISO

A. Rodriguez

2011



Universidad Nacional de La Plata  
FACULTAD DE INFORMATICA  
Biblioteca  
50 y 120 La Plata,  
biblioteca@info.unlp.edu.ar  
Tel (54-221) 423-0124 - int.: 59



DIF-03873



BIBLIOTECA  
FAC. DE INFORMÁTICA  
U.N.L.P.

# OpenUP/MMU-ISO

Soporte para un proceso de desarrollo de software  
conforme al Modelo ISO de Madurez en Usabilidad

Andrés Rodríguez

Director: Dr. Gustavo Rossi

Tesis presentada a la Facultad de Informática  
de la Universidad Nacional de La Plata  
como parte de los requerimientos para obtener el grado de

**Magister en Ingeniería de Software**

La Plata

11 de octubre de 2010

An aerial photograph of a tropical coastline. A large, dark bay is visible on the left side of the frame. To the right, a large, forested peninsula extends into the water. The land is covered in dense green vegetation, and the water is a deep blue. The sky is clear and light blue.

©2010

by

Andrés Santiago Rodríguez



# Agradecimientos

Este trabajo llevó más tiempo del que debería y no hubiera sido posible sin el apoyo de todos los que me acompañaron durante el trayecto. Impulsando, ayudando, aconsejando, sosteniendo, muchos me ayudaron para que finalmente llegara a la meta.

En primer lugar quiero agradecer a Nico, Fede, Mateo y Gaby. Fueron quienes más aguantaron el largo camino hasta este documento y a quienes más privé por llegar hasta acá.

Luego al viejo que lo mira desde el cielo, a mamá y a Juan. Siempre impulsaron y apoyaron las ganas de seguir creciendo y aprendiendo.

Finalmente al Lifa, que me brindó un ambiente propicio para desarrollar esta faceta. Son muchas las personas de Lifa que me ayudaron. Gustavo, Silvia, Gabriel y Alicia, que siguen confiando en mi trabajo más allá de toda lógica. Casco, que nunca dejó que perdiera las esperanzas. Y todos los lifianos de transferencia, especialmente mis compañeros de Medifé, que me dieron material de sobra para comprobar la necesidad de tareas de diseño centrado en usuario en el desarrollo de software.

Cada uno de ellos sabe cuánto esfuerzo costó terminar esta tesis, cuánto dejé en el camino y cuánto les debo a cada uno. Simplemente ¡gracias!

# Índice general

Índice de figuras	XIII
Índice de tablas	XVII
<b>1. Introducción</b>	<b>1</b>
<b>I Ingeniería de software, Usabilidad y Diseño Centrado en el Usuario</b>	<b>5</b>
<b>2. Definiciones preliminares</b>	<b>7</b>
2.1. Usabilidad . . . . .	7
2.2. Diseño de interacción . . . . .	9
2.3. Diseño Centrado en Usuario . . . . .	9
2.4. Capacidad y madurez	11
<b>3. Los procesos de desarrollo de software y el DCU</b>	<b>13</b>
3.1. Introducción . . . . .	13
3.2. Aproximaciones desde el campo HCI . . . . .	14
3.2.1. La ingeniería de usabilidad según Nielsen . . . . .	14
3.2.2. El ciclo de vida en estrella de Hix y Hartson	15
3.2.3. Diseño Centrado en el Uso de Constantine . . . . .	15
3.2.4. Ciclo de vida de Ingeniería de Usabilidad por Mayhew . . . . .	17
3.2.5. Los escenarios de Carroll y Rosson . . . . .	17
3.2.6. El proceso DCU de ISO . . . . .	18
3.3. Aproximaciones desde la Ingeniería de Software . . . . .	20
3.3.1. La propuesta de Costabile . . . . .	20
3.3.2. El Proceso Unificado . . . . .	21
3.3.3. DSDM . . . . .	22
3.3.4. Wisdom . . . . .	24
3.4. Oportunidad de intervención	25
<b>II Capacidad y Madurez en Usabilidad con el Proceso Unificado</b>	<b>27</b>
<b>4. Proceso Unificado open source: OpenUP</b>	<b>29</b>
4.1. Introducción . . . . .	29
4.2. El Eclipse Process Framework . . . . .	29
4.3. La biblioteca de prácticas para el EPF . . . . .	37
4.4. OpenUP . . . . .	49

4.4.1.	Configuración OpenUP/Basic . . . . .	51
4.4.2.	Fase de Inicio . . . . .	52
4.4.3.	Fase de Elaboración . . . . .	53
4.4.4.	Fase de Construcción . . . . .	54
4.4.5.	Fase de Transición . . . . .	54
<b>5.</b>	<b>Capacidad y madurez en usabilidad</b>	<b>57</b>
5.1.	Introducción . . . . .	57
5.2.	Modelos de madurez en usabilidad . . . . .	58
5.2.1.	Madurez en Usabilidad Corporativa de Nielsen . . . . .	58
5.2.2.	Ehrlich y Rohn . . . . .	59
5.2.3.	Lundmark y Toresson . . . . .	59
5.2.4.	Trillium . . . . .	59
5.2.5.	IBM, Evaluación de liderazgo en usabilidad . . . . .	61
5.2.6.	El método de Schaffer . . . . .	61
5.2.7.	INUSE HCS . . . . .	61
5.2.8.	INUSE Procesos . . . . .	62
5.2.9.	KESSU . . . . .	62
5.3.	El Modelo de Madurez en Usabilidad de ISO	64
5.3.1.	Modelo de referencia . . . . .	64
5.3.2.	El contenido de MMU-ISO	65
<b>6.</b>	<b>Conformidad de OpenUP/Basic al MMU-ISO</b>	<b>71</b>
6.1.	Introducción . . . . .	71
6.2.	Metodología de assesment . . . . .	71
6.3.	Resultados . . . . .	74
6.3.1.	Capacidad en DCU1 (Asegurar el contenido de DCU en la estrategia de sistemas) . . . . .	75
6.3.2.	Capacidad en DCU2 (Planificar y gestionar el proceso DCU) . . . . .	75
6.3.3.	Capacidad en DCU3 (Especificar los requerimientos de stakeholders y organización . . . . .	77
6.3.4.	Capacidad en DCU4 (Entender y especificar el contexto de uso)	78
6.3.5.	Capacidad en DCU5 (Producir soluciones de diseño) . . . . .	79
6.3.6.	Capacidad en DCU6 (Evaluar los diseños contra los requerimientos)	80
6.3.7.	Capacidad en DCU7 (Presentar y operar el sistema) . . . . .	81
6.4.	Perfiles de capacidad y posibilidades de extensión . . . . .	81
<b>III</b>	<b>Soporte para obtener Capacidad y Madurez en Usabilidad con OpenUP</b>	<b>85</b>
<b>7.</b>	<b>Extensión OpenUP/MMU-ISO</b>	<b>87</b>
7.1.	Introducción . . . . .	87
7.1.1.	La creación de Métodos para la Biblioteca de Prácticas del EPF . . . . .	87
7.1.2.	La contribución de esta Tesis: una Práctica y tres Configuraciones de método . . . . .	89
7.1.3.	Trabajos relacionados y antecedentes . . . . .	89
7.2.	Práctica: DesCU - Desarrollo Centrado en el Usuario . . . . .	93
7.2.1.	Roles . . . . .	94
7.2.2.	Productos de trabajo . . . . .	102
7.2.3.	Tareas . . . . .	121
7.2.4.	Orientaciones . . . . .	139
7.3.	Configuraciones de Método: OpenUP/MMU-ISO	139

7.3.1. OpenUP/MMU-ISO N1 . . . . .	139
7.3.2. OpenUP/MMU-ISO N2 . . . . .	154
7.3.3. OpenUP/MMU-ISO N3 . . . . .	157
<b>8. Plugins del EPF Composer</b>	<b>161</b>
8.1. El EPF Composer . . . . .	161
8.2. El uso del Composer . . . . .	164
8.3. Plugins lógicos y físicos en la EPL . . . . .	167
8.4. La familia de plugins OpenUP/MMU-ISO . . . . .	170
8.4.1. El plugin de la Práctica DesCU . . . . .	170
8.4.2. El plugin de Configuración OpenUP/MMU-ISO N1	178
8.4.3. El plugin de Configuración OpenUP/MMU-ISO N2	184
8.4.4. El plugin de Configuración OpenUP/MMU-ISO N3	184
<b>9. Conclusiones</b>	<b>187</b>
<b>Bibliografía</b>	<b>191</b>
<b>IV Apéndices</b>	<b>197</b>
<b>A. MMU-ISO: Procesos DCU</b>	<b>199</b>
<b>B. MMU-ISO: Niveles de capacidad</b>	<b>209</b>
<b>C. Assessment de OpenUP/Basic</b>	<b>213</b>
<b>D. Software adjunto</b>	<b>221</b>

# Índice de figuras

2.1. Componentes de usabilidad segun Norma ISO 9241 . . . . .	8
3.1. Modelo en Estrella, de [Hix and Hartson 1993]	15
3.2. Procesos lógicos en el Diseño Centrado en Uso . . . . .	16
3.3. Ciclo de ingeniería de Usabilidad según [Mayhew 1999] . . . . .	18
3.4. Ingeniería de usabilidad basada en escenarios, segun [Rosson and Carroll 2002] . . . . .	19
3.5. El proceso DCU en ISO 13407, [ISO/IEC 1999] . . . . .	20
3.6. Inclusión de usabilidad en ciclo de vida de software según Costabile . . . . .	21
3.7. Arquitectura de procesos en RUP, según [IBM 2005a] . . . . .	22
3.8. Ejemplo de Mapa de Navegación en el Plugin UX de RUP, según [IBM 2005b]	23
3.9. Ciclo de vida DSDM . . . . .	23
3.10. Fases del proceso en WISDOM . . . . .	24
3.11. Flujos de tareas, actividades, modelos y diagramas en WISDOM	25
4.1. Esquema de un Framework de Prácticas . . . . .	30
4.2. Marco conceptual para uso de SPEM 2.0	30
4.3. Elementos de Metamodelo SPEM 2.0 . . . . .	31
4.4. Perfiles UML definidos en SPEM 2.0 para el paquete Contenido de Método	31
4.5. Perfiles UML definidos para el paquete Estructura de Procesos en SPEM 2.0	32
4.6. Relaciones entre definiciones de Rol, Tarea y Producto de trabajo . . . . .	32
4.7. WBS de Fases de Ciclo de vida riesgo-valor	34
4.8. Diagrama de Flujo de Trabajo en SPEM 2.0 . . . . .	35
4.9. Diagrama de detalle de actividad en SPEM 2.0 . . . . .	36
4.10. Diagrama de Dependencias de Producto de Trabajo en SPEM 2.0 . . . . .	36
4.11. Bloques de Prácticas EPL . . . . .	38
4.12. Ciclo de vida de la iteración en OpenUP . . . . .	43
4.13. Ciclo de vida Riesgo-Valor . . . . .	45
4.14. Curvas de reducción de riesgo y entrega de valor	46
4.15. TDD en contexto . . . . .	48
4.16. Ubicación de OpenUP . . . . .	50
4.17. Actividades de una iteración en la Fase de Inicio . . . . .	52
4.18. Actividades de una iteración en la Fase de Elaboración . . . . .	53
4.19. Actividades de una iteración en la Fase de Construcción	54
4.20. Actividades de una iteración en la Fase de Transición . . . . .	55
5.1. Procesos de Diseño en Usabilidad de KESSU	64
5.2. Ciclo de vida de procesos DCU en MMU-ISO . . . . .	67
6.1. Perfil de Capacidad para un Nivel	72
6.2. Resultados para el Proceso DCU 1 . . . . .	74

6.3. Cantidad de atributos por logro alcanzado para el Nivel 1 . . . . .	82
6.4. Cantidad de atributos por logro alcanzado para el Nivel 2 . . . . .	82
6.5. Cantidad de atributos por logro alcanzado para el Nivel 3 . . . . .	83
6.6. El perfil de capacidad por proceso DCU en niveles 1 a 3 de MMU-ISO . . .	83
7.1. Patrón de entrega de plugin WEUCD, de [ATG 2010] . . . . .	91
7.2. Flujo de actividades de Diseño de Usabilidad, según [Goransson et al. 2003]	93
7.3. Asignaciones del Rol Patrocinador . . . . .	96
7.4. Asignaciones del Rol Líder Técnico de Dominio . . . . .	97
7.5. Asignaciones del Rol Representante de Usuarios . . . . .	98
7.6. Rol de Usuario final . . . . .	99
7.7. Rol de Especialista en Experiencia de Usuario . . . . .	100
7.8. Asignaciones del Rol Diseñador de Experiencia de Usuario . . . . .	101
7.9. Asignaciones del Rol Tester de Experiencia de Usuario . . . . .	102
7.10. Perfil de Usuario según Hackos para un agente de viajes . . . . .	106
7.11. Ejemplo de Personaje, de [Pruitt and Adlin 2006] . . . . .	107
7.12. Ejemplo de Personaje, de [Pruitt and Adlin 2006] . . . . .	107
7.13. Lista de control para definir las tres Cs del Rol de Uso de [Constantine and Lockwood 1999] . . . . .	108
7.14. Notación en HTA, de [Lorés 2001] . . . . .	112
7.15. Ejemplo de HTA, de [Lorés 2001] . . . . .	113
7.16. Uso de CTT en la elaboración de un Modelo de Tareas . . . . .	115
7.17. Prototipo de IU de baja fidelidad . . . . .	117
7.18. Prototipo de IU de alta fidelidad . . . . .	117
7.19. Ejemplo de Storyboard . . . . .	119
7.20. Ejemplo de storyboard de UX con prototipos de pantallas de baja fidelidad	119
7.21. Mapa de navegación . . . . .	121
7.22. Mapa de relaciones entre Práctica DesCU y procesos MMU-ISO . . . . .	123
7.23. Ciclo de Vida de MMU-ISO . . . . .	126
7.24. Definición de la Tarea Identificar y Especificar el contexto de uso . . . . .	128
7.25. Definición de la Tarea Diseñar la UX . . . . .	131
7.26. Definición de la Tarea Revisar el Diseño de la Experiencia de usuario . . .	132
7.27. Definición de la Tarea Ejecutar pruebas de usabilidad . . . . .	134
7.28. Definición de la Tarea Ejecutar pruebas de usabilidad . . . . .	136
7.29. Definición de la Tarea Producir material de entrenamiento y soporte al usuario . . . . .	137
7.30. Definición de la Tarea Guiar y dar soporte al usuario en implementación .	138
7.31. Iteración típica de Fase de Inicio con DesCU . . . . .	140
7.32. Actividad Iniciar el Proyecto en OpenUP/MMU-ISO N1 . . . . .	141
7.33. Actividad: Identificar y refinar requerimientos con la Práctica DCU . . . . .	143
7.34. Actividad: Acordar el abordaje técnico con la Práctica DCU . . . . .	144
7.35. Actividad: Planificar y gestionar la iteración en OpenUP/MMU-ISO N1 . .	145
7.36. Iteración típica de Fase de Elaboración con DCU . . . . .	147
7.37. Actividad: Desarrollar la arquitectura y la UX en OpenUP/MMU-ISO N1	147
7.38. Actividad: Desarrollar un incremento de solución . . . . .	149
7.39. Actividad: Testear la Solución . . . . .	151
7.40. Iteración tipo de la Fase Construcción . . . . .	152
7.41. Iteración tipo de la Fase Transición . . . . .	153
7.42. Actividad: Guiar y dar soporte al usuario en implementación . . . . .	153
7.43. Detalle de la Actividad: Iniciar el Proyecto en OpenUP/MMU-ISO N3 . . .	158
8.1. Pantalla de edición de una Tarea en el Composer . . . . .	162
8.2. Estructura de organización del repositorio en el Composer . . . . .	163

8.3. Creación de un Rol dentro del Paquete de Contenido de Método	165
8.4. WBS del Ciclo de Vida OpenUP/MMU-ISO Nivel 1	165
8.5. Ejemplo de sitio web generado con Composer	167
8.6. Tipos de plugins en EPL y relaciones entre ellos	168
8.7. Partes de un plugin y sus relaciones	169
8.8. Ubicación del Plugin Práctica DCU en Composer	171
8.9. Plugins físicos de la Práctica DCU en Composer	171
8.10. Estructura interna del plugin Base en la Práctica DCU	173
8.11. Relación entre roles básicos de EPL y roles de DCU	173
8.12. Descripción de Diseñador de UX	174
8.13. Descripción de Tester de UX	174
8.14. Definiciones de Tarea incluidas en el Plugin Base de la Práctica DCU	175
8.15. Definición de Tarea Entender y Especificar Contexto de Uso. Solapa Descripción	176
8.16. Definición de Tarea Entender y Especificar Contexto de Uso. Solapa Pasos	176
8.17. Definición de Tarea Entender y Especificar Contexto de Uso. Solapa Productos de Trabajo	177
8.18. Definición de Tarea Entender y Especificar Contexto de Uso. Solapa Pre-visualización	177
8.19. Definición de Tarea Entender y Especificar Contexto de Uso. Solapa Pre-visualización	179
8.20. Responsabilidad de Productos de Trabajo del Diseñador de UX	179
8.21. Responsabilidad de Productos de Trabajo del Especialista de UX	179
8.22. Tareas definidas para contribuir asignaciones a las básicas	180
8.23. Asignación de roles a Entender y especificar Contexto de Uso mediante contribución	180
8.24. Asignación de roles a Entender y especificar Contexto de Uso mediante contribución	181
8.25. Ubicación de plugins físicos para configuraciones OpenUP/MMU-ISO Nx	183
8.26. Iteración tipo en fase de Transición de OpenUP/MMU-ISO N1	183
8.27. WBS del Proceso de Entrega OpenUP/MMU-ISO N3	185
C.1. Resultados para el Proceso DCU 1	214
C.2. Evidencia para determinar Capacidad en DCU2	215
C.3. Resultados del assessment de DCU 3	216
C.4. Evidencia de capacidades OpenUP en DCU 4	217
C.5. Evidencia de capacidad OpenUP en DCU 5	218
C.6. Resultados del assessment para DCU 6	219

# Índice de tablas

3.1. DCU vs Diseño Centrado en Uso . . . . .	16
4.1. Mapeo entre principios de OpenUP y valores del Manifiesto Agil [Kent Beck 2001] . . . . .	51
5.1. Modelo de Madurez en Usabilidad de Proyecto	60
5.2. Modelo de Madurez en Usabilidad de Schaffer . . . . .	62
5.3. Prácticas de gestión en INUSE HCS	63
6.1. Esquema de hoja de assessment . . . . .	73
6.2. Atributos a mejorar para dar conformidad a MMU-ISO en Nivel 1	84
7.1. Productos de Trabajo a incorporar . . . . .	103
7.2. Criterios generales de usabilidad . . . . .	110
7.3. Medidas para objetivos de usabilidad . . . . .	111
7.4. Métodos para elaborar Modelos de Tareas . . . . .	112
8.1. Relación entre Productos de Trabajo y Slots . . . . .	175
8.2. Relación de las Tareas en Práctica DCU y las Disciplinas de la EPL	178
8.3. Asignaciones de Roles a Tareas de DCU	182

# Capítulo 1

## Introducción

### Motivación y alcance

La usabilidad es un atributo de calidad cuya importancia ha sido reconocida plenamente por la comunidad de ingeniería de software. Ya no se discute la necesidad de que cualquier producto pensado para interactuar con personas resulte fácil de utilizar, apropiado a las tareas y usuarios para las que fue concebido y cuyo uso brinde diferentes grados de satisfacción al usuario. La literatura da cuenta de fracasos de productos de software o proyectos de desarrollo que nunca llegaron a instancias productivas por fallas asociables con usabilidad, como ha mostrado [Landauer 1996], por ejemplo.

La expansión de las GUIs aumentó las posibilidades de interacción de usuarios y ha impulsado esa toma de conciencia. Lamentablemente, también ha servido para asociar casi en forma exclusiva la usabilidad de una aplicación con el diseño visual de sus pantallas. Es común que los ingenieros de software, cuyas prácticas dominan los procesos y técnicas de construcción de software consideren la usabilidad como un tema de interfaz de usuario, que hay que abordar al final del desarrollo, luego de que la parte funcional del sistema ya ha sido especificada, diseñada y, en algunos casos, construida.

Desde el campo de estudios sobre la Interacción Humano Computadora (HCI) se han desarrollado un conjunto de teorías, métodos, técnicas y herramientas para construir sistemas más usables. Se ha mostrado que para conseguir usabilidad hace falta abordar el diseño de la interacción deseada de manera integral, enfocando no sólo en las pantallas, sino en todos los aspectos del software que impactarán finalmente sobre la calidad del producto durante su uso [Sharp et al. 2007].

También desde la ingeniería de software se ha mostrado la relación entre la usabilidad de un sistema y aspectos de su desarrollo que tienen poco o nada que ver con sus interfaces gráficas. Por ejemplo, las decisiones tempranas sobre la arquitectura del software pueden afectar de manera significativa el nivel de usabilidad alcanzable por un sistema ([Bass and John 2001a],[Folmer and Bosch 2005]).

Para hacer frente al desafío de incorporar prácticas de desarrollo que promuevan el logro de mejores niveles de usabilidad, tanto desde la HCI como desde la IS se han realizado propuestas de cambios en procesos y métodos de trabajo.

Desde el campo de la Interacción Humano Computadora (HCI) se sostiene que la forma de desarrollo denominada “Diseño centrado en el usuario” es necesaria para garantizar el logro de los objetivos de usabilidad en el producto final. Esta estrategia implica el involucramiento activo a lo largo de todo el ciclo de vida del producto de los diferentes afectados por el sistema final, desde su usuario directo hasta quienes pagan por el software (que suelen no ser las mismas personas).

Por su parte, la IS ha planteado la necesidad de establecer una Ingeniería de Software

Centrada en Personas (HCSE) para resumir la idea de la necesidad de compatibilizar el enfoque proveniente de HCI y el tradicional de la IS [Seffah et al. 2005], [Seffah et al. 2009]. Sin embargo, aún los métodos de IS no incorporan masivamente las prácticas, técnicas, herramientas y métodos de interacción y usabilidad. Seffah ha indicado que “la estructura y técnicas de HCI son todavía relativamente desconocidas, subutilizadas, difíciles de dominar y con poca integración esencial en equipos de desarrollo de software” [Seffah and Metzker 2004].

Los métodos de usabilidad y HCI en general pueden ser difíciles de aplicar a la ingeniería de software debido a las diferencias conceptuales y superposiciones entre ambos campos. Estas superposiciones no están formuladas claramente desde el punto de vista de la IS forzando a las organizaciones que desarrollan software a realizar costosas investigaciones para introducir las técnicas de usabilidad en tales prácticas. Por ejemplo, las actividades de ingeniería de requerimientos que son abordadas tanto por HCI como por la IS.

La importancia del esfuerzo de integración es a menudo erróneamente minimizada. Para los desarrolladores promedio, la usabilidad sólo afecta a la interfaz de usuario y ésta sólo consiste en los elementos concretos con los cuales el usuario interactúa y su comportamiento de respuesta, por lo tanto se considera sólo un asunto de diseñadores gráficos. Tales errores simplifican el problema de integración en la mente del ingeniero de procesos: como es un tema de diseñadores gráficos, sólo requiere el agregado de actividades de usabilidad en las áreas de proceso en las cuales estos temas son considerados, con lo cual se termina la superposición. Si consideramos la usabilidad como un tema de proceso de desarrollo, esto implica que hay que incluir actividades de usabilidad a lo largo de todo el proceso, con el desafío de integrar diferentes culturas de desarrollo en el mismo tipo de actividad.

Se sostiene que existe una relación directa entre el nivel de usabilidad que alcanza un producto y la utilización de un proceso de diseño centrado en usuario para desarrollarlo. Sin embargo, no todos los procesos se pueden convertir en centrados en usuario simplemente haciendo unas pocas modificaciones. Por ejemplo, la transformación requerida para un proceso o cultura organizacional basados en especificaciones detalladas que son producidas antes que cualquier diseño e implementación sería muy drástica e impracticable. La complejidad del lado humano en HCI hace casi imposible crear un diseño correcto en la primera intención. Aspectos cognitivos, sociológicos, educativos, físicos y emocionales pueden jugar un rol importante en cualquier interacción entre el usuario y el sistema y no se pueden predecir completamente. De las características de un proceso centrado en usuario, la iteratividad es la única que es intrínsecamente inherente al proceso de software.

Otra de las dificultades se relaciona con la mejora controlada de los procesos, que permita acompañar desde la usabilidad el crecimiento de un equipo u organización en sus capacidades de ingeniería de software. Para ello se han propuesto diferentes modelos de capacidad y madurez (por ejemplo CMMI y SPICE/ISO 15504[ISO/IEC 2006]). En el campo de la usabilidad también se han desarrollado algunos de estos modelos que pueden ser empleados como hojas de ruta para crecer en capacidad y madurez de usabilidad de manera previsible. El Modelo de Capacidad y Madurez en Usabilidad de la Organización Internacional de Estándares (ISO) que fue desarrollado de manera iterativa y con revisión de expertos, intenta proveer una base para que quienes planifican sepan qué actividades centradas en los usuarios incluir en un proyecto en particular, así como asistir a quienes desean mejorar la manera en que sus equipos realizan tales actividades.

El MMU-ISO está contenido en el reporte ISO TR 18529 que tiene como modelo de referencia a SPICE y utiliza su misma estructura: una dimensión de procesos versus una de capacidades. MMU-ISO define siete áreas de proceso para el Desarrollo Centrado en el Usuario: Asegurar contenido DCU en la estrategia de sistemas, Planificar y gestionar el proceso DCU, Especificar los requerimientos de stakeholders y organización, Entender

y especificar el contexto de uso, Producir soluciones de diseño, Evaluar los diseños contra los requerimientos, Presentar y operar el sistema. La capacidad en uno de estos procesos se mide en la escala de seis grados propuesta por ISO 15504: Incompleto, Realizado, Gestionado, Establecido, Predecible, Optimizado.

En este contexto, la tesis se plantea dos objetivos. Por un lado, extender uno de los procesos de desarrollo más adoptados en la comunidad de IS (el Proceso Unificado) para facilitar su instanciación como centrado en usuario y mejorar las promesas de usabilidad al aplicarlo. Por otro, proveer con esa extensión una especie de hoja de ruta que permita a las organizaciones de software crecer en la capacidad y madurez con la que emplean las prácticas de usabilidad.

Para ello será necesario revisar las propuestas de incorporación al desarrollo de software de actividades de usabilidad, describir el MMU-ISO y analizar el Proceso Unificado a la luz del MMU de ISO. Esta evaluación permitirá determinar que el nivel de Capacidad y Madurez en Usabilidad alcanzable con ese proceso.

Luego se propondrán las adaptaciones necesarias al Proceso Unificado que permitan satisfacer la necesidad de incluir actividades de usabilidad y hacerlo de una manera que permita conformar al MMU-ISO como hoja de ruta. Utilizaremos para este trabajo la versión open source del Proceso Unificado, liberada por IBM a la Fundación Eclipse y conocida como OpenUP.

## Estructura de la tesis

La tesis está organizada en tres partes. La primera (Capítulos 2 y 3) plantea las definiciones básicas que utilizaremos en el resto del trabajo (usabilidad, diseño de interacción, capacidad en usabilidad, etc.) y revisa los métodos de desarrollo de software que proponen la inclusión del diseño centrado en usuario generadas desde el campo de HCI y desde la Ingeniería del Software.

La segunda parte se compone de tres capítulos. En el primero, presentamos la versión open source del Proceso Unificado, denominada OpenUP, que es una instanciación del framework de buenas prácticas conocido como EPL (Eclipse Practice Library). En el segundo, presentamos el MMU-ISO. Cerramos esta segunda parte con un assessment de la conformidad al MMU-ISO de OpenUP con el objetivo de determinar las oportunidades de integración de usabilidad en el Proceso Unificado.

Finalmente, en la tercera parte presentamos la contribución de la tesis: una nueva Práctica para agregar a la EPL, el Desarrollo Centrado en Usuario (DesCU), y tres instanciaciones del framework basadas en OpenUP que permitirán alcanzar los niveles 1, 2 o 3 de Capacidad en Usabilidad en el marco del MMU-ISO. En el último capítulo mostramos la implementación de la Práctica DesCU y de las tres configuraciones como extensiones o plugins para el Eclipse Process Framework Composer.

## Parte I

# Ingeniería de software, Usabilidad y Diseño Centrado en el Usuario

## Capítulo 2

# Definiciones preliminares

### Introducción

Esta tesis entrelaza la noción de usabilidad con procesos de desarrollo de software y modelos de capacidad y madurez. Para facilitar la unificación del lenguaje en el resto del trabajo, en este capítulo simplemente presentamos las definiciones básicas de los cuatro conceptos que utilizaremos: *usabilidad*, *diseño de interacciones*, *diseño centrado en usuario* y *capacidad en usabilidad*.

### 2.1. Usabilidad

Nacida en el marco de los estudios sobre la interacción entre las personas y los sistemas de software, conocido como HCI (por las iniciales de Human Computer Interaction), la usabilidad ha recorrido un camino largo hasta comenzar a ser reconocida en las últimas dos décadas como una contribución importante a la calidad de un producto.

Desde las aplicaciones que se ejecutan en máquinas de escritorio, pasando por las que corren en dispositivos móviles de comunicaciones hasta aquellas que están embebidas en artefactos de uso tan específico como un medidor de nivel de glucosa en sangre, todas deben gran parte de su éxito a la mayor o menor posibilidad de que quienes deben interactuar con ellas puedan hacerlo con algún grado de eficiencia, eficacia y satisfacción.

Existen diferentes definiciones de usabilidad. Algunas provienen de los pioneros del campo, otras de los intentos por sistematizar y encuadrar a la disciplina en el contexto de los procesos de construcción de software.

Uno de los profesionales más respetados del campo y gestor de gran difusión alcanzada por la disciplina, Jakob Nielsen, la presentó en “Usability engineering” vinculada a cinco atributos esenciales que debe mostrar un sistema: facilidad de aprendizaje, eficiencia, recordabilidad, gestión de errores, satisfacción de usuario [Nielsen 1993].

La norma ISO 9126 siguió de alguna manera este enfoque al definir la usabilidad con cuatro atributos del sistema: facilidad para aprender, capacidad de comprensión, operabilidad y atractivo [ISO/IEC 2000b].

La definición que aplicaremos en esta tesis es la que luego ISO estableció en la Norma 9241-11 ISO/IEC [1998]:

La medida en la que un producto se puede usar por los usuarios especificados para alcanzar sus metas con efectividad, eficiencia y satisfacción en un contexto de uso especificado

Esta definición representa un contexto más amplio de usabilidad, enfatizando la relación entre la usabilidad de un producto y el contexto de uso para el cual fue pensado. La figura

2.1 muestra gráficamente los componentes de usabilidad planteados en ISO 9241-11 y la relación entre ellos.

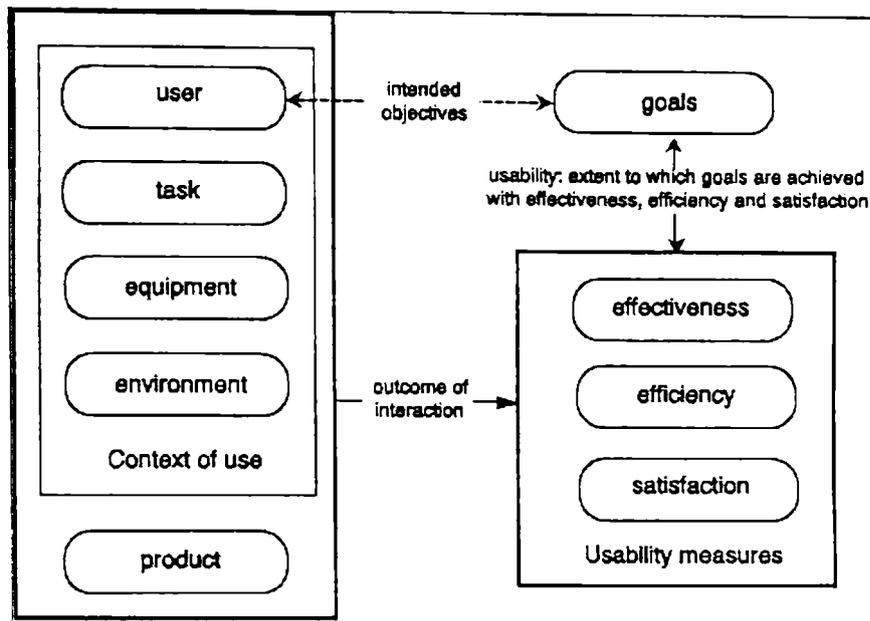


Figura 2.1: Componentes de usabilidad según Norma ISO 9241

La definición de ISO 9241-11 ha comenzado a convertirse en el estándar de facto, utilizada entre otros para la elaboración del CIF (Common Industry Format) para las pruebas de usabilidad, acordado entre las principales corporaciones y organizaciones involucradas en el desarrollo de software [ANSI 2001].

Es interesante observar que esta definición indica que la usabilidad no es una propiedad absoluta de un producto sino que existe en relación con los usuarios especificados de ese producto (lo que es lo mismo que decir que un sistema puede ser muy usable para algunos, pero inusable para otros). Incluso, para cada usuario la usabilidad estará relacionada con el logro de las metas en un entorno de uso definido. Finalmente, la usabilidad es una característica mensurable definida con tres atributos: efectividad, eficiencia y satisfacción.

Por ejemplo, un objetivo de usabilidad para un sistema para gestión de financiadoras de salud podría establecerse de la siguiente manera:

El 85% [efectividad] de 10 liquidadores con experiencia [usuario especificado] pueden liquidar la facturación de una internación por una apendisectomía [meta especificada] en menos de una hora [eficiencia] con una tasa de satisfacción de 7 puntos sobre una escala de 1 a 10 [satisfacción] en cualquier estación de trabajo de la empresa [entorno de uso]

Cualquier producto tiene normalmente diferentes grupos de usuarios. Algunas de las metas serán compartidas por más de un grupo y habrá otras metas exclusivas de algún grupo. Esta diferencia entre los usuarios, sus metas y sus entornos de uso se extiende a la usabilidad y por lo tanto las medidas esperables de la misma serán diferentes para cada grupo.

La usabilidad no es, entonces, una medida única, sino un conjunto de mediciones de efectividad, eficiencia y satisfacción que a su vez son funciones de los usuarios, sus metas y sus entornos de uso.

## 2.2. Diseño de interacción

Las definiciones de usabilidad indican como hemos visto que el logro de niveles adecuados de usabilidad está íntimamente relacionado con las interacciones que el producto o sistema permite y soporta en su relación con los usuarios.

Esa interacción no es un producto natural o algo que viene dado simplemente por la aplicación correcta de técnicas de Diseño de software o de Diseño gráfico para las pantallas. Debe ser diseñada explícitamente.

Se trata de plantear el Diseño del software en un marco más amplio, el de crear experiencias de usuario que mejoren y aumenten el modo en que la gente trabaja, se comunica e interactúa. Como ha señalado Winograd, se trata de “diseñar espacios para la comunicación e interacción humana” [Winograd 1997].

¿Cómo se integra el Diseño de interacciones con las otras disciplinas que participan en el Diseño de sistemas basados en software como la ingeniería del software? Sharp y sus colegas [Sharp et al. 2007] plantean la misma analogía con la construcción de edificios que alguna vez utilizara Terry Winograd. Se trata de observar los diferentes aspectos de un mismo edificio que abordan los arquitectos y los ingenieros civiles. Los primeros están preocupados con la gente y sus interacciones entre sí y con los espacios construidos (¿es correcto el balance entre espacios públicos y privados? están suficientemente cerca los lugares para cocinar y para comer? ¿el lugar se puede usar para vivir de otra manera?). Los ingenieros, en cambio se interesan en aspectos vinculados con la realización del proyecto (costo, duración, estructura, regulaciones, métodos constructivos, etc).

El Diseño de interacción entonces tiene que ver con el Diseño *para* la experiencia del usuario. Como plantea Jesse Garret: *todos los productos que son utilizados por alguien tienen una experiencia de usuario: diarios, botellas de ketchup, sillones, sweaters* [Garrett 2002]. Y lo mismo vale para los productos y sistemas basados en software. Se trata de diseñar para una relación entre la gente y el producto. Es obvio que uno no puede diseñar la experiencia del usuario, sino diseñar para esa experiencia. No se puede diseñar una situación de satisfacción o de efectividad, pero se pueden incluir en el sistema características que generen o evoquen satisfacción y que permitan alcanzar las metas deseadas con el compromiso óptimo entre las habilidades y capacidades del usuario y los desafíos de la tarea.

En este contexto, el diseño de la interacción es la parte del proceso donde las metas de usabilidad deberían ser tenidas en cuenta para establecer las acciones de diseño necesarias para que el producto final provea una experiencia de usuario que permita alcanzar tales metas.

## 2.3. Diseño Centrado en Usuario

Para obtener productos usables, prevalece como paradigma del proceso de ideación, construcción e implementación lo que se conoce como *Diseño Centrado en el Usuario* (DCU), también denominado ingeniería de usabilidad o diseño centrado en las personas.

El Diseño Centrado en Usuario ha sido descrito por ejemplo por [Hix and Hartson 1993], [Beyer and Holtzblatt 1998], [Mayhew 1999] o [Sharp et al. 2007]. La organización ISO también ha planteado el DCU como forma de obtener productos usables en las Normas 13407 [ISO/IEC 1999] y el Reporte 18529 [ISO/IEC 2000a].

En [Sharp et al. 2007], uno de los libros de texto más completos sobre el Diseño de interacción, Sharp y sus colegas citan el seminal trabajo de Gould y Lewis [Gould and Lewis 1985], que propugna tres principios claves de un proceso DCU:

1. Poner foco en usuarios y sus tareas desde el comienzo. Esta debería ser la fuerza conductora del desarrollo. Conocer a los usuarios implica estudiar su comportamiento y

el contexto de uso para diseñar un sistema que les dé soporte. Además los usuarios deben ser consultados durante todo el proceso de desarrollo y sus opiniones, tenidas en cuenta seriamente.

2. Establecer metas claras de usabilidad y experiencia que permitan realizar mediciones empíricas para saber si se avanza en su obtención durante el proyecto
3. Diseñar de manera iterativa, característica esencial de un proceso que permita a usuarios y diseñadores comprender a fondo el dominio a medida que se encuentran soluciones y se descubren nuevos problemas.

Otros autores respetados en la formación de profesionales de HCI, Ben Schneiderman [Shneiderman et al. 2009] y John Carroll [Carroll 1995], caracterizan un proceso de desarrollo para la usabilidad con cuatro atributos en la misma línea:

- El diseño es un proceso, no es un estado ni puede ser representado adecuadamente en forma estática.
- El proceso de diseño es esencialmente no jerárquico, ni top-down ni bottom-up.
- El proceso es radicalmente transformador, involucra el desarrollo de soluciones parciales y provisionales que incluso pueden ser descartadas para el diseño final
- El diseño involucra intrínsecamente descubrir nuevas metas

Como vemos un proceso de desarrollo para la usabilidad tiene dos características esenciales: **focaliza desde el comienzo en usuarios reales** y los involucra en el proyecto para conocer sus requerimientos cabalmente y es **intrínsecamente iterativo**, un devenir de aproximaciones sucesivas e incrementales hacia la solución de un problema que sólo será plenamente descubierto al llegar al punto final.

En esta tesis utilizaremos como referencias principales los estándares y reportes de ISO. Más adelante describiremos en detalle el proceso DCU definido en las mismas. Aquí planteamos sólo las principales actividades que ISO establece para un proceso DCU [ISO/IEC 1999]:

- Entender y especificar el contexto de uso: quién es el usuario, su entorno de uso y las tareas para las que el producto será utilizado.
- Especificar los requerimientos del usuario y la organización: determinar los criterios de éxito de usabilidad del producto en términos de las tareas del usuario. Establecer guías y restricciones de diseño.
- Producir soluciones de diseño: incorporar en las soluciones todo el conocimiento disponible sobre interacción hombre-computadora
- Evaluar los diseños contra los requerimientos: testear los diseños contra las tareas del usuario.

Estas actividades lamentablemente todavía no forman parte de todos los procesos de desarrollo de software. En algunos casos, aunque estén presentes, su impacto en el proceso de desarrollo es casi imperceptible. Varios trabajos han señalado el desafío de mejorar la inserción del DCU en las organizaciones de desarrollo de software, por ejemplo el conjunto de trabajos presentados en la serie de Conferencias para una Ingeniería de Software Centrada en las Personas (Human Centered Software Engineering), recogidas en [Seffah et al. 2005] y [Seffah et al. 2009]



## 2.4. Capacidad y madurez

En el dominio de los modelos de mejoras de procesos de software, un nivel de capacidad (*capability level*) se define por el logro alcanzado en el manejo de un conjunto apropiado de prácticas específicas y genéricas para un área de proceso. En cambio, un nivel de madurez (*maturity level*) se define como una mejora de procesos transversal a un conjunto predefinido de áreas en las cuales todas las metas en el conjunto son obtenidas [SEI 2006].

Por lo tanto, podemos decir que la capacidad en usabilidad mide el nivel con que una organización lleva adelante alguna de las áreas o aspectos de un proceso de diseño centrado en usuario. La madurez en usabilidad es una evaluación de qué tan avanzada está una organización respecto de las diferentes vistas del proceso de desarrollo.

Ambas, capacidad y madurez en usabilidad, definen *la característica de una organización que determina su habilidad para desarrollar en forma consistente productos con un nivel de usabilidad alto y competitivo* [Jokela 2001a].

Cuando una organización de desarrollo alcanza alto nivel de capacidad en usabilidad, el proceso DCU que utiliza es lo suficientemente efectivo y eficiente para generar productos y sistemas usables. Del mismo modo, un bajo nivel de capacidad en usabilidad suele revelar la inexistencia (o falta de efectividad) de un proceso DCU.

¿Cómo conseguir o mejorar el nivel de capacidad en usabilidad de una organización? En el mundo de la ingeniería de software se han desarrollado modelos de evaluación de procesos para guiar los esfuerzos de mejora, por ejemplo [ISO/IEC 2006], [Kuvaja 1995], [Paulk 1995]. Mediante la evaluación de los procesos se puede identificar las fortalezas y debilidades de una organización de software y usar esta información para encarar acciones de mejora. De la misma manera, en el campo del diseño centrado en usuario, existen modelos de capacidad en usabilidad.

Siguiendo la misma lógica, se desarrollaron modelos de capacidad y madurez en usabilidad. Mediante la evaluación de la capacidad en usabilidad se pueden desarrollar acciones de mejora que aumenten la capacidad en usabilidad de una organización.

Este trabajo utilizará el modelo de capacidad en usabilidad presentado por la ISO en ISO TR 18529 [ISO/IEC 2000a] para proponer una extensión de la versión open source del Proceso Unificado cuya instanciación brinde un proceso con conformidad a ese modelo de capacidad y madurez.

## Capítulo 3

# Los procesos de desarrollo de software y el DCU

### 3.1. Introducción

Los métodos de usabilidad y HCI en general pueden ser difíciles de aplicar a la ingeniería de software debido a las diferencias conceptuales y superposiciones entre ambos campos. Algunas actividades relacionadas con la ingeniería de requerimientos y el testing son abordadas tanto por usabilidad como por la ingeniería de software. Estas superposiciones no están formuladas claramente desde el punto de vista de la IS forzando a las organizaciones que desarrollan software a realizar costosas investigaciones para introducir las técnicas de usabilidad en tales prácticas.

La importancia del esfuerzo de integración es a menudo erróneamente minimizada debido a una percepción equivocada de la usabilidad: se la considera sólo vinculada con la interfaz de usuario. Para los desarrolladores promedio, la interfaz de usuario consiste en los elementos concretos con los cuales el usuario interactúa y su comportamiento de respuesta, por lo tanto se considera sólo un asunto de diseñadores gráficos. Tales errores simplifican el problema de integración en la mente del ingeniero de procesos: como es un tema de diseñadores gráficos, sólo requiere el agregado de actividades de usabilidad en las áreas de proceso en las cuales estos temas son considerados, con lo cual se termina la superposición. Si consideramos la usabilidad como un tema de todo el proceso de desarrollo, esto implica que hay que incluir actividades de usabilidad a lo largo del ciclo de vida completo, con el desafío de integrar diferentes culturas de desarrollo en el mismo tipo de actividad. Varios autores han señalado por ejemplo, la relación entre las decisiones tempranas sobre la arquitectura de software y la usabilidad que es posible alcanzar luego [Bass and John 2001b],[Bass and John 2001a],[Folmer and Bosch 2005].

En el capítulo anterior mostramos la relación entre el nivel de usabilidad de un producto y un proceso de diseño centrado en usuario. Sin embargo, no todos los procesos se pueden convertir en centrados en usuario simplemente haciendo unas pocas modificaciones. La transformación requerida para que un proceso o la cultura de una organización basada en un ciclo de vida en cascada se convierta en centrada en usuario sería muy drástica. Esta aproximación implica que especificaciones detalladas son producidas antes que cualquier diseño e implementación. La complejidad del lado humano en HCI hace casi imposible crear un diseño correcto en la primera intención. Aspectos cognitivos, sociológicos, educativos, físicos y emocionales pueden jugar un rol importante en cualquier interacción entre el usuario y el sistema y no se pueden predecir completamente. De las características de un proceso centrado en usuario, la iteratividad es la única que es intrínsecamente inherente al proceso de software.

A continuación mostramos algunas de las propuestas de integración de actividades de usabilidad y diseño de interacción en los procesos de desarrollo de software. Comenzamos con las que surgieron del campo de HCI y luego completamos con las originadas en la Ingeniería de Software. Cerramos el capítulo describiendo la oportunidad de intervención que detectamos y que motiva gran parte de este trabajo.

## 3.2. Aproximaciones desde el campo HCI

En la literatura de HCI existen varios reportes de modelos para procesos de desarrollo de software que buscan integrar las técnicas de diseño de interacción para la usabilidad con las actividades propias de los ingenieros de software. En esta parte del capítulo vamos a presentar algunos de ellos. Hacemos una breve referencia al tipo de proceso propuesto por Nielsen y después describimos los procesos propuestos por Hix y Hartson (conocido como *Ciclo de Vida en Estrella*), Constantine y Lockwood (*Diseño Centrado en el Uso*), el *Ciclo de Vida de la ingeniería en usabilidad* de Mayhew, el *Ingeniería de Usabilidad Basada en Escenarios* de Carroll y Rosson y la propuesta del *Proceso del Diseño Centrado en el Usuario* de las Normas ISO.

### 3.2.1. La ingeniería de usabilidad según Nielsen

Aunque ha sido superado por intentos posteriores, incluimos aquí una brevísima descripción del ciclo de vida para la ingeniería de usabilidad propuesto por Nielsen en *Usability Engineering* [Nielsen 1993], porque contiene las semillas de procesos que luego vieron la luz a raíz de la difusión de este trabajo.

Nielsen enfatiza que la ingeniería de usabilidad no es algo que se consiga en un solo intento, sino un conjunto de actividades que tienen lugar a lo largo de todo el ciclo de vida de un producto, con actividades importante que tienen lugar en las etapas tempranas del diseño, mucho antes de que la interfaz de usuario sea siquiera imaginada ([Nielsen 1993], pág. 71). Las principales etapas de un modelo de ciclo de vida para la ingeniería de usabilidad señaladas en esa obra son:

1. Conocer al usuario (características de los usuarios individuales, tareas actuales y deseadas de los usuarios, análisis de funciones, evolución del usuario y su trabajo, análisis de la competencia)
2. Establecer metas de usabilidad.
3. Diseño en paralelo.
4. Diseño participativo.
5. Diseño coordinado de la interfaz completa.
6. Aplicar guías y heurísticas para análisis.
7. Prototipar.
8. Pruebas empíricas.
9. Diseño iterativo, capturar el rationale del diseño.
10. Reunir feedback del uso real.

### 3.2.2. El ciclo de vida en estrella de Hix y Hartson

El **Ciclo de vida en Estrella** [Hix and Hartson 1993] es un proceso centrado en usuario que establece las principales actividades de usabilidad.

No prescribe un orden particular para estas actividades, pero reserva un rol prominente a la evaluación de usabilidad, colocada en el centro de la estrella como puede observarse en la Figura 3.1.

Hix y Hartson describen los caminos de comunicación que debería seguir entre actividades de usabilidad y diseño de software. Separan estrictamente el desarrollo de la IU del desarrollo del resto del software, con dos actividades que las conectan: el análisis del sistema y el testing o evaluación. Constituye una aproximación simplista a la integración entre SE y HCI, pero los autores reconocen que la investigación es necesaria para entender mejor y dar soporte a las necesidades reales de comunicación en procesos complejos.

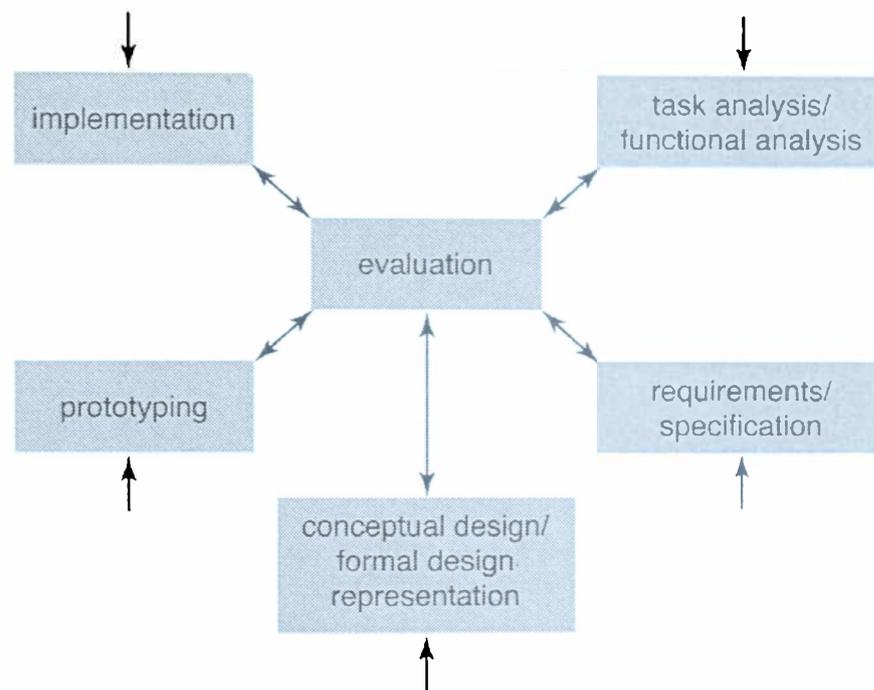


Figura 3.1: Modelo en Estrella, de [Hix and Hartson 1993]

### 3.2.3. Diseño Centrado en el Uso de Constantine

[Constantine and Lockwood 1999] proponen el *Método de Diseño Centrado en el Uso* como una colección de actividades coordinadas que contribuyen a la usabilidad. El modelo de estas actividades incluye algunas que corresponden con un proceso de desarrollo de software más amplio, junto con actividades exclusivamente de usabilidad. El modelo que Constantine y Lockwood proponen es atractivo para los ingenieros de software porque están más cercanos que otros a la clase de modelización usada en su disciplina. En particular, los *casos de uso esenciales*, que son la piedra angular de esta aproximación, son una reinterpretación de la técnica de los casos de uso, muy popular en la orientación a objetos.

Sostienen que más allá de todos los debates sobre terminologías y responsabilidades respecto de las interfaces entre las personas y el software, finalmente se requiere la solución de problemas de diseño en tres áreas íntimamente relacionadas: arquitectura, presentación e interacción. Estos tres aspectos del diseño de interfaces interactúan fuertemente entre sí y son finalmente inseparables. Cualquier elección en uno de los aspectos condiciona las posibilidades en los restantes.

Como su nombre lo indica, el centro de atención no está puesto en el usuario sino en el USO, es decir en las tareas que los usuarios desean realizar y cómo lo hacen. Esta diferencia de énfasis se traduce en diferentes prácticas con impactos significativo en el ciclo de desarrollo. La Tabla 3.1 resume las diferencias del Diseño Centrado en el Uso con el DCU según los autores [Constantine and Lockwood 2002].

Diseño centrado en el usuario	Diseño centrado en el uso
Foco en el usuario: experiencia del usuario, satisfacción del usuario	Foco en el uso: herramientas mejoradas para soportar el logro de tareas
Guiado por el input del usuario	Guiado por modelos
Involucramiento sustancial del usuario: estudios de usuarios, diseño participativo, retroalimentación de usuario, testeo de usuario	Involucramiento selectivo del usuario: modelización exploratoria, validación de modelos, inspecciones de usabilidad estructuradas
Descripción de usuarios, características de usuario	Modelos de relaciones de usuarios con el sistema
Modelos de diseño realistas	Modelos de diseño abstractos
Diseño por prototipado iterativo	Diseño a través de modelos
Procesos variados, a menudo informales o no especificados	Procesos sistemáticos, completamente especificados

Tabla 3.1: DCU vs Diseño Centrado en Uso

El Diseño Centrado en el Uso está dirigido por tres modelos abstractos, estrechamente relacionados: el modelo del rol, el modelo de la tarea y el modelo del contenido. El primero captura las características destacables del usuario que participan en su relación con el sistema. El segundo, la estructura de la tarea que el usuario debe realizar con el sistema. El modelo del contenido, representa los elementos y organización de la interfaz de usuario necesaria para dar soporte a las tareas identificadas [Constantine and Lockwood 2002].

La Figura 3.2 de [Constantine and Lockwood 2002] muestra el proceso lógico del método. Los tres modelos centrales al método se desarrollan en forma paralela y muy relacionada con otros tres: un modelo del dominio (en la forma de glosario, por ejemplo), un modelo de las reglas del negocio que da forma a la lógica subyacente y restricciones de la aplicación y un modelo operacional que captura las características importantes del entorno de trabajo o el contexto operativo.

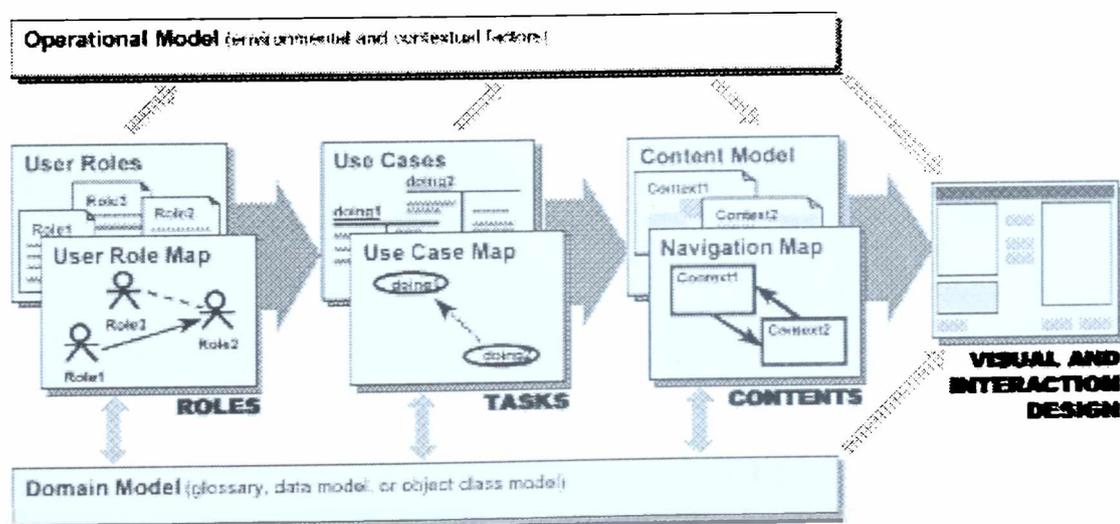


Figura 3.2: Procesos lógicos en el Diseño Centrado en Uso

### 3.2.4. Ciclo de vida de Ingeniería de Usabilidad por Mayhew

Deborah Mayhew propone el Ciclo de vida de la ingeniería de usabilidad para el desarrollo de interfaces de usuario usables [Mayhew 1999]. El proceso estructura actividades en tres fases: análisis de requerimientos, diseño-test-desarrollo e instalación. Extrañamente, esta aproximación se apoya en un ciclo de vida general muy similar al de cascada, aunque dentro de cada fase prevalezca el enfoque iterativo.

Aunque Mayhew sostiene que el método está dirigido sólo al desarrollo de la interfaz de usuario, las actividades incluidas en el ciclo de vida abrazan una parte importante de las tareas relacionadas con requerimientos. Se pueden identificar vínculos con la Ingeniería de Software orientada a objetos y con los métodos de prototipado rápido. La autora reconoce que la integración de ingeniería de usabilidad con ingeniería de software debe ser personalizada en cada organización o proyecto y que las superposiciones entre ambas no están completamente claras.

El método de Mayhew se desarrolla como muestra la Figura 3.3:

- Análisis de requerimientos: con la construcción del perfil del usuario, análisis en contexto de las tareas, establecimiento de las metas de usabilidad hasta elaborar una Guía de Principios de diseño para el proyecto.
- Diseño, testing y desarrollo estructurado en tres niveles con nivel de abstracción decreciente. En el primer nivel incluye la reingeniería del trabajo, un diseño del modelo conceptual, elaboración de maquetas y prototipos, evaluación iterativa. En el segundo, se establecen los estándares y guías: diseño, prototipado y evaluación de pantallas y guías de estilo. Finalmente, en el tercer nivel se realiza el diseño detallado de la IU y su evaluación en un proceso iterativo.
- Instalación, permite obtener el feedback del usuario sobre el producto ya instalado para poder mejorarlo.

Algunos puntos que Mayhew plantea como la filosofía detrás del Ciclo de vida de la ingeniería de usabilidad permiten comprender mejor los alcances de esta disciplina:

1. El diseño de la IU es la clave
2. La integración de la Ingeniería de Usabilidad con la ingeniería de software tiene que ser personalizada y adaptada a los requerimientos del desarrollo y a los recursos disponibles. Por ejemplo el análisis de sistemas tradicional y el Análisis de Tareas en Contexto son actividades altamente relacionadas pero no son la misma cosa y requieren muy diferentes clases de expertise. La forma de integrarlas para evitar duplicación de esfuerzos y producir los resultados deseados es algo que está pendiente de resolución. Mientras tanto, la forma de integrar la Ingeniería de Usabilidad en la metodología del proyecto requiere una adaptación en cada proyecto.
3. El análisis de requerimientos se compensa, es preferible un cuidadoso (y a menudo colaborativo) diseño y una evaluación informal, que un diseño casual y una evaluación formal.

### 3.2.5. Los escenarios de Carroll y Rosson

John Carroll y Mary Beth Rosson propusieron como elemento nucleador para el diseño de sistemas usables el trabajo con escenarios. En [Rosson and Carroll 2002] presentan el método, cuyo proceso general se muestra en la Figura 3.4.

Los autores definen un *escenario* se como una historia o relato sobre la manera en que la gente realiza una actividad, mientras que un *escenario-problema* es una historia sobre el dominio del problema tal como existe antes de introducir la nueva tecnología.

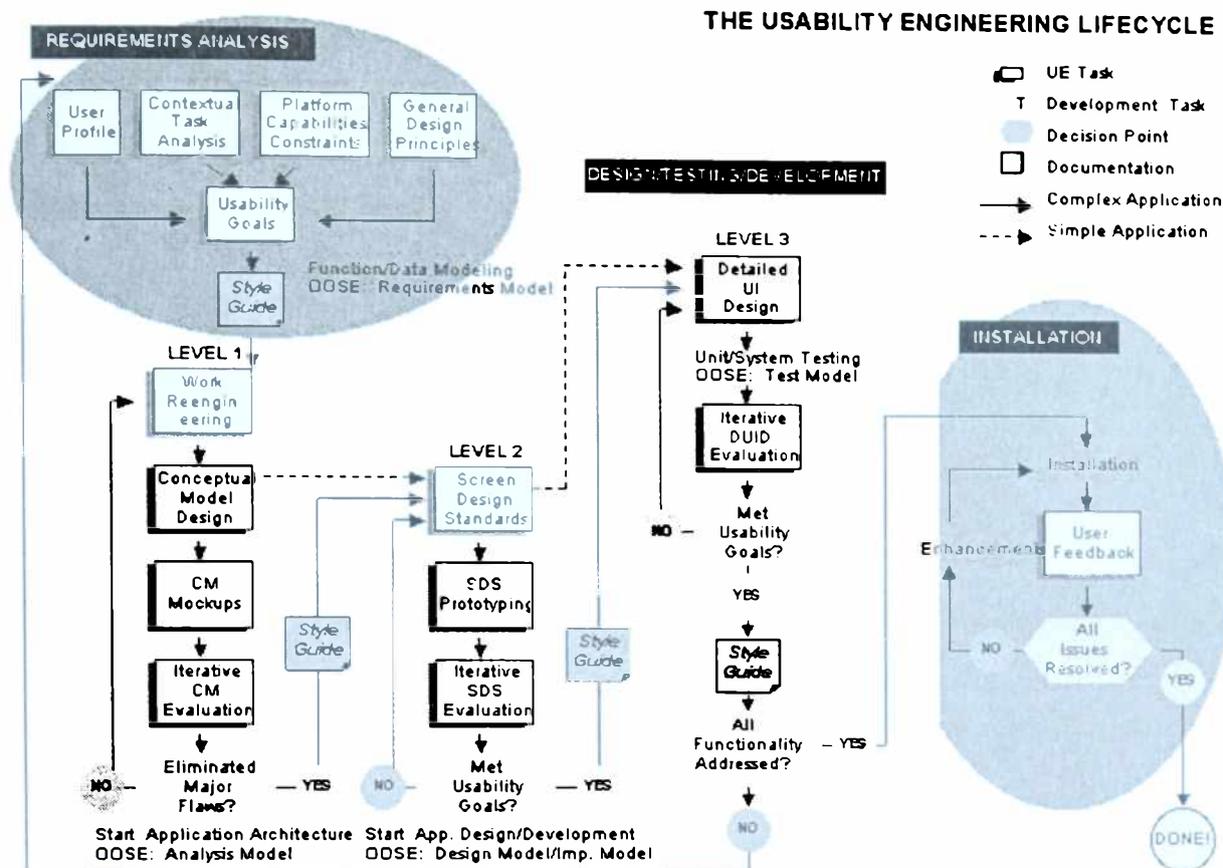


Figura 3.3: Ciclo de ingeniería de Usabilidad según [Mayhew 1999]

A partir de la idea de considerar que los escenarios constituyen una manera adecuada de administrar los compromisos existentes en la ingeniería de usabilidad (tomar decisiones al mismo tiempo que se deja el diseño abierto, balancear acción y reflexión, analizar el uso actual pero permitir que evolucione, etc.), Rosson y Carroll plantean un framework de proceso de tres etapas: análisis, diseño, evaluación de prototipos.

### 3.2.6. El proceso DCU de ISO

La ISO estableció en su estándar ISO 13407 [ISO/IEC 1999] una guía de referencia sobre actividades de diseño a lo largo de todo el ciclo de vida de sistemas interactivos basados en software, enfoque claramente centrado en usuarios.

La Norma ISO 13407 que fue establecida en 1999, está dirigida a quienes conducen los procesos de diseño y aunque no brinda ningún detalle respecto de métodos o técnicas específicas a utilizar propone un ciclo de vida completo para el desarrollo de sistemas con un enfoque centrado en usuarios. ISO 13407 identifica cuatro principios centrales que caracterizan el diseño centrado en el usuario y que no están ligados a ninguna fase específica del ciclo de desarrollo:

- inclusión activa de los usuarios y una clara comprensión de los mismos y de los requerimientos de las tareas
- distribución adecuada de funciones entre el usuario y la tecnología
- iteración de soluciones de diseño
- diseño multidisciplinar

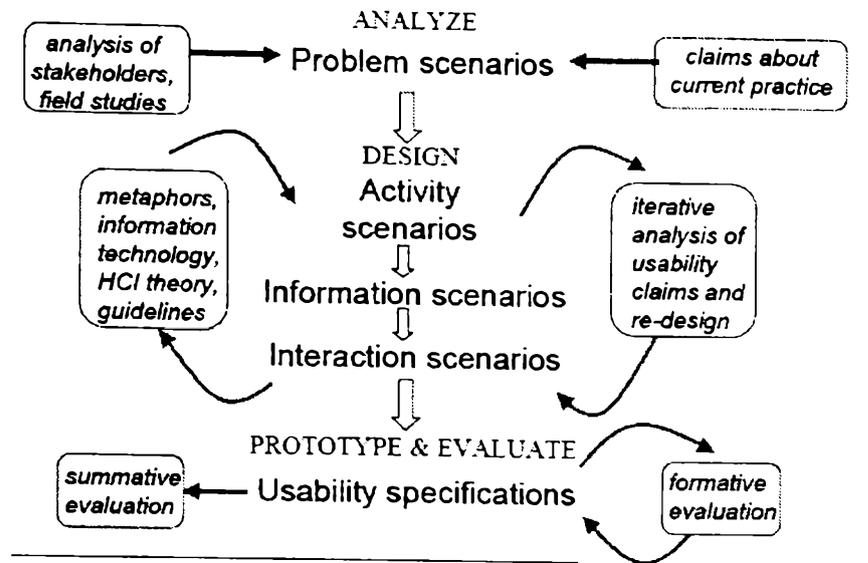


Figura 3.4: Ingeniería de usabilidad basada en escenarios, según [Rosson and Carroll 2002]

La estructura del proceso DCU, con su naturaleza fuertemente iterativa se ilustra en la figura 3.5, extraída de la propia ISO 13407.

La norma comienza señalando que en todas las actividades de desarrollo de sistemas (análisis, diseño, testing, etc) debe planificarse la inclusión de tareas centradas en las personas. Este requerimiento, que claramente excede la mera integración de actividades de usabilidad al diseñar la interfaz de usuario, está reclamando una hoja de ruta para adaptar las técnicas de usabilidad en el proceso general de desarrollo de software. Retomaremos este requerimiento al momento de plantear la contribución de esta tesis.

Luego, ISO 13407 establece el contenido de los cuatro bloques principales de actividades:

**Entender y especificar el contexto de uso** Se trata de comprender las características de los usuarios, las tareas y el entorno físico y de organización definen el contexto en el que se utilizará el sistema. Es importante entender e identificar los detalles de este contexto para guiar las primeras decisiones de diseño y proveer una base para la evaluación.

**Especificar los requerimientos de usuarios y la organización** Esta actividad debería crear un planteo explícito de los requerimientos de los usuarios y la organización relacionados con la descripción del contexto de uso. Los siguientes aspectos que deben considerarse incluyen por ejemplo la performance requerida del sistema nuevo versus objetivos financieros y operativos; requerimientos legislativos y normativos relevantes, incluyendo cuestiones de seguridad y salud; cooperación y comunicación entre usuarios y otras partes relevantes; la performance deseada de las tareas; interfaz hombre-computadora y diseño de puesto de trabajo, etc.

**Producir soluciones de diseño** El corazón de todo proceso DCU es la generación en forma iterativa e incremental de soluciones de diseño a las situaciones problemáticas planteadas, utilizando las prácticas del estado del arte, experiencia y el conocimiento de los participantes. ISO 13407 adhiere al enfoque de diseño de interacción que parte “desde cero”, proponiendo la mejor división de tareas entre tecnología y personas, sin dar por sentada ninguna solución previa y se apoya como actividad por antonomasia en el prototipado.

**Evaluar las soluciones de diseño contra los requerimientos de usuario y la organización**

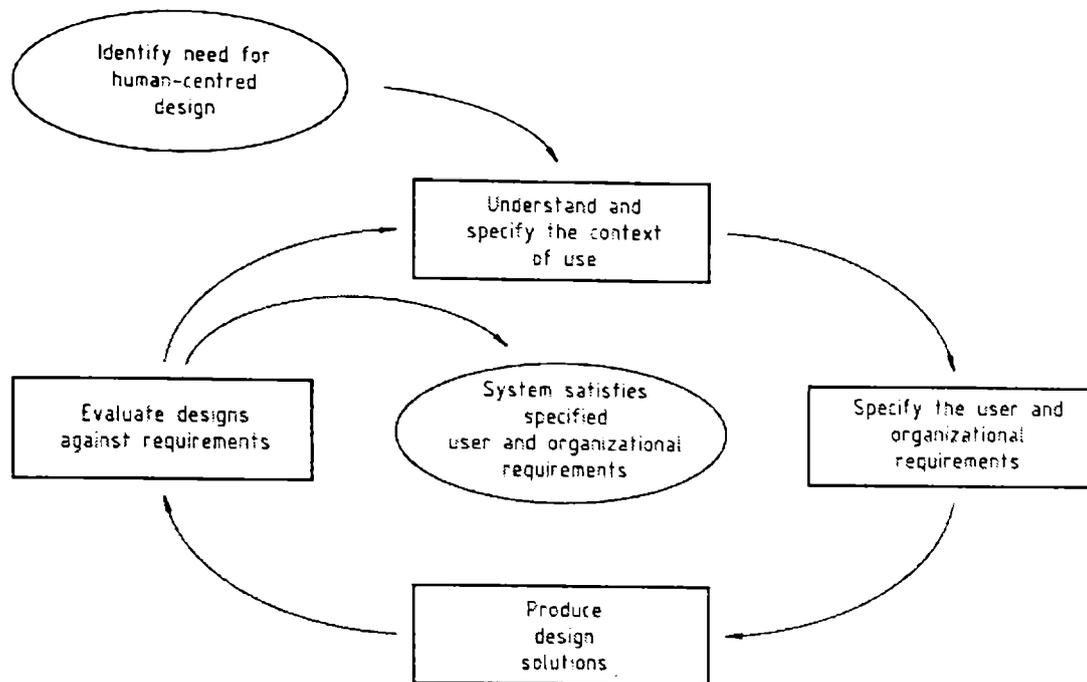


Figura 3.5: El proceso DCU en ISO 13407, [ISO/IEC 1999]

Se trata de reunir el mejor feedback posible sobre el diseño, a partir de los usuarios y representantes de todas las partes involucradas en la relación con el sistema final.

### 3.3. Aproximaciones desde la Ingeniería de Software

En esta sección repasaremos algunos de los intentos provenientes desde la Ingeniería de software por integrar usabilidad y diseño centrado en usuarios en métodos y procesos de desarrollo. Comenzamos con la propuesta de Maria Francesca Costabile, basada en un ciclo de cascada con retornos [Costabile 2001], luego revisamos el Proceso Unificado y su principal extensión para DCU, “El plugin de la experiencia de usuario para RUP” [IBM 2005b]. Este repaso continúa con el Dynamic Systems Development Method, conocido como DSDM [DSDM-Consortium 2010] y finaliza con Wisdom de Nunes y Cunha, inspirado en el Proceso Unificado [Nunes and Cunha 2001].

#### 3.3.1. La propuesta de Costabile

En [Costabile 2001] se ofrece una integración de prácticas centradas en usuario en el proceso de desarrollo de software con el objetivo de aumentar la usabilidad del producto final. Se concentra en tres principios del DCU: analizar los usuarios y las tareas, diseñar e implementar en forma iterativa e incremental a través de prototipos de complejidad creciente y evaluar los diseños con usuarios.

La propuesta de Costabile implica tomar un ciclo de vida de software y modificarlo para incluir actividades de usabilidad. Extrañamente se basa en el ciclo en cascada, al que agrega dos actividades exclusivas de usabilidad (analizar los usuarios y las tareas, realizar escenarios y especificaciones de interfaces de usuario) y extiende con dos momentos de prototipado y testing, como se muestra en la Figura 3.6. Como propone la posibilidad de retornar desde cualquier actividad a una fase previa sostiene que esto da soporte a la iteratividad que requiere un proceso centrado en usuarios.

La elección del ciclo de vida en cascada es la principal debilidad de esta propuesta. Aunque se puedan hacer retornos a fases previas, éstos están planteados como correcciones

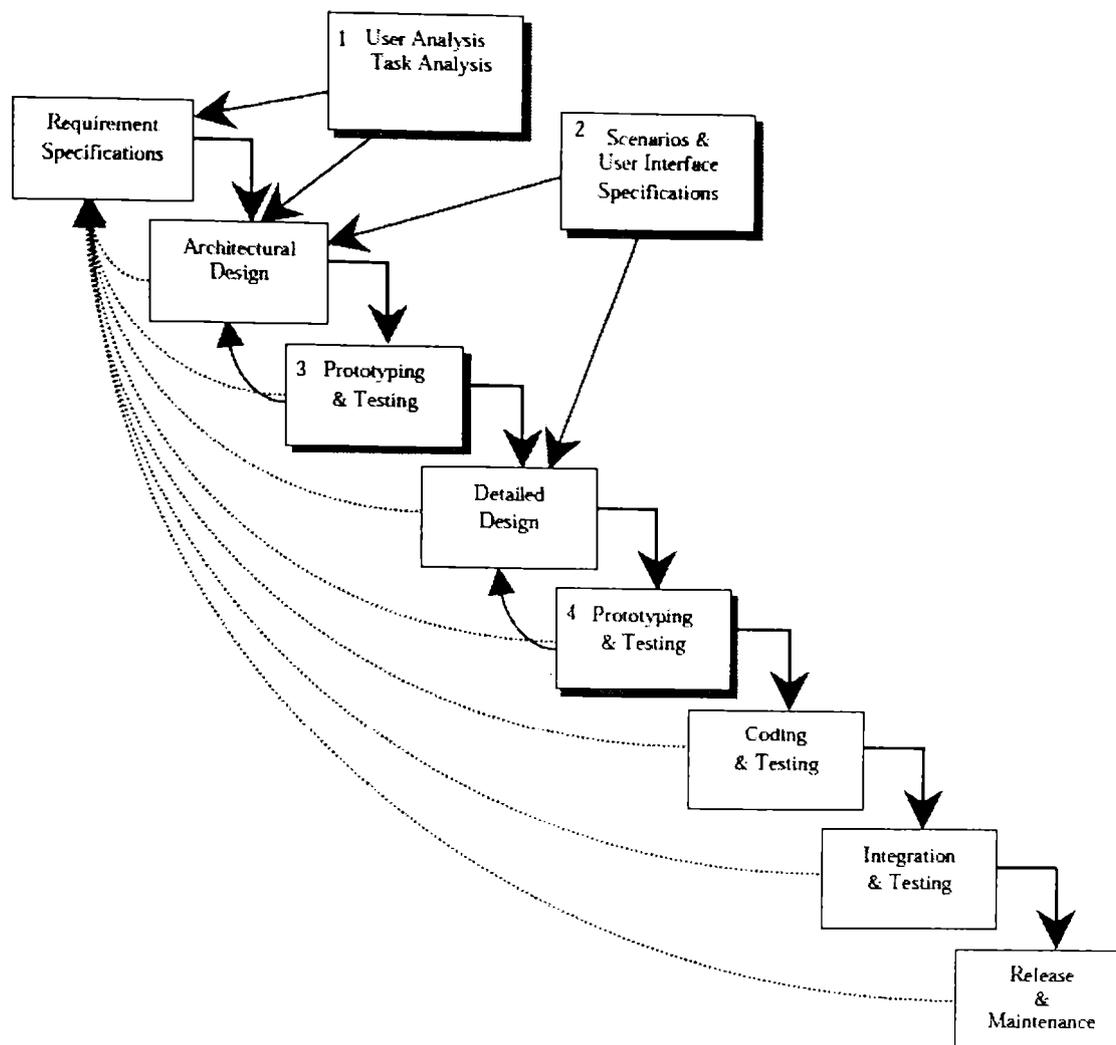


Figura 3.6: Inclusión de usabilidad en ciclo de vida de software según Costabile

de errores. La idea de iteración en DCU está más relacionada con el hecho de que no es posible congelar alguna versión de requerimientos o aún de solución parcial de diseño, sino que ambas especificación y resolución de problema avanzan juntas en el proceso. Larman ha mostrado claramente las fallas de este ciclo, que terminan conspirando contra el aprovechamiento de las ventajas de la iteración incremental: dificultad para mitigar riesgos, especulación en la elicitación de requerimientos y rigidez en su especificación, aumento de la complejidad por falta de adaptabilidad del proceso Larman [2003].

### 3.3.2. El Proceso Unificado

El Proceso Unificado [Jacobson et al. 1999] ha recibido una gran atención de parte de la comunidad de desarrolladores de software. Entre otras razones, porque sus principales autores y sponsors se cuentan entre los principales metodologistas de la orientación a objetos: James Rumbaugh, Ivar Jacobson y Grady Booch.

Este método organiza el desarrollo en cuatro fases dentro de las cuales se realizan iteraciones que generan valor en forma incremental. Se identifican algunos elementos básicos del método (roles, artefactos, tareas) que se organizan en disciplinas (requerimientos, análisis, diseño, testing, etc.).

La versión más difundida del Proceso Unificado y que se convirtió en el estándar de desarrollo iterativo es la conocida como RUP o Rational Unified Process [Krutchen 2000] (actualmente distribuida por IBM, antes por Rational Software Co.). La Figura

3.7 muestra la arquitectura general de RUP en los dos ejes: en el horizontal, el ciclo de vida en cuatro fases; en el vertical, las disciplinas que organizan las actividades por su “naturaleza” (en términos de RUP).

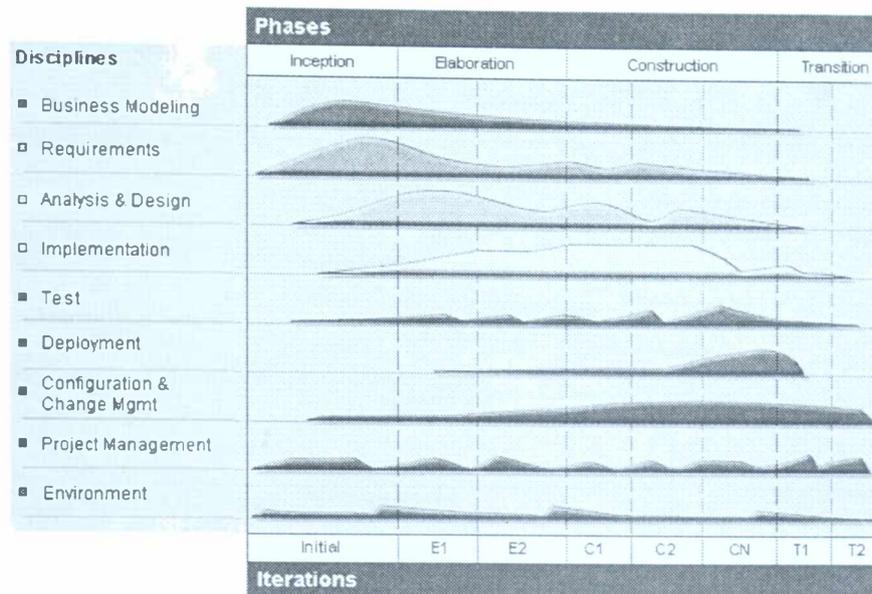


Figura 3.7: Arquitectura de procesos en RUP, según [IBM 2005a]

RUP tiene algunas de las características que son propias de un proceso centrado en usuario y utiliza productos de trabajo similares. Por ejemplo, es iterativo e incremental y los casos de uso modelan los requerimientos de forma similar al análisis de tareas para la comprensión del contexto de uso que propone ISO 13407. Sin embargo, es esencialmente un proceso centrado en la arquitectura y en la versión original no incluye ningún elemento específico para el desarrollo de la interacción de usuario.

El plugin de la Experiencia de Usuario para RUP (UX-RUP) [IBM 2005b] intenta integrar en RUP el trabajo realizado en el dominio del desarrollo web relacionado con los contenidos y concepto de sitio. Está basado en el trabajo de Jim Conallen sobre modelización de aplicaciones web [Conallen 2002]. De acuerdo con Conallen, el término Experiencia de Usuario se utiliza para describir el equipo y las actividades de los especialistas responsables por mantener la interfaz de usuario consistente con los paradigmas actuales y apropiada al contexto en el cual se espera que se utilice el sistema.

Incorpora como producto de trabajo básico el Modelo de la UX, que intenta describir los elementos de la experiencia del usuario con el sistema (las pantallas y formularios), el contenido dinámico y los caminos de navegación entre los elementos para ejecutar la funcionalidad del sistema. Este modelo está constituido por tres productos: un conjunto de Elementos de UX (pantallas y formularios de ingreso), un conjunto de Storyboards de UX (que representan la implementación, en términos de interacción de usuario, de los Casos de Uso) y un Mapa de navegación entre todos los elementos de UX del sistema. La Figura 7.21 muestra un ejemplo de Mapa de Navegación.

Este plugin es claramente un avance en la inclusión de DCU y usabilidad en RUP. Sin embargo, está acotado a algunas partes del proceso y unos pocos modelos de trabajo.

### 3.3.3. DSDM

El DSDM (Dynamic Systems Development Method) es un framework de métodos para el desarrollo de software de manera ágil. Se basa en el involucramiento activo del usuario, un desarrollo iterativo e incremental y sensible a los cambios de requerimientos [DSDM-Consortium 2010].

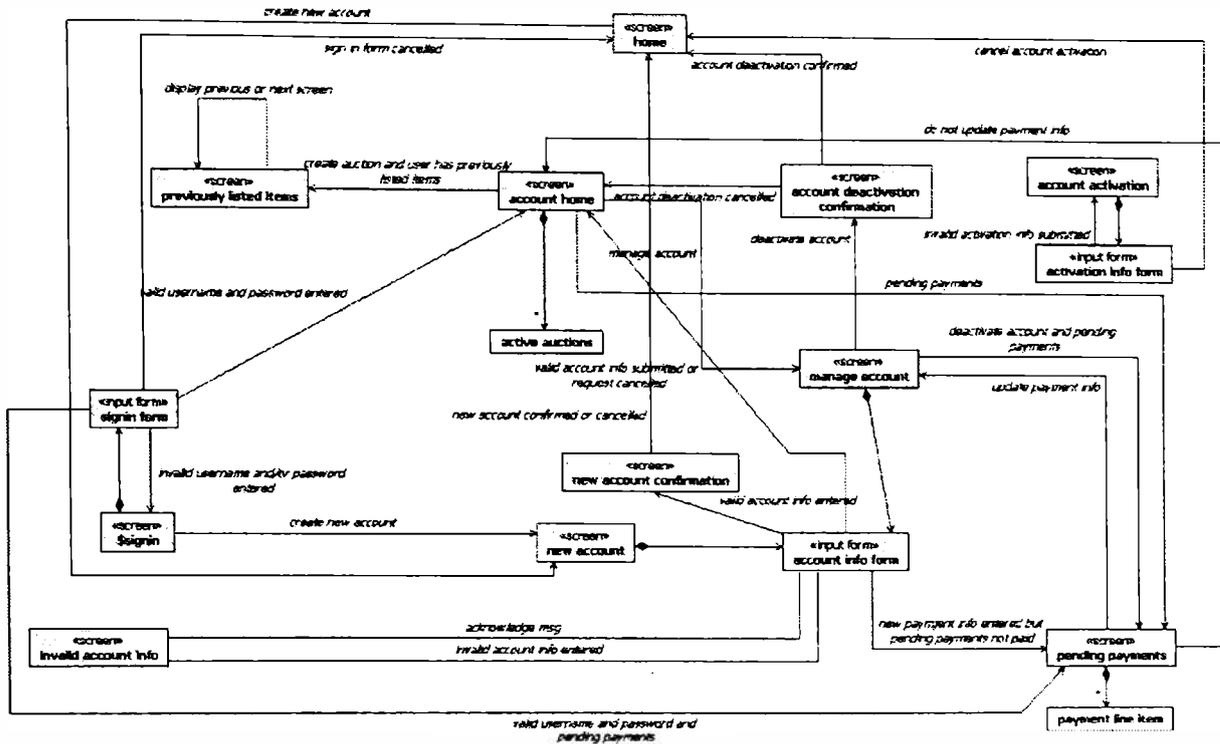


Figura 3.8: Ejemplo de Mapa de Navegación en el Plugin UX de RUP, según [IBM 2005b]

Fue desarrollado y es mantenido por el Consorcio DSDM, una alianza entre proveedores y expertos de desarrollo de sistemas de información del Reino Unido. La versión actual, la 4.2, fue lanzada en 2003.

El ciclo de vida de DSDM consiste en tres fases: fase del pre-proyecto, fase del ciclo de vida del proyecto, y fase del post-proyecto. La fase del ciclo de vida del proyecto se subdivide en 5 etapas: estudio de viabilidad, estudio de la empresa, iteración del modelo funcional, diseño e iteración de la estructura, e implementación. La figura 3.9 muestra el ciclo de vida en la fase Proyecto.

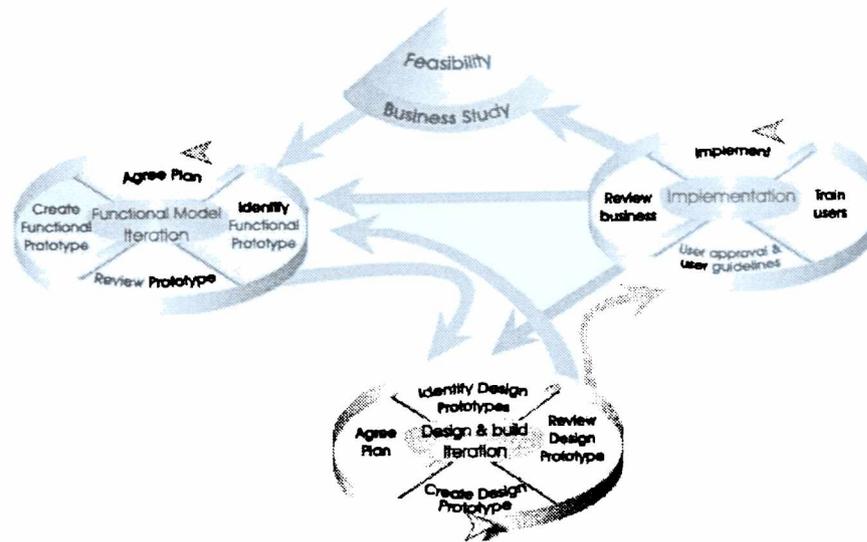


Figura 3.9: Ciclo de vida DSDM

DSDM está sustentado en nueve principios, varios de ellos compartidos con el diseño centrado en usuario tales como involucrar al usuario, dar poder al equipo para tomar decisiones, entrega frecuente de productos, desarrollo iterativo e incremental, priorizar el

valor entregado como criterio de aceptación.

### 3.3.4. Wisdom

Desarrollado por Nunes y Cunha, WISDOM (Whitewater Interactive System development with Object Models) [Nunes and Cunha 2001] se plantea como un método liviano alrededor de tres componentes principales: un framework de procesos de desarrollo, un conjunto de modelos UML y una extensión de la notación UML para permitir la representación y modelización de los componentes interactivos.

El ciclo de vida del proceso de WISDOM es similar al Proceso Unificado (utiliza las mismas cuatro fases evolutivas) y organiza las actividades en cuatro flujos de trabajo: requerimientos, análisis, diseño y evaluación. En el corazón del proceso Nunes y Cunha ubican al prototipado evolutivo que avanza en sucesivas refinaciones hacia el sistema final. Al igual que el PU, WISDOM utiliza los Casos de Uso como herramienta de requerimientos.

Las Figuras 3.10 y 3.11 muestran el ciclo de vida propuesto y algunas de las extensiones a UML que se introducen para generar los modelos de la interacción de usuario.

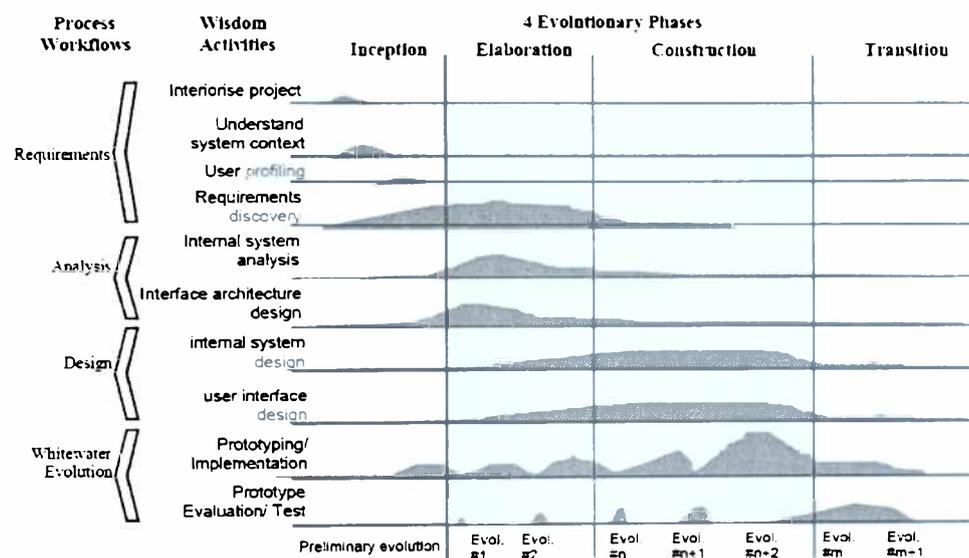


Figura 3.10: Fases del proceso en WISDOM

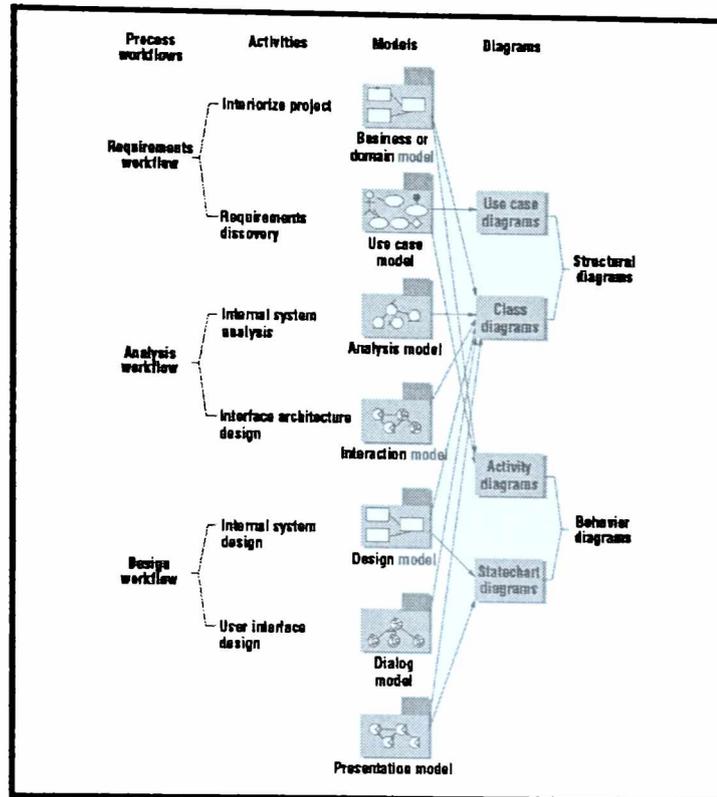


Figura 3.11: Flujos de tareas, actividades, modelos y diagramas en WISDOM

### 3.4. Oportunidad de intervención

En este capítulo, se han presentado diferentes propuestas de integrar en procesos de diseño y construcción de software tareas vinculadas con la obtención de usabilidad en el producto final, provenientes tanto desde el campo de la HCI como de la Ingeniería de Software.

Antes habíamos mostrado que el logro de buenos niveles de usabilidad guarda una relación estrecha con procesos centrados en usuario, que a su vez se caracterizan por ser esencialmente iterativos. También mencionamos la importancia de un abordaje que no se concentre sólo en la interfaz de usuario, sino en todas las interacciones que el producto habilita o restringe. Finalmente, señalamos que la efectividad y eficiencia de los procesos de desarrollo centrados en usuario se pueden aumentar mejorando la capacidad y madurez en usabilidad de una organización.

Todas las propuestas que presentamos incluyen en alguna medida el carácter iterativo en el ciclo de vida y la participación más o menos comprometida de los usuarios finales (ya comentamos la falencia de Costabile al presentar como iterativo un ciclo en cascada con retornos).

Más despareja resulta la extensión de aspectos del sistema que están cubiertos por tareas de usabilidad. Sólo DSDM, Constantine y el Proceso ISO plantean claramente la necesidad de abarcar todo el sistema o producto. El resto restringe en algún punto la tarea a los aspectos vinculados explícitamente con las IUs.

Pero el punto que a nuestro juicio compromete más la posibilidad de mejorar de manera previsible la obtención de niveles de usabilidad en los productos de software es la escasa o nula guía que estas propuestas brindan para integrarlas con la ingeniería del software en las actividades de mejoras de procesos. En efecto, es clara la carencia en esas propuestas de alguna forma de hoja de ruta que además de incluir actividades de usabilidad en un ciclo de vida de desarrollo, guíe al equipo en la forma de incrementar la capacidad en usabilidad.

Creemos que una forma de contribuir a superar esta carencia es dar soporte a la integración de diseño y desarrollo centrado en usuario en procesos de ingeniería de software de manera tal que se puedan mejorar los niveles de capacidad y madurez en usabilidad de una organización.

Vamos a tomar como base uno de las metodologías de desarrollo que alcanzó mayor difusión en la construcción de software, como es el Proceso Unificado. En particular, utilizaremos la versión liberada como OpenUP por la Fundación Eclipse. Nos proponemos analizarla para encontrar los huecos que deben ser cubiertos para mejorar las actividades de DCU en ese proceso y desarrollar las extensiones que sean necesarias.

Como guía para esos cambios emplearemos el Modelo de Capacidad y Madurez en Usabilidad propuesto por la ISO en los tres primeros niveles de Capacidad (El proceso realizado, El proceso gestionado y el Proceso establecido). Intentaremos extender la Libería de Procesos de Eclipse para permitir instancias de OpenUP que puedan satisfacer a los niveles 1 a 3 del MMU-ISO.

## Parte II

# Capacidad y Madurez en Usabilidad con el Proceso Unificado



## Capítulo 4

# Proceso Unificado open source: OpenUP

### 4.1. Introducción

Para nuestra propuesta de inclusión de DCU en un proceso de desarrollo de software tomamos como base la versión open source del Proceso Unificado, que fuera liberada por la Fundación Eclipse a partir de una contribución por parte de IBM de los elementos básicos de RUP, en el marco del proyecto Eclipse Process Framework (EPF) [ECLIPSE 2006].

El proyecto EPF tiene dos objetivos: a) proveer un framework extensible y herramientas de ejemplo para la ingeniería de procesos de software (autoría de métodos y procesos, gestión de bibliotecas, configuración y publicación de procesos) y b) proveer contenido de ejemplo para procesos de desarrollo y gestión de software que sea extensible y soporte el desarrollo iterativo, ágil e incremental y que se sirva para un conjunto amplio de plataformas y aplicaciones.

El framework, que describiremos en detalle en la sección siguiente, se compone de un meta-modelo basado en la versión 2.0 del SPEM de la OMG [OMG 2008] (que a su vez reconoce entre las fuentes a la Unified Method Architecture de IBM) y un núcleo extensible de herramientas (funcionalidad básica y una API que permita la autoría de métodos y procesos). OpenUP/Basic, como veremos, es una configuración de procesos basada en el Proceso Unificado y que utiliza los elementos *core* provistos por el EPF. Una de las herramientas de ejemplo liberadas, el EPF Composer será utilizado para la elaboración de las configuraciones del proceso que integren el DCU.

En las secciones siguientes de este capítulo presentamos el EPF y la versión básica de OpenUP.

### 4.2. El Eclipse Process Framework

El corazón del EPF se denomina **Framework Unificado de Métodos (UMF)**. Se trata de un framework que se basa en un conjunto de prácticas provenientes de diferentes contextos y desarrolladas por diferentes organizaciones que comparten una infraestructura común. En cierto sentido, puede considerarse como un “lenguaje común” que permite la interoperación de prácticas. Un *framework de prácticas* organiza una biblioteca de métodos como un conjunto de prácticas con alta cohesión y bajo acoplamiento. El contenido se comparte entre las prácticas, se captura en elementos *core*, categorizados en Slots de Productos de Trabajo (WPS). Las configuraciones de prácticas (incluyendo procesos

transversales a varias de ellas) se ensamblan seleccionando un subconjunto específico de esas prácticas. La Figura 4.1 muestra gráficamente esta descripción.

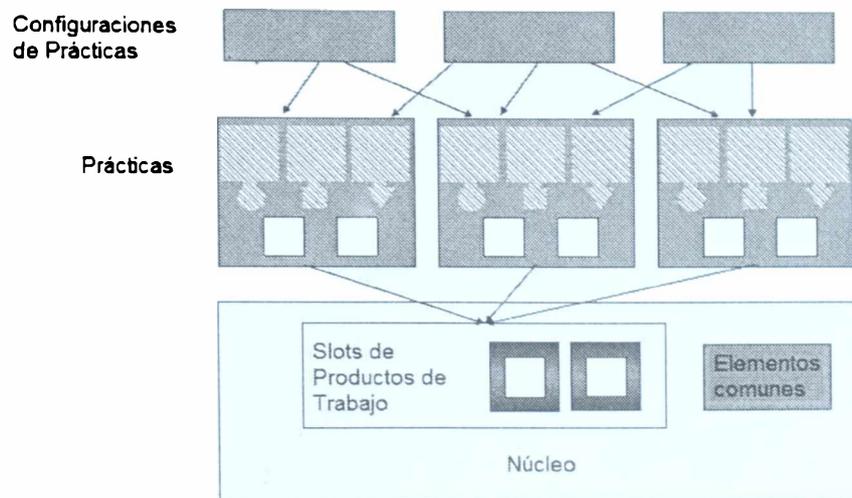


Figura 4.1: Esquema de un Framework de Prácticas

Ya señalamos que el metamodelo empleado en EPF y EPL el propuesto por el Object Management Group en la Versión 2.0 de su Software & Systems Process Engineering Meta-Model Specification [OMG 2008]. La concepción y el marco de uso del SPEM 2.0, adoptada por el EPF, se muestra en la Figura 4.2. Se trata de proveer una representación estandarizada y bibliotecas de contenido de método que sea reutilizable, dar soporte para una actividad sistemática de desarrollo, gestión y crecimiento de procesos de desarrollo de software, facilitar el despliegue de un subconjunto de contenidos y procesos necesarios mediante la configuración de procesos específicos y dar un respaldo adecuado a la implementación concreta de los procesos configurados.

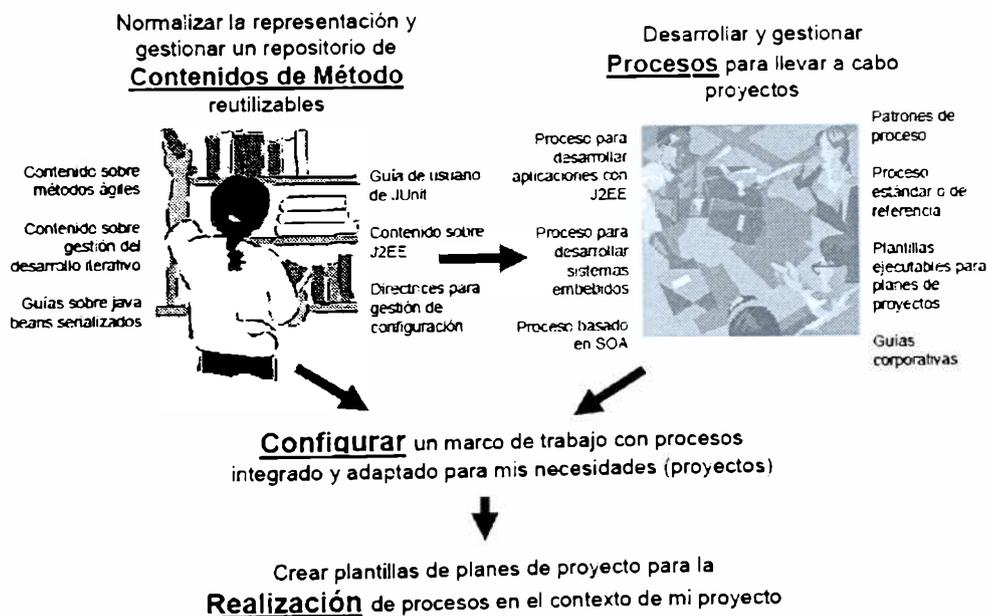


Figura 4.2: Marco conceptual para uso de SPEM 2.0

Este meta-modelo, que provee el lenguaje para describir los contenidos y procesos del método, separa las definiciones de Contenido del Método de su aplicación en Procesos de entrega, tal como muestra la Figura 4.3. El Contenido se manifiesta en principio mediante definiciones de productos de trabajo, de roles, de tareas y orientaciones. Estas últimas

(que pueden ser guías, listas de control, ejemplos, hojas de ruta, etc) están en la intersección de Contenido y Procesos porque pueden considerarse como un background para un contenido tanto como para un proceso específico. A la derecha del diagrama están los procesos definidos en SPEM 2.0. El elemento principal es la Actividad, que puede anidarse para definir estructuras de desglose de tareas (WBS) o relacionarse con otras actividades para definir un flujo de trabajo. Las Actividades también tienen referencias a los Contenidos de Método, representadas con el concepto "usa". Las Actividades se utilizan para definir Procesos.

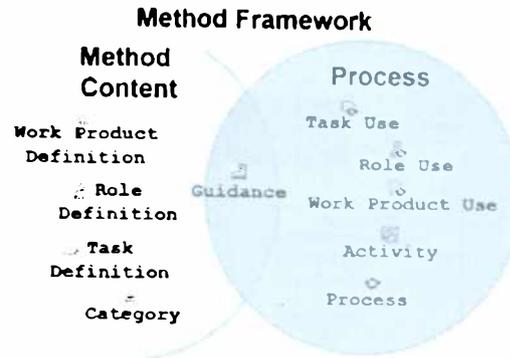


Figura 4.3: Elementos de Metamodelo SPEM 2.0

La referencia completa al SPEM 2.0 se puede encontrar en [OMG 2008], transcribimos aquí sólo las definiciones breves de los elementos de contenido y proceso que describiremos más adelante dentro de las Prácticas de la EPL. Como contexto de estas definiciones, mostramos con las Figuras 4.5 y 4.4 los estereotipos UML que emplea SPEM 2.0 en el diseño de los Paquetes "Estructura de Procesos" y "Estructura de Contenido de Método".

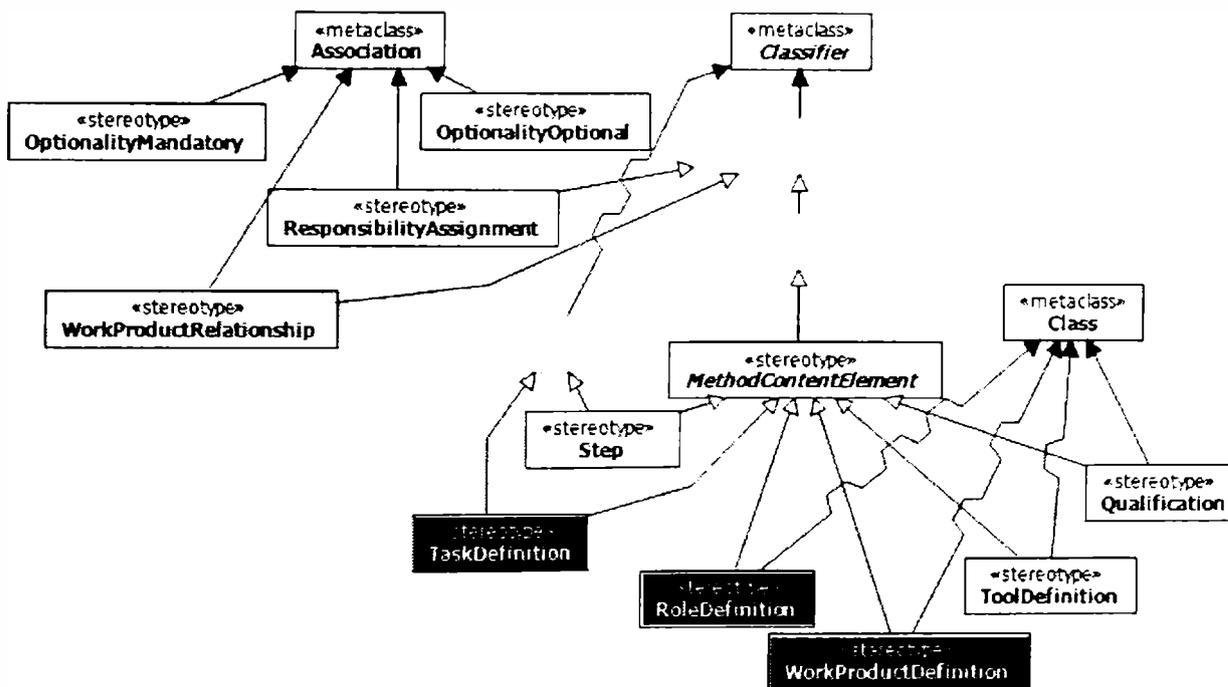


Figura 4.4: Perfiles UML definidos en SPEM 2.0 para el paquete Contenido de Método

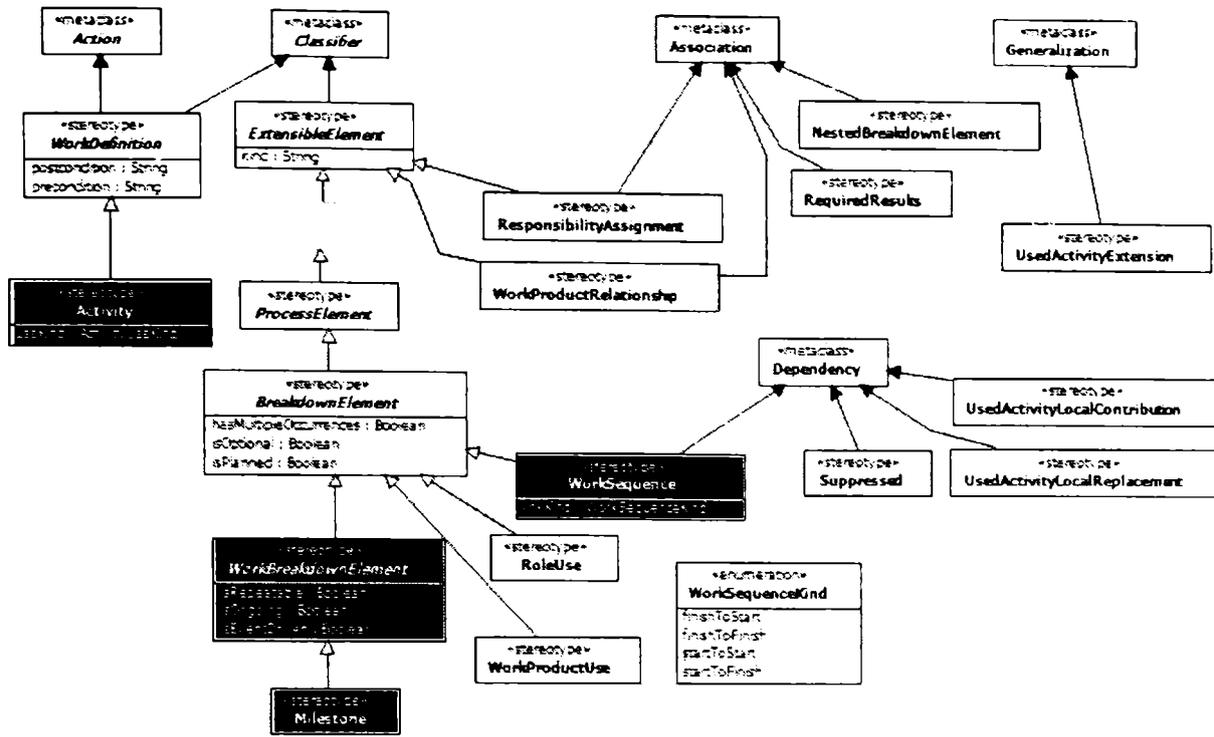


Figura 4.5: Perfiles UML definidos para el paquete Estructura de Procesos en SPEM 2.0

## Los elementos de contenido

### Definición de Rol

Un rol se define como un conjunto de habilidades, competencias y responsabilidades relacionadas. Las personas o las herramientas que desempeñan un rol realizan tareas y generan productos de trabajo. Para algunas tareas y productos de trabajo los que desempeñan roles son directamente responsables de realizar las tareas y generar los productos. Para otras, simplemente colaboran para alcanzar la meta.

Un rol puede ser desempeñado por una persona o un conjunto de personas y a su vez una misma persona o conjunto puede desempeñar varios roles. Por ejemplo, hay dos roles en las diferentes versiones del Proceso Unificado. Gerente de Proyecto e Ingeniero de Procesos que en proyectos pequeños pueden ser desempeñados por la misma persona, mientras que en proyectos de gran magnitud es muy probable que existan varias personas desempeñando cada uno de esos roles.

La Figura 4.6 muestra las relaciones entre los Rol, Tarea y Productos de trabajo.

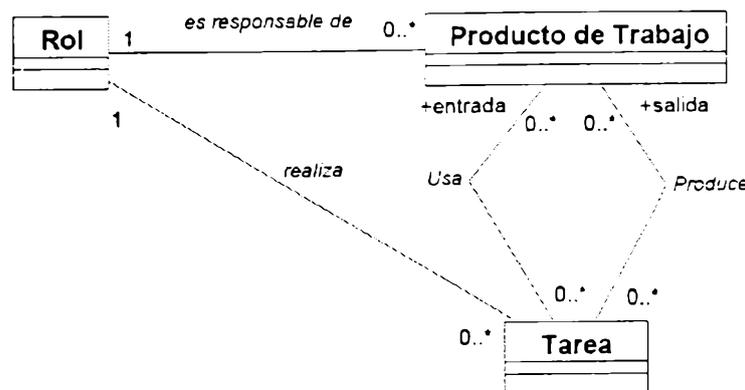


Figura 4.6: Relaciones entre definiciones de Rol, Tarea y Producto de trabajo

## Definición de Producto de trabajo

Un Producto de trabajo es un elemento del contenido del método que es usado, modificado y producido por una Definición de Tarea. Sólo un rol es responsable por cada producto de trabajo.

SPEM 2.0 distingue tres clases de productos de trabajo: artefacto, resultado y entregable. Los artefactos son tangibles y pueden estar incluidos en otros artefactos. Por ejemplo, un plan de desarrollo es un artefacto que puede contener una lista de riesgos, entre otros artefactos. Un resultado habitualmente es intangible y no reusable. Por ejemplo, podría ser la mejora en performance o la instalación de una herramienta. Un entregable es el valor provisto a los involucrados internos o externos y se compone de otros dos productos de trabajo: artefactos y resultados.

## Definición de Tarea

Una Definición de Tarea describe el trabajo realizado por instancias de Definición de Rol. Está asociada con Productos de Trabajo de entrada y salida y dirigida por una meta.

Puede considerarse como la unidad de trabajo asignable. La tarea describe el trabajo a ser realizado y comúnmente un conjunto opcional de pasos. Las tareas usualmente duran entre unas pocas horas y unos días y afectan solamente un número pequeño de Productos de trabajo. Debido a su granularidad, las tareas a menudo se repiten en iteraciones y pueden ser muy pequeñas para la planificación. Si son muy grandes, es conveniente desglosarlas en Pasos (una Definición de Paso representa la unidad más pequeña de trabajo a realizar).

Por ejemplo, la Definición de Tarea “Desarrollar el Modelo de Casos de Uso” describe el trabajo necesario para completar ese modelo: identificar y nombrar casos de uso y actores, escribir descripciones breves, modelar los CU y sus relaciones, completar las descripciones detalladas del flujo básico y los flujos alternativos, etc. Todas estas partes (Pasos) contribuyen a la meta de desarrollar el modelo de casos de uso, pero pueden realizarse en diferentes puntos de un proceso. Al comienzo se pueden identificar y nombrar los CU. Mucho más tarde en el proceso se completarán las descripciones detalladas de todos los flujos. El conjunto de los Pasos describe el método para desarrollar un Modelo de Casos de Uso. Aplicarlo en un proceso consiste en definir cuáles pasos se completan antes de pasar de una iteración a la siguiente.

## Los elementos de proceso

Los elementos de proceso están organizados a partir de la definición de un **Elemento de desglose** (*Breakdown element*). Se trata de una generalización abstracta de cualquier tipo de elemento que aparece en un proceso. Tiene tres propiedades importantes: a) admiten repeticiones en un mismo proceso, b) son opcionales y c) es planificado (se incluye al generar un plan de proyecto).

Dentro de los Elementos de Desglose, se cuentan los **Elementos de Desglose de Trabajo** (Actividades e Hitos). A la posibilidad de repetición que heredan de los Elementos de Desglose (y que permiten, por ejemplo, la realización de iteraciones), le suman una naturaleza continua (sin duración fija o estado final) y que están condicionados por sucesos, es decir que inicia porque un evento especial ha sucedido, no porque estaba programado en un cronograma.

Una **Actividad** constituye la unidad básica de trabajo dentro de un Proceso. Está asociada con Productos de Trabajo y con Roles. El meta-modelo soporta el anidamiento de Actividades, de modo que forman una WBS. Como una actividad puede contener otras actividades, el ingeniero de de proceso puede crear una jeraquía tan compleja como necesite.

El Flujo entre los Elementos de desglose de Trabajo se representa mediante **Secuencias de trabajo**. Cada secuencia conecta dos Elementos de Desglose de trabajo: un predecesor y un sucesor.

La Figura 4.7 ejemplifica una WBS asociada a un Elemento de Desglose de Trabajo (el Patrón de entrega de Fases para el ciclo de vida riesgo-valor).

Breakdown Element	Steps	Index	Predecessors	Model	Info	Type	Planned	Repeatable	Multiple Occurrences	Ongoing Event	Driven	Optional	Team
Inception Phase	1					Phase	✓						
Inception Iteration [1..n]	2					Iteration	✓	✓					
Lifecycle Objectives Milestone	3	2				Milestone	✓						
Elaboration Phase	4	1				Phase	✓						
Elaboration Iteration [1..n]	5	3				Iteration	✓	✓					
Lifecycle Architecture Milestone	6	5				Milestone	✓						
Construction Phase	7	4				Phase	✓						
Construction Iteration [1..n]	8	6				Iteration	✓	✓					
Initial Operational Capability Milestone	9	8				Milestone	✓						
Transition Phase	10	7				Phase	✓						
Transition Iteration [1..n]	11	9				Iteration	✓	✓					
Product Release Milestone	12	11				Milestone	✓						

Figura 4.7: WBS de Fases de Ciclo de vida riesgo-valor

Una **Fase** agrupa elementos en un período significativo del proyecto. Una Fase finaliza con un punto de control de gestión importante, con un Hito o con un conjunto de entregables concluidos. En términos de SPEM una Fase es una Actividad que cumple  $ESREPETIBLE = FALSE$ .

La **Iteración** representa un conjunto de actividades anidadas que se repiten más de una vez. Permite al ingeniero de procesos organizar ciclos repetitivos de trabajo. Para SPEM, una Iteración es una Actividad con  $ESREPETIBLE = TRUE$ .

Un **Hito** representa un evento significativo para el desarrollo de un proyecto, por ejemplo una decisión importante, la conclusión de un entregable, la finalización de una fase. Se trata de un Elemento de desglose de trabajo, de manera que tiene que aparecer en una WBS y puede tener relaciones de precedencia.

Un **Patrón de proceso** (Capability pattern) es un “fragmento de proceso” que puede contener Actividades e Hitos y que es reutilizable como solución a algún tipo de problema o situación habitual. Agrupar elementos de Proceso en Patrones de Capacidad permite a los ingenieros de proceso reusar estos fragmentos y componer Procesos de Entrega a partir de ellos. Si un patrón común, como el plan de una iteración, es necesario en diferentes partes del proceso de entrega, se puede reutilizar generando un Patrón de proceso.

Un **Proceso de despliegue** es un ciclo de vida del proyecto de punta a punta ensamblados a partir de un conjunto de Actividades, Fases, Iteraciones, Patrones de Proceso e Hitos. Por ejemplo, RUP se distribuye con tres procesos de Entrega: clásico (proyectos grandes), pequeño (proyectos pequeños), medium (proyectos medianos).

## Diagramas de proceso

Existen tres diagramas que son relevantes en el contexto de nuestra tesis: el flujo de trabajo, el detalle de actividades y las dependencias de productos de trabajo.

### Diagrama de flujo de trabajo

Se trata de una versión adaptada de los Diagramas de Actividad UML que puede contener un punto de inicio y uno de fin, nodos de decisión, links, actividades, barras de sincronización, fases de descriptores de tareas e hitos. Provee una visión de alto nivel de las actividades y su secuencia. La Figura 4.8 muestra un ejemplo. Las barras de sincronización muestran un posible paralelismo entre actividades. Las flechas en los links muestran orden de ejecución entre los elementos de proceso.

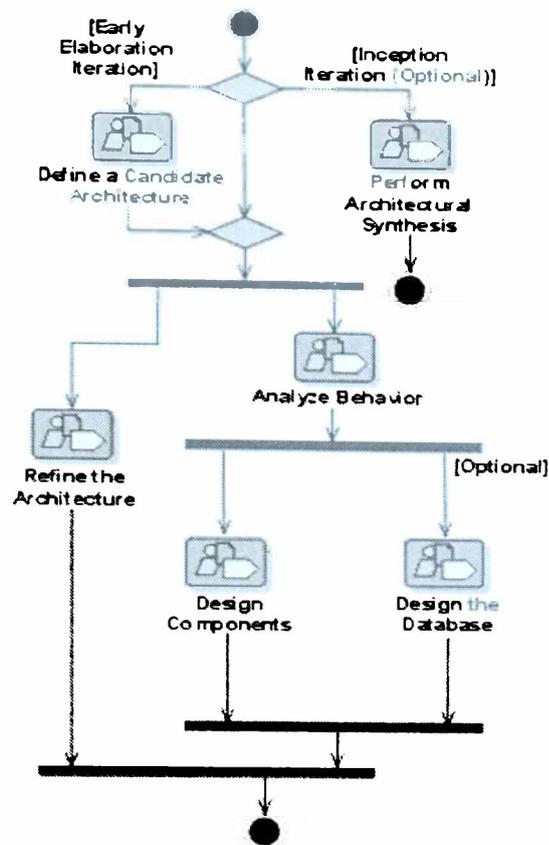


Figura 4.8: Diagrama de Flujo de Trabajo en SPEM 2.0

### Diagrama de detalle de actividades

Brinda una visión general de los descriptores de las tareas dentro de una actividad, los descriptores de los roles asociados con las tareas y los descriptores de productos de trabajo de entrada y salida. Los dos diagramas en la Figura 4.9 muestran para cada rol responsable en la dimensión horizontal la lista de tareas en la actividad y en la vertical el flujo de productos de trabajo. Un descriptor es básicamente un objeto de referencia dentro de un proceso que representa la ocurrencia de un elemento de contenido del método. Tiene sus propias relaciones y documentación que define la diferencia entre la implementación por defecto y esta particular ocurrencia del elemento en el proceso.

### Diagrama de dependencia de Producto de Trabajo

Ilustra las relaciones y dependencias entre varios productos de trabajo. Se utilizan para mostrar trazabilidad de productos de una actividad. La Figura 4.10 muestra un diagrama de este tipo utilizando los iconos de SPEM 2.0 para los elementos de proceso.

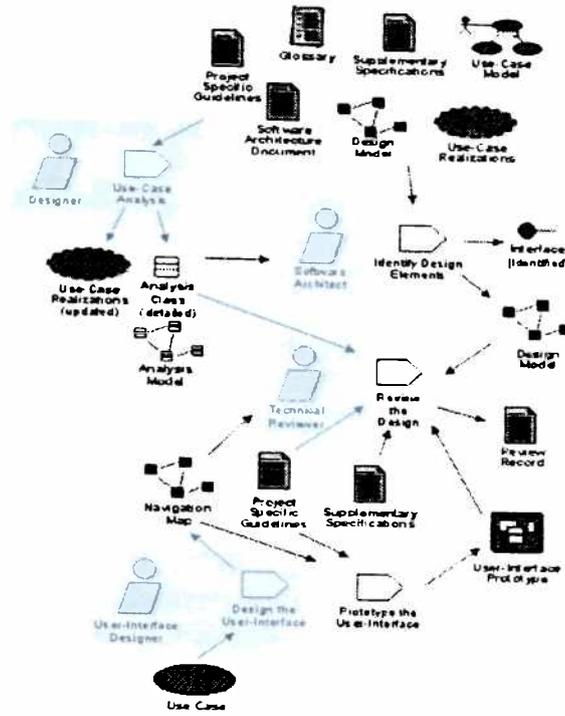


Figura 4.9: Diagrama de detalle de actividad en SPEM 2.0

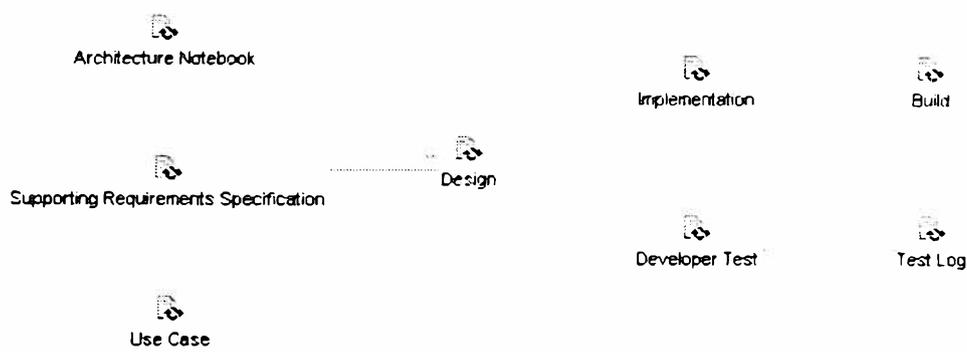


Figura 4.10: Diagrama de Dependencias de Producto de Trabajo en SPEM 2.0

### 4.3. La biblioteca de prácticas para el EPF

La *Biblioteca de Prácticas del Framework de Procesos Eclipse (EPL)* es una biblioteca que conforma al UMF, liberada junto con la versión 1.5.0.1 del Eclipse Process Framework Composer [Eclipse 2008a].

En el contexto del Eclipse Process Framework, una práctica es una aproximación a resolver uno o más problemas que se producen comúnmente. Estas prácticas están destinadas a ser adoptadas, habilitadas y configuradas como “cadenas” de procesos .

Estas prácticas ofrecen las siguientes ventajas:

**Adaptabilidad y escalabilidad** El conjunto de prácticas incluidas soporta un amplio rango de soluciones.

**Adopción incremental** Cada práctica se describe como una capacidad autocontenida que puede ser adoptada por una organización o un proyecto.

**Facilidad de configurar y usar** Crear un método es tan simple como seleccionar las prácticas a adoptar y publicar los resultados. Cada práctica se agrega a sí misma al framework de modo que el contenido pueda ser visto ya sea por práctica o por productos, roles, etc.

**Desarrollo comunitario** El desarrollo de la Biblioteca Open Source de Prácticas se desarrolla por la comunidad del Eclipse Process Framework

Por otro lado, se sostiene que las buenas prácticas incluidas en el framework han surgido del análisis de un gran número de proyectos exitosos [Kroll and MacIsaac 2006].

Finalmente, señalemos que el *Método de Autoría de Métodos (MAM)* es una aproximación a la creación de métodos centrada en arquitectura, orientada a la calidad y basada en prácticas. MAM provee guías para crear métodos que serán incluidos dentro de la Biblioteca de Prácticas EPF [Eclipse 2008b]. Más adelante retomaremos las indicaciones del MAM para generar una de las contribuciones de esta tesis.

#### Los bloques para construir prácticas

A continuación describimos brevemente las definiciones de los Roles, Tareas y Productos de trabajo incluidos en EPF como elementos de Contenidos de Método sobre los que se construyen las Prácticas de la EPL y que se muestran en la Figura 4.11.

##### Roles

La EPL presenta siete roles básicos, necesarios para equipos pequeños y sin distribución geográfica:

**Analista** representa los concerns del cliente y los usuarios finales mediante la obtención de input directo de los involucrados para comprender el problema a resolver y la captura o planteo de prioridades para los requerimientos.

**Arquitecto** es responsable de definir la arquitectura del software, lo que incluye tomar las decisiones técnicas claves que restringirán el diseño en líneas generales.

**Desarrollador** es responsable de desarrollar una parte del sistema, incluyendo el diseño para adecuarla a la arquitectura, posiblemente el prototipado de la interfaz de usuario y la implementación, test de unidad e integración de los componentes que son parte de la solución.

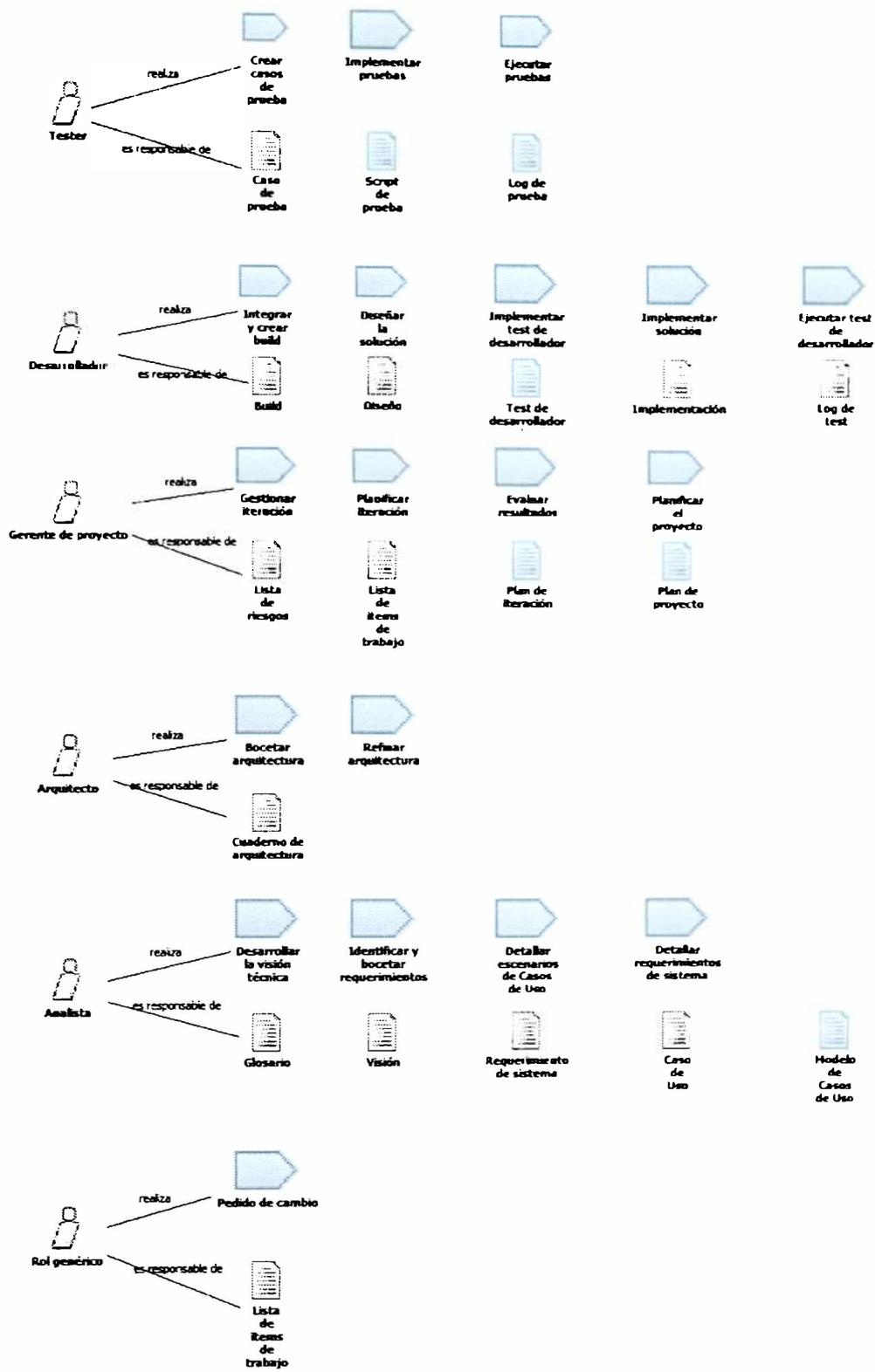


Figura 4.11: Bloques de Prácticas EPL

**Stakeholder** representa los grupos de interés cuyas necesidades pueden ser satisfechas por el proyecto. es un rol que puede ser desempeñado por cualquiera que será, real o potencialmente, afectado por el resultado del proyecto.

**Tester** es responsable de las actividades centrales del esfuerzo de prueba, incluyendo la identificación, definición, implementación y conducción de las pruebas necesarias, tanto como de registrar y analizar los resultados.

**Gerente de proyecto** lidera la planificación del proyecto, coordina las interacciones con los involucrados y mantiene el equipo focalizado en alcanzar los objetivos del proyecto

**Rol genérico** cualquier interesado en el proyecto o integrante del equipo que no está comprendido en los roles anteriores

### Tareas

La EPL se construye sobre 18 tareas que los diferentes Roles pueden realizar como actores principales (y por lo tanto son responsables de que se ejecuten) o secundarios (proveyendo información o soporte para ejecutar la tarea). Las tareas están asociadas con conceptos, orientaciones y listas de control.

**Bocetar la arquitectura** Consiste en delinear las primeras decisiones de arquitectura que guiarán el desarrollo y las pruebas. Se comunica a todo el equipo para que se tenga suficiente información sobre el tipo de aproximación técnica que se adopta.

**Evaluar resultados** determinar el éxito o fracaso de una iteración y aplicar las lecciones aprendidas para modificar el proyecto o mejorar el proceso.

**Crear Casos de prueba** Desarrollar casos y datos de prueba para cada requerimiento a testear.

**Definir la visión** del sistema, describiendo los problemas y características sobre la base de los requerimientos de stakeholders.

**Detallar escenarios de Casos de Uso** Describir los flujos de los casos de uso en suficiente detalle para poder validar la comprensión de los requerimientos, asegurar la satisfacción de expectativas de stakeholders y permitir que comience el desarrollo del software.

**Encontrar y bocetar requerimientos** identificar y capturar los requerimientos funcionales y no funcionales del sistema en un nivel de abstracción alto que permita obtener una aproximación inicial del alcance del trabajo por delante

**Implementar tests de desarrollador** implementar una o más pruebas que permitan la validación de una implementación del desarrollador

**Implementar scripts de tests** implementar los scripts para las pruebas que permitan validar una solución

**Implementar la solución** escribir el código fuente para proveer nueva funcionalidad o corregir defectos

**Integrar y crear un build** integrar todos los cambios introducidos por los desarrolladores en la base de código

**Gestionar una iteración** monitorear el estado del proyecto, identificar cualquier bloqueo u oportunidad de avance. Identificar y manejar las excepciones, problemas y riesgos. Comunicar el estado del proyecto.

**Planificar una iteración** planificar el alcance y las responsabilidades para una iteración, identificando el próximo incremento de las capacidades del sistema.

**Planificar el proyecto** elaborar un plan del proyecto que resuma las actividades principales y provea un acuerdo entre todos los involucrados sobre las metas y forma de lograrlas. Obtener el acuerdo de stakeholders para iniciar el proyecto y el compromiso del equipo para avanzar.

**Refinar la arquitectura** a partir de la arquitectura bocetada, esta tarea implica tomar decisiones para concretarla en los detalles y proveer el soporte al desarrollo del sistema

**Requerir cambios** capturar y registrar los pedidos de cambio

**Ejecutar tests de desarrollador** ejecutar pruebas contra la implementación individual del desarrollador

**Ejecutar tests** ejecutar los scripts de pruebas adecuados, analizar los resultados y comunicarlos al resto del equipo

**Diseñar la solución** identificar los necesarios para resolver alguna funcionalidad en particular y disponer las interacciones, comportamientos, relaciones y datos necesarios para ello

**Detallar requerimientos de sistema** describir los requerimientos suplementarios de alcance de todo el sistema en suficiente detalle para permitir su validación y el inicio de su desarrollo

### Productos de trabajo

Mediante la realización de las diferentes tareas, los roles producen, modifican o utilizan 17 artefactos incluidos en EPL. La mayoría de los artefactos se asocia con plantillas y listas de control. La plantilla o *template* provee orientación adicional para completar el artefacto y la lista de control ayuda a verificar la calidad del resultado.

**Cuaderno de arquitectura** Describe el rationale, los supuestos, explicaciones e implicancias de las decisiones que se toman para constituir la arquitectura

**Build** versión operativa del sistema o una parte que demuestra un subconjunto de las capacidades a proveer por el producto final

**Diseño** describe la realización de una funcionalidad requerida y sirve como abstracción del código fuente

**Test de desarrollador** valida un aspecto específico de un elemento de implementación

**Glosario** define los términos importantes utilizados en el proyecto

**Implementación** archivos de código y datos, archivos de soporte como ayuda online, que representan las partes en crudo del sistema en construcción

**Plan de iteración** plan de bajo nivel que describe objetivos, asignaciones y criterios de evaluación para una iteración

**Plan de proyecto** plan de alto nivel que incluye la información para gestionar el proyecto, describiendo las iteraciones y sus metas

**Lista de riesgos** lista ordenada de riesgos abiertos y conocidos del proyecto

**Especificación de requerimiento de sistema** captura los atributos de calidad y restricciones que tienen alcance en todo el sistema

**Caso de prueba** especificación de un conjunto de inputs, condiciones de ejecución y resultados esperados de una prueba para evaluar un aspecto particular de un escenario

**Log de prueba** reúne la salida en bruto capturada durante una corrida de pruebas

**Script de prueba** contiene las instrucciones paso a paso para ejecutar una prueba, puede ser un documento para hacer pruebas manuales o código para su ejecución automática

**Modelo de Casos de Uso** captura el modelo de las funcionalidades y entorno requerido del sistema y sirve como contrato entre el cliente y el equipo

**Caso de Uso** captura el comportamiento del sistema que permite mostrar un resultado observable y de valor para los que interactúan con el sistema

**Visión** define la vista que los stakeholders tienen de la solución técnica a desarrollar, especificada en términos de necesidades y características de los stakeholders

**Lista de ítems de trabajo** contiene la lista de todo el trabajo programado para realizarse dentro del proyecto. Cada ítem de trabajo puede contener referencias a información relevante necesaria para realizarlo

## Las prácticas de la EPL

Ya hemos mencionado que en el marco de la EPL una práctica define una manera de resolver uno o más problemas que se producen habitualmente. La EPL organiza doce buenas prácticas de ingeniería de software en dos grupos:

- **Prácticas del Kernel Ágil**, que incluyen todas las buenas prácticas de métodos ágiles:
  - Desarrollo iterativo
  - Planificación en dos niveles
  - Equipo integrado
  - Testing concurrente
  - Integración continua
- **Prácticas de EPF para escalar la agilidad**, que permiten extender el corazón de agilidad a procesos de mayor envergadura:
  - Arquitectura evolutiva
  - Diseño evolutivo
  - Ciclo de vida riesgo-valor
  - Visión compartida
  - Gestión grupal de cambios
  - Desarrollo conducido por testing
  - Desarrollo conducido por casos de uso

## Práctica: Desarrollo iterativo

Esta práctica se basa en la idea de que el software se construye en una serie de incrementos a lo largo de una secuencia de iteraciones. Cada incremento agrega un subconjunto de la funcionalidad final del sistema. De esta forma, el sistema crece y deviene más completo en el transcurrir de las iteraciones.

Se incluyen tres tareas, que producen otros tantos artefactos:

**Planificar la iteración** consiste en priorizar los items de trabajo, definir los objetivos de la iteración, comprometer el trabajo para los items priorizados, identificar y actualizar la lista de riesgos, definir los criterios de evaluación y refinar la definición y alcance del proyecto. El resultado principal de esta tarea es el Plan de Iteración, pero también resultan actualizados la Lista de items y la Lista de riesgos.

**Gestionar la iteración** Esta tarea es responsabilidad central del Gerente de Proyecto, que debe evaluar permanentemente el estado del trabajo, identificar elementos de bloqueo y oportunidades, manejar las excepciones, los problemas y los riesgos y comunicar cuando corresponda el estado del proyecto. El output de esta tarea es la versión actualizada de los mismos artefactos de la tarea anterior.

**Evaluar resultados** El propósito de esta tarea es demostrar el valor entregado por el incremento de solución desarrollado en la iteración y aplicar las lecciones aprendidas para modificar o mejorar el proyecto. Si bien el principal responsable es también el Gerente del Proyecto, todos los actores del equipo contribuyen a esta tarea que se efectúa a modo de revisión al finalizar cada iteración. El resultado es la versión actualizada del Plan de iteración (especialmente en los aspectos a ser tenidos en cuenta en la próxima iteración) y la Lista de Items.

Es central incluir en esta descripción tres conceptos claves que se introducen en la EPL respecto de otros métodos basados en el Proceso Unificado: el ciclo de vida de cada iteración, los micro-incrementos y las retrospectivas

**El ciclo de vida de las iteraciones** En EPL cada iteración tiene su propio ciclo de vida, comenzando con una planificación y finalizando con un incremento estable del sistema, una Revisión de Iteración (¿logramos lo que nos propusimos?) y una Retrospectiva (¿hay mejor forma de hacerlo?).

La planificación de la iteración, la estimación y el monitoreo de avance está centrado en los items de trabajo que constituyen la base de los micro-incrementos.

Una iteración comienza con una reunión de planificación de un par de horas. Los dos primeros días típicamente se focalizan en mejorar la planificación y la arquitectura base para entender las dependencias y el ordenamiento lógico de los ítems de trabajo. La mayor parte del tiempo de una iteración se enfoca en ejecutar los micro-incrementos. Cada uno de estos debería entregar un código testeado para integrar y otros artefactos validados. EPL incorpora un poco más de disciplina al establecer que los builds stables deben producirse al final de cada semana. La última semana de la iteración tiene un énfasis más fuerte en pulir y corregir bugs. La iteración finaliza con una evaluación (que incluye a los stakeholders) de lo que se construyó y una retrospectiva que permita entender cómo mejorar el proceso para la próxima iteración.

**Los micro-incrementos** representa el resultado de unas pocas horas hasta unos pocos días de trabajo realizado por una o pocas personas colaborando para alcanzar una meta. Algunos ejemplos de micro-incrementos serían: identificar un involucrado (un paso dentro de una tarea), determinar la forma de abordaje técnico para la persistencia (una tarea

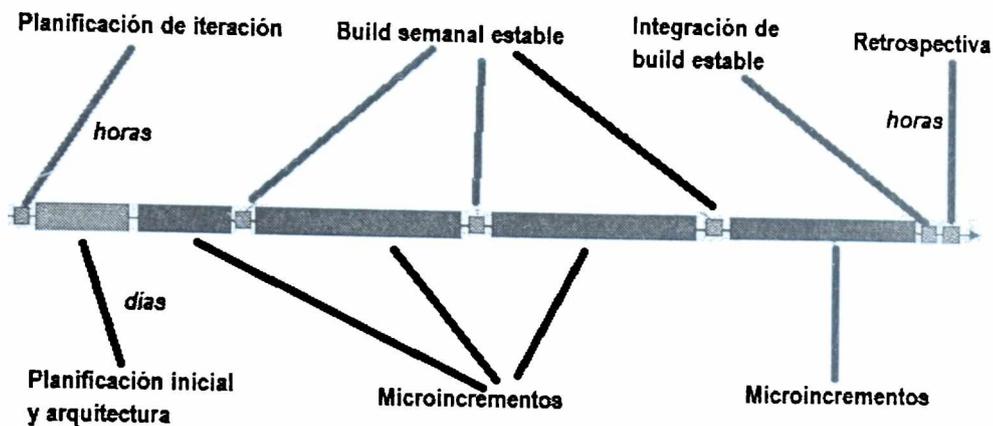


Figura 4.12: Ciclo de vida de la iteración en OpenUP

con un foco específico), desarrollar un incremento de solución para el Flujo Principal del Caso de Uso 1 (una tarea con un foco específico)

El desarrollo evoluciona en micro-incrementos a través de la ejecución simultánea de varios ítems de trabajo. Al compartir el avance de todos los mi mediante las reuniones diarias y las herramientas de colaboración, se alcanza la transparencia y comprensión del trabajo de otros requerido para un efectivo trabajo de equipo. Al mismo tiempo se demuestra el progreso continuo con la evolución de un mi a la vez.

**Las retrospectivas** realizadas no sólo al finalizar el proyecto, sino tras cada iteración y release, proveen una forma efectiva de monitorear la salud del proyecto y ayudar al equipo a aprender de las lecciones.

Preguntas del estilo de “Qué funcionó bien para nosotros en la iteración anterior?”, “Qué cosas no funcionaron bien?”, “Qué deberíamos hacer de manera diferente o qué mejora deberíamos intentar en la próxima iteración?”, junto con las acciones propuestas en consecuencia ayudan a priorizar mejoras y mantener bajo control el pulso del proyecto.

### Práctica: Planificación del Proyecto en dos niveles

Incorpora el concepto de una planificación de alto nivel o macro para el alcance completo del proyecto y una de bajo nivel o micro para los próximos incrementos e iteraciones.

La práctica de Planificar en dos niveles enfatiza la necesidad de planificar en detalle sólo la próxima iteración y mantener un nivel más alto de abstracción para el conjunto del proyecto. Por otra parte, alienta a los gerentes a sentirse cómodos con la re-planificación como una forma necesaria de asegurar precisión en las fechas y costos de entrega, considerando los eventos que impactan en el proyecto.

En consecuencia, la práctica incluye una sola tarea para el nivel macro: Planificar el proyecto, que consiste en consensuar un plan sobre la forma en que el proyecto alcanzará sus metas. Esta tarea es la oportunidad para que el equipo acuerde sobre algunas variables importantes del proyecto: alcance, objetivos, calendario inicial, entregables. Focalizando en una forma auto-organizativa característica de procesos ágiles, permite al equipo definir criterios de éxito y formas de trabajo a ser empleadas. Si bien la colaboración y el consenso constituyen metas de esta tarea, el Gerente del Proyecto, como responsable de la misma, debe asegurar el compromiso de todos con el plan. El producto de la tarea (Plan del proyecto) quedará sujeto a la actualización de acuerdo con el feedback y cambios en el contexto durante el resto del proyecto.

Para la EPL una planificación incluye establecer un equipo cohesivo, estimar el tamaño del proyecto, evaluar los riesgos, prever la velocidad y duración del proyecto, esbozar el ciclo de vida del proyecto, establecer costos y articular el valor, planificar el despliegue.

Para el nivel micro, la tarea Planificar la iteración quedará vinculada con la práctica Desarrollo iterativo, en las planificaciones de cada iteración

### Práctica: Equipo integrado

Describe cómo un equipo de desarrollo se autoorganiza para trabajar efectivamente. Esta práctica se funda sobre el hecho de que el factor más importante para la productividad son los recursos humanos y el modo en que interactúan. Propone estrategias para aumentar la productividad grupal mediante coordinación tanto en el nivel de la estructura del equipo como dentro del equipo mismo.

La práctica caracteriza al equipo integrado como autoorganizado, transversal, fluido y altamente colaborativo.

Esta práctica no incluye productos de trabajo específicos (obviamente, tampoco tareas propias), sino que propone tres guías o directrices para la acción:

- **Encuentros diarios** ala Scrum [Schwaber 2004]. Breves reuniones de todo el equipo para mantener el pulso del proyecto.
- **Mantener un ritmo sostenible**, evolución de la “semana de 40 horas” de Extreme Programming. Con esta estrategia se busca evitar el burnout del equipo que se produce cuando el ritmo al que se aspira esta fuera de toda posibilidad de sostenimiento.
- **Autoorganizar la asignacion de tareas**. también tomado de Scrum, con el propósito de que sea el propio equipo (y cada uno de sus integrantes) los que decidan en cada momento las asignaciones que se toman, promovienod de esta manera un mayor compromiso con cada tarea. Queda espacio, sin embargo, para el liderazgo del Gerente de Proyecto en la resolución de conflictos y la vigilancia de que las tareas son abordadas dentro de las habilidades disponibles.

### Práctica: Testing concurrente

Describe la incorporación del testing en un proceso ágil de desarrollo. El término concurrente pretende indicar que el testing debe abordarse a lo largo de toda la iteración de manera paralela al desarrollo, evitando que se separen las pruebas en una actividad al final de la iteración o la release. Obviamente. esto requiere una gran integración de los diferentes roles del equipo, en particular desarrolladores y testers.

La práctica comprende tres tareas: Crear casos de prueba, Implementar pruebas y Ejecutar las pruebas. De ellas surgirán tres artefactos: el Caso de Prueba, el Log de la prueba y el Script de la prueba.

**Crear casos de prueba** Los casos de prueba se originan en los requerimientos del sistema y son desarrollados en paralelo con su especificación. Si bien el rol responsable de esta tarea es el Tester, también intervienen los Analistas, Desarrolladores y Stakeholders para asegurar consenso sobre las condiciones de satisfacción de las pruebas ya que estas funcionarán en algún momento como los criterios de aceptación.

**Implementar las pruebas** Consiste en implementar los Scripts de prueba, que contienen las instrucciones paso a paso para la ejecución de la prueba. En esta tarea, además de diseñar e implementar el ejecutable del Script, se definen los datos necesarios y se organiza los Scripts en suites o grupos relacionados.

**Ejecutar las pruebas** Esta tarea ejecuta los Scripts, analiza los resultados y los comunica al equipo.

La definición de la práctica se completa con cuatro orientaciones: Mantener suites de pruebas automatizadas, Programar pruebas automatizadas, Ideas de prueba y Suites de Pruebas.

### Práctica: Integración continua

Esta práctica plantea que cada miembro del equipo integra su trabajo en forma frecuente (al menos diariamente). La idea que sustenta esta práctica es que el esfuerzo requerido para integrar un sistema crece exponencialmente con el tiempo. Integrando el sistema frecuentemente, los aspectos problemáticos de la integración son detectados en forma precoz, cuando resultan más fáciles de corregir y por lo tanto el esfuerzo global de integración se reduce. Lo que a su vez se traduce en productos de mayor calidad y cronogramas de entrega más previsibles.

La tarea es **Integrar y crear un build**, que consiste en integrar los cambios hechos por un desarrollador en el código base del sistema (normalmente en el workspace del desarrollador), crear el build, ejecutar pruebas sobre los elementos integrados y liberar los cambios probados (cuya forma concreta depende del sistema de gestión de la configuración utilizado).

El artefacto involucrado es el **Build** y la práctica se completa con dos orientaciones sobre la manera de efectuar la **Integración continua** y de **Promover (liberar) los cambios realizados**.

### Práctica: Ciclo de vida riesgo-valor

Provee el Ciclo de Vida del Proceso Unificado. Este ciclo de vida identifica cuatro fases: Inicio, Elaboración, Construcción y Transición (ver Figura 4.3), cada una de las cuales intenta balancear el valor provisto contra la mitigación de riesgo apropiada a la fase.

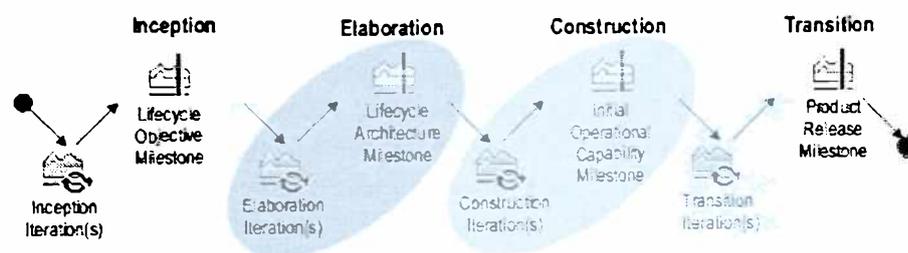


Figura 4.13: Ciclo de vida Riesgo-Valor

Disponer de un ciclo de vida para el proyecto provee a todos los involucrados de puntos de sincronización a lo largo de todo el proyecto, facilitando la supervisión y decisiones de avanzar-detener en momentos oportunos. Cada fase tendrá un conjunto definido de metas, una o más iteraciones, tareas y productos específicos para satisfacer las necesidades puntuales del proyecto en ese momento. El ciclo de vida en fases tiene por objetivo focalizar en dos drivers básicos para todos impulsores de un proyecto: reducir el riesgo y crear valor. Con la presencia de hitos de finalización de fase, traducidos en preguntas sencillas, el equipo de trabajo se mantiene enfocado en minimizar los riesgos y aumentar el valor entregado, tal como sugiere el diagrama de la figura 4.14

Bastará con plantearse en el hito final de cada fase:

- Inicio: ¿Estamos de acuerdo en el alcance y objetivos del proyecto? ¿Creemos que el proyecto debería continuar?
- Elaboración: ¿Estamos de acuerdo en la arquitectura a utilizar? ¿El valor entregado hasta ahora y los riesgos restantes son aceptables?

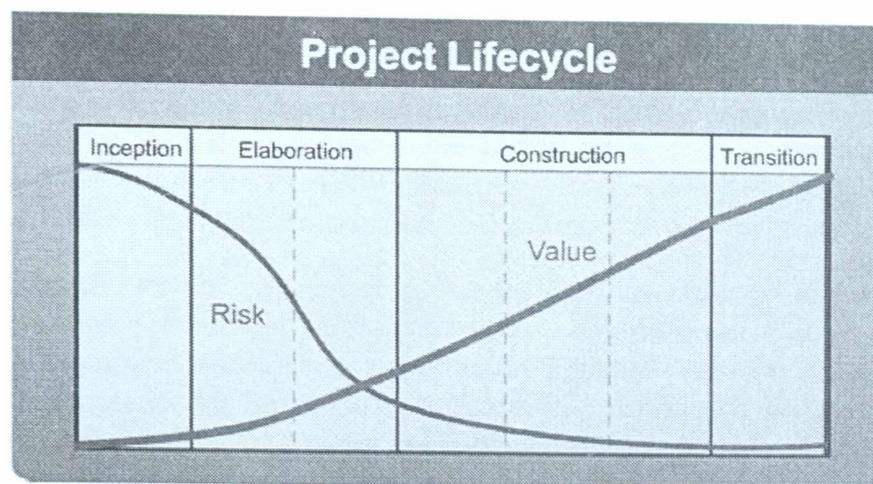


Figura 4.14: Curvas de reducción de riesgo y entrega de valor

- **Construcción:** ¿Tenemos una aplicación suficientemente avanzada como para enfocarnos en el ajuste y asegurarnos de un despliegue exitoso?
- **Transición:** ¿La aplicación está lista para ser liberada?

Si la respuesta es afirmativa, el proyecto estará en condiciones de pasar a la siguiente fase. De lo contrario, se agrega una nueva iteración a la fase o el cliente puede decidir cancelar el proyecto.

### Práctica: Gestión grupal de Cambios

Esta práctica intenta capturar los pedidos de cambio que pasan a ser gestionados como items de trabajo. Se describe el mecanismo de gestión de cambios más simple: cualquiera del equipo puede pedir un cambio al sistema (o a los artefactos de soporte) en cualquier momento, ya sea para corregir un defecto como para agregar una mejora. El pedido es capturado en el backlog de pendientes. En cada planificación (recordemos la práctica de Planificar en dos niveles, el proyecto y cada iteración), el equipo revisa el backlog, prioriza el trabajo y decide los cambios que serán implementados a continuación.

La práctica incluye sólo la tarea Pedir un cambio, que deriva en una actualización de la Lista de items de trabajo. La orientación "Presentar pedidos de cambio" incluye guías concretas sobre el tipo de información que debe capturarse en cada pedido con el objetivo de facilitar su priorización y determinar el trabajo involucrado en su satisfacción.

### Práctica: Arquitectura evolutiva

Describe la forma de construir y mejorar en incrementos la arquitectura del software, al mismo tiempo que se descubren y enfrentan las cuestiones arquitectónicas durante el desarrollo. El objetivo es reducir el riesgo técnico sin un esfuerzo muy importante de arquitectura al comienzo del trabajo.

Se plantean tres principios claves:

- Realizar arquitectura "just in time" para el resto del proyecto, difiriendo para otras fases los aspectos que no son indispensables en la actual o la próxima.
- Documentar las decisiones arquitectónicas, utilizando el único Producto de Trabajo de la práctica: el Cuaderno de Arquitectura.
- Implementar y probar funcionalidad clave como una forma de enfrentar aspectos arquitectónicos, utilizando los prototipos rápidos para validar las asunciones que se hacen desde la arquitectura.

La práctica incluye dos tareas:

**Bocetar la arquitectura** , que busca capturar las ideas centrales, metas, requerimientos y restricciones de la arquitectura del sistema, sus principales particiones, los mecanismos de arquitectura a emplear, las interfaces con otros sistemas, etc. Todas las decisiones son documentadas en el Cuaderno de arquitectura.

**Refinar la arquitectura** , que construye sobre la arquitectura esbozada y transformándola en decisiones e implementaciones concretas que dan soporte al desarrollo. Toma en cuenta no sólo los antecedentes arquitectónicos, sino cualquier producto desarrollado hasta el momento, de forma de evolucionar capturando las decisiones y cambios hechos en el desarrollo. Esta parte es fundamental, ya que la implementación real es la única prueba de que la arquitectura es viable. Todas las decisiones que se tomen en esta tarea sobre actualizaciones de metas, requerimientos, restricciones, mecanismos de arquitectura, etc. son documentadas y comunicadas al resto del equipo.

### **Práctica: Diseño evolutivo**

Describe un abordaje al diseño que asume que evolucionará a través del tiempo, minimizando la documentación, pero proveyendo guía para tomar las decisiones de diseño y comunicarlas. En la esencia de esta práctica, cada pasada de diseño agrega, refina y refactoriza una solución mediante una mejor comprensión de los requerimientos, la identificación de los elementos de diseño y su colaboración, el perfeccionamiento de las opciones de diseño, su comunicación, el feedback sobre la arquitectura, etc.

La práctica incluye una tarea, un producto de trabajo y tres orientaciones para diseñar:

**Diseñar la solución** , que consiste en identificar los elementos y sus interacciones, comportamiento y relaciones necesarios para implementar alguna funcionalidad. Se completa con la exposición visual del diseño para comunicarlo.

**Diseño** , que describe la realización de la funcionalidad requerida y sirve como abstracción para el código fuente.

**Analizar el diseño, Evolucionar el diseño y Refactorizar el diseño** , guías para llevar a cabo los pasos de **Diseñar la solución** en diferentes contextos de proyecto.

### **Práctica: Visión compartida**

Esta práctica da soporte a la definición y comunicación de la visión global del proyecto. Se funda en la idea que establecer y mantener una visión compartida del problema a ser resuelto y de las propiedades de alto nivel de la solución propuesta mantiene alineadas las expectativas y reduce el riesgo asociado con la aceptación del usuario.

Para lograrlo, se centra en el Producto de trabajo Visión, que no sólo debe identificar claramente los diferentes stakeholders, sino proveer una comprensión unificada del problema, capturar sus necesidades, las restricciones del proyecto y dar background y contexto para los requerimientos que luego serán detallados. Tanto como establecer una Visión compartida al comienzo, lo es mantenerla durante todo el proyecto. El producto Visión establece una perspectiva de largo plazo y que provee información para análisis costo-beneficio y priorizaciones de trabajo.

Los componentes de la práctica son:

**Desarrollar la visión técnica** que documenta en el producto Visión la comprensión del problema a resolver y las características de la solución que se propone. El responsable es el Analista, pero deben participar activamente también los Stakeholders identificados, el Gerente de Proyecto y el Arquitecto.

**Visión** es el producto resultado de tarea anterior, sirve para comunicar los “qué” y “por qué” del proyecto y es el documento contra el que deben ser validadas las decisiones de alto nivel tomadas en el desarrollo.

**Glosario** es el artefacto que define los términos importantes empleados en el proyecto y que contribuye a la consistencia del trabajo, unificando el vocabulario de todos los involucrados. item[**Orientación: Técnicas de obtención de requerimientos y Revisiones efectivas de requerimientos**] son dos orientaciones que se incluyen para mejorar la comprensión de las necesidades de los stakeholders.

### Práctica: Desarrollo conducido por pruebas (TDD)

Describe una aproximación al desarrollo en el cual los casos de prueba son definidos primero y el código se desarrolla para superar esas pruebas. La idea es que mediante ciclos muy cortos, se incorpore al sistema sólo el código necesario para superar las pruebas, que a su tiempo han sido definidas con base en los requerimientos.

El TDD ha sido descrito y analizado en varios textos, por ejemplo [Beck 2002], [Astels 2003]. Algunas de las ventajas de esta práctica consisten en que es más fácil determinar en qué consiste un comportamiento correcto ya que es necesario definirlo para poder hacer la prueba, los errores se descubren antes, es imprescindible una estrecha colaboración entre los desarrolladores, diseñadores y arquitectos, es más fácil saber cuándo el código está terminado (cuando la prueba pasa exitosamente) y se produce una “natural” separación de concerns entre quienes desarrollan código nuevo y quienes mejora el código existente.

En EPL se intenta hacer compatible la utilización de TDD con las otras prácticas (arquitectura y diseño evolutivo, por ejemplo) mediante las orientaciones incluidas en la **Guía: Usar TDD en contexto**, que presenta, por ejemplo, la combinación de TDD con diseño evolutivo, según muestra la figura 4.15

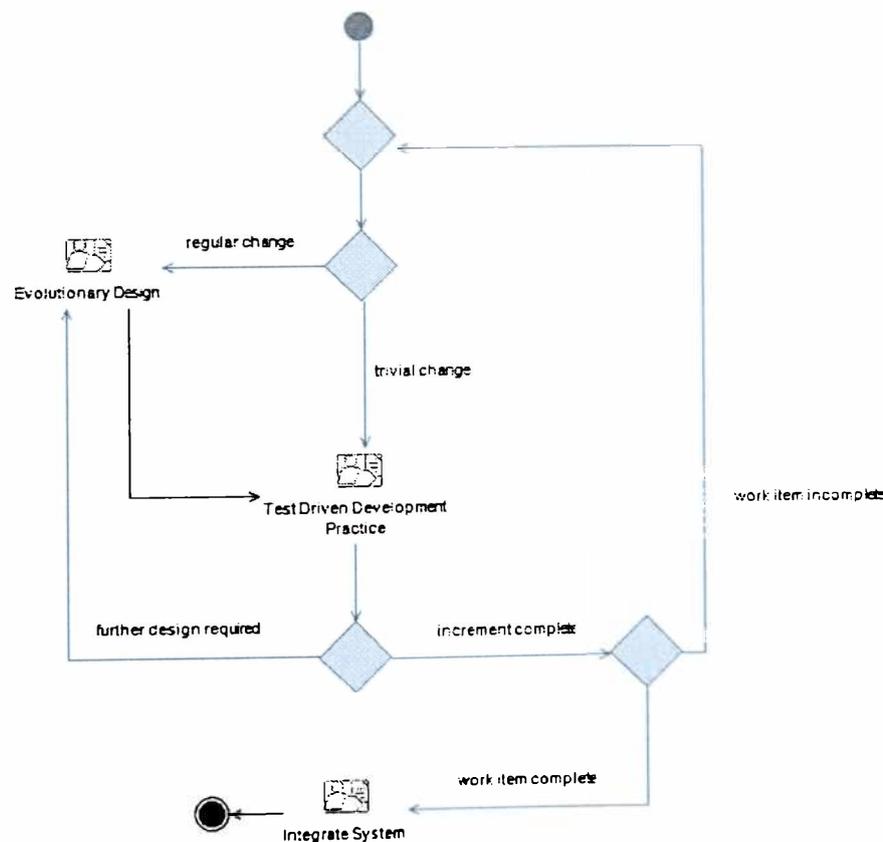


Figura 4.15: TDD en contexto

Además, la práctica incluye:

**Implementar test de desarrollador** que permita la validación del código a implementar por el desarrollador (normalmente en unidades menores a la especificación funcional)

**Implementar solución** mediante código que agregue funcionalidad al sistema o corrija errores.

**Tarea: Ejecutar test de desarrollador** contra el código propio, para validar las implementaciones realizadas en la iteración.

#### **Test de desarrollador**

**Implementación** código, archivos de datos u otros archivos que representan las partes con las que se elabora el Build del sistema.

#### **Práctica: Desarrollo conducido por Casos de Uso**

Describe cómo capturar requerimientos con una combinación de casos de uso y requerimientos generales y conducir el desarrollo y las pruebas a partir de esos casos de uso.

Al igual que la anterior, esta práctica ha sido extensamente documentada en numerosas fuentes, por ejemplo [Jacobson et al. 1992] y [Cockburn 2000]. Sólo reiteramos aquí que los tres conceptos claves de esta práctica son el Actor (persona o sistema que interactúa con el sistema en desarrollo), Caso de uso (la interacción entre el sistema y uno o más actores y que produce un resultado observable y de valor para los actores involucrados) y Modelo de Caso de uso (el conjunto de Casos de Uso que representa las funcionalidades requeridas del sistema).

EPL organiza esta práctica con tres tareas, tres artefactos y cuatro orientaciones:

**Identificar y esbozar requerimientos** capturando los requisitos funcionales y no funcionales que permitan estimar el alcance del trabajo.

**Detallar escenarios de Casos de Uso** con el suficiente detalle para permitir su validación con los stakeholders.

**Detallar requerimientos comunes a todo el sistema** entendiendo por tales a los que no pueden ser restringidos a un caso de uso en particular.

**Caso de uso, Artefacto: Modelo de Casos de Uso** ya descriptos.

**Requerimiento común a todo el sistema** captura atributos de calidad y restricciones con alcance en todo el sistema.

**Detallar escenarios de casos de uso** Identificar y esbozar actores y casos de uso, Desarrollar la especificación de requerimientos comunes al sistema, Realización de caso de uso

## **4.4. OpenUP**

OpenUP es una configuración de procesos elaborada a partir del EPF y la EPL, como la versión open source del Proceso Unificado. OpenUP adopta una filosofía pragmática y ágil que focaliza en la naturaleza colaborativa del desarrollo de software. Es un proceso agnóstico respecto de las herramientas y de bajo nivel de ceremonial. Utilizando el esquema de Cockburn [Cockburn 2002], podemos ubicarlo en el cuadrante inferior izquierdo, tal

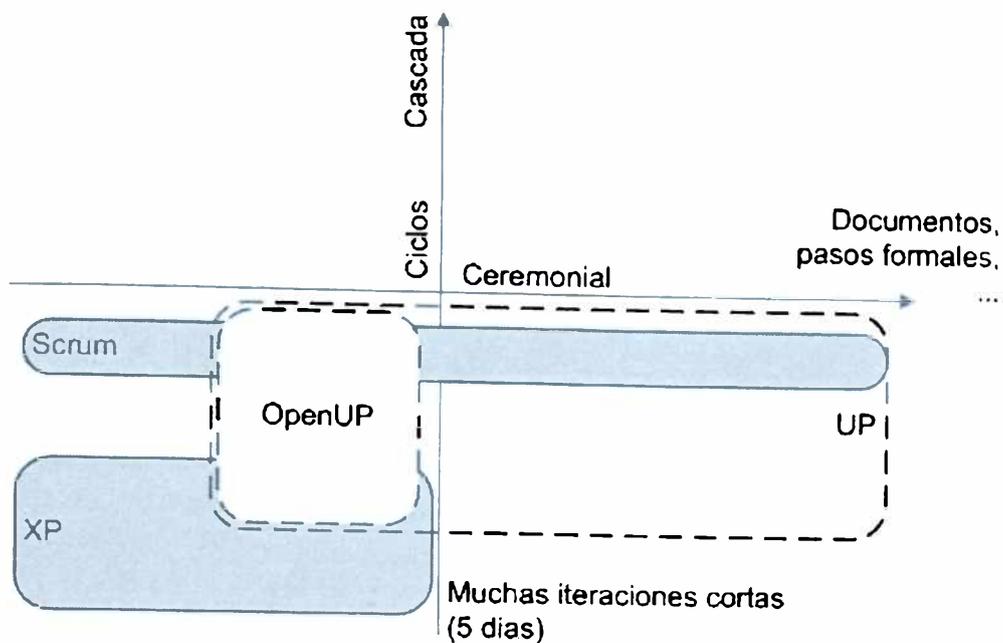


Figura 4.16: Ubicación de OpenUP

como muestra la Figura 4.16, con un bajo nivel de ceremonial y un rango de medio a alto en el número de ciclos o iteraciones durante el proyecto.

La versión básica de OpenUP se presenta como un proceso mínimo, completo y extensible, orientado fundamentalmente a trabajos en proyectos de magnitud pequeña y mediana por equipos concentrados geográficamente. Es **mínimo** porque contiene sólo los roles, tareas y orientaciones considerados imprescindibles para la realización del proyecto; **completo**, porque cubre las disciplinas esenciales del ciclo de vida del desarrollo de software; **extensible**, porque sirve como una base que puede ser adaptada o extendida según las necesidades.

Son cuatro los principios básicos sobre los que se sustenta la filosofía de OpenUP [Kroll and MacIsaac 2006]:

**Colaborar** Este es el principio central y abarcador que tiene relación principalmente con el comportamiento humano. Este principio es el que motivó la mayor parte de las orientaciones incluidas en OpenUP, principalmente en el Área de la Gestión del Proyecto, tales como Estimación ágil, Priorización de Items de trabajo, Asignación de trabajos auto organizada. También influyó al elegir los roles que deberían participar en las tareas y qué productos serían incluidos.

**Balancear** Este principio tiene que ver con maximizar el valor entregado por el proyecto dentro del tiempo y presupuesto disponible. Intenta capturar algunas prácticas de gestión de requerimientos que son claves para entender y priorizar las necesidades y requerimientos para su implementación.

**Focalizar** Este principio promueve una comprensión común de la arquitectura de la solución y asegura que todos entienden la "big picture" para saber de qué forma su trabajo encaja con el resto y provee input para la optimización.

**Evolucionar** La meta es entregar valor y minimizar el riesgo tan pronto como sea posible produciendo software funcional en cada iteración que pueda ser demostrado a los interesados para obtener su feedback. Se basa en el reconocimiento de que uno no puede saber todo desde el comienzo y, aunque ello ocurra, las cosas pueden cambiar. Las iteraciones acomodadas en el contexto de las fases aseguran que uno no itera para siempre.

Aprovechando las características de la EPL, OpenUP intenta obtener el mejor compromiso entre el enfoque ordenado por disciplinas del Proceso Unificado y las ventajas propuestas por las mejores prácticas del Movimiento Ágil. Como puede verse en el Cuadro 4.1, que presenta un mapeo entre los Principios básicos de OpenUP y el Manifiesto Ágil, los valores centrales del Manifiesto encuentran correspondencia en los principios de OpenUp.

Principios clave de OpenUP	Valores del Manifiesto Ágil
Colaborar para alinear intereses y compartir entendimiento	Individuos e interacciones sobre los procesos y las herramientas
Evolucionar para obtener continuamente feedback y mejorar	Responder al cambio antes que seguir los planes
Balancear las prioridades en competencia para maximizar el valor para los involucrados	Colaboración con el cliente sobre la negociación del contrato
Focalizar en la arquitectura temprano para minimizar los riesgos y organizar el desarrollo	Software funcionando antes que una documentación extensa

Tabla 4.1: Mapeo entre principios de OpenUP y valores del Manifiesto Ágil [Kent Beck 2001]

#### 4.4.1. Configuración OpenUP/Basic

La configuración OpenUP/Basic[Eclipse 2008c] organiza el contenido y los procesos del método a partir de la Práctica Ciclo de Vida riesgo-valor, con las cuatro fases corresponden al núcleo del Proceso Unificado.

Antes de describir los principales patrones de entrega de cada fase, digamos que en OpenUP/Basic, se emplean todas las prácticas definidas en la EPL, pero están organizadas en dos categorías ad hoc: Gestión y Técnicas, que no mapean directamente a mencionadas para la EPL (Kernel ágil y Escalabilidad):

- Prácticas de Gestión
  - Desarrollo iterativo
  - Ciclo de vida riesgo-valor
  - Planificación en dos niveles
  - Equipo integrado
  - Gestión grupal del cambio
- Prácticas Técnicas
  - Testing concurrente
  - Integración continua
  - Arquitectura evolutiva
  - Diseño evolutivo
  - Visión compartida
  - Desarrollo conducido por Testing
  - Desarrollo conducido por Casos de Uso

### 4.4.2. Fase de Inicio

El propósito es entender el alcance del problema y la factibilidad de una solución, mediante el logro de objetivos tales como: entender lo que hay que construir, identificar la funcionalidad clave del sistema, determinar al menos una solución posible, entender el costo, tiempos y riesgos asociados con el proyecto. La figura 4.17 muestra el flujo de trabajo asociado a una iteración tipo en esta fase.

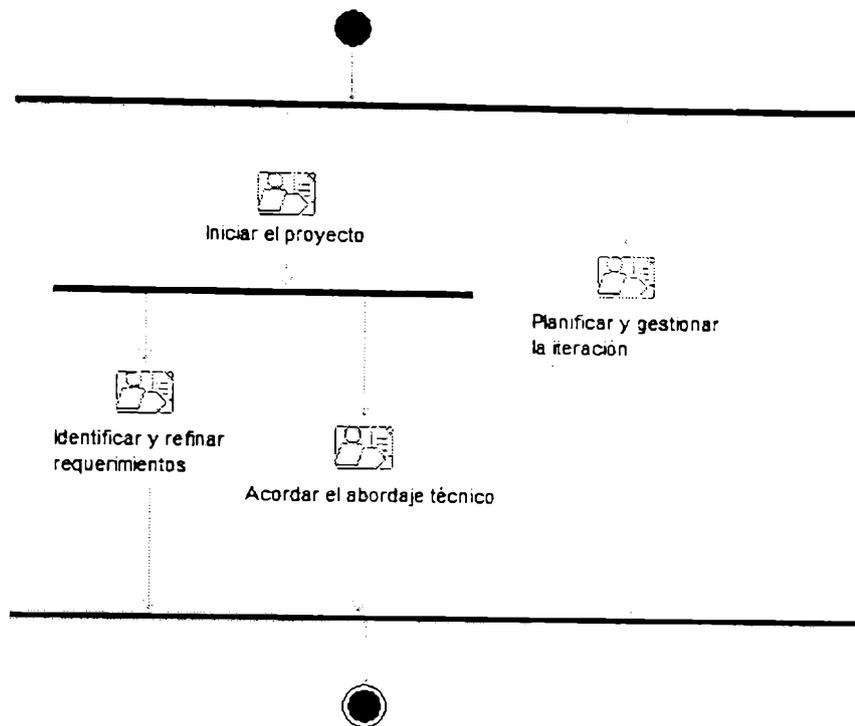


Figura 4.17: Actividades de una iteración en la Fase de Inicio

Las cuatro Actividades incluidas en este Patrón de Proceso consisten en:

- Actividad: Iniciar el proyecto** , lanzamiento del proyecto, acuerdo con Stakeholders sobre alcance y planes iniciales. Incluye dos tareas: Desarrollar la visión técnica (a cargo del Analista) y Planificar el proyecto (a cargo del Gerente de Proyecto).
- Actividad: Identificar y refinar requerimientos** , para detallar un conjunto de Casos de Uso y requerimientos de sistema. El WBS de la Actividad incluye: Identificar y bocetar requerimientos, Detallar Escenarios de Casos de Uso y Detallar Requerimientos comunes al Sistema (todas a cargo del Analista) y Crear casos de Prueba (bajo responsabilidad del Tester).
- Actividad: Acordar el enfoque técnico** , mediante la tarea Bocetar la arquitectura, con el Arquitecto como responsable principal, pero con la participación adicional del Gerente de Proyecto, el Analista, el Desarrollador y el Stakeholder.
- Actividad: Planificar y gestionar la iteración** , que consiste en iniciar la iteración y permitir a los miembros del equipo tomar tareas de desarrollo y luego monitorear y comunicar el estado del proyecto a los Stakeholders externos, identificando y manejando las excepciones y problemas. Esta Actividad se desarrollará en cada una de las iteraciones a lo largo de las cuatro fases del ciclo de vida.

En el **Hito Objetivo de Ciclo de Vida**, se evalúa el avance hacia el logro de los objetivos de la Fase y se toma una decisión para desplegar la solución el entorno operativo.

### 4.4.3. Fase de Elaboración

En esta fase el propósito es validar la arquitectura propuesta para la solución, especialmente en lo referido a su factibilidad y los compromisos que supone.

Los principales objetivos de la fase son: obtener una comprensión más detallada de los requerimientos; diseñar, implementar, validar y establecer la base de la arquitectura; mitigar los riesgos esenciales y producir estimaciones adecuadas de cronograma y costos. En la figura 4.18 se pueden ver las actividades que constituyen el flujo de trabajo de una iteración en esta fase.

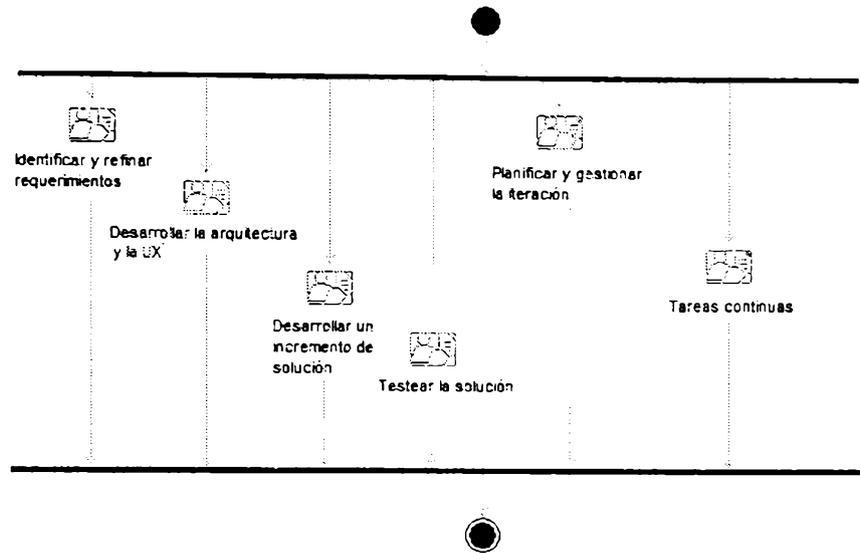


Figura 4.18: Actividades de una iteración en la Fase de Elaboración

El flujo de trabajo de esta Fase incluye:

**Actividad: Planificar y gestionar la iteración** incluida en la Fase anterior.

**Actividad: Identificar y refinar requerimientos** abordando otro conjunto de requisitos.

**Actividad: Desarrollar la arquitectura** consiste básicamente en la tarea Refinar la arquitectura, desarrollando en cada iteración un conjunto de requisitos de arquitectura para dar soporte a las actividades de desarrollo priorizadas.

**Actividad: Desarrollar un incremento de solución**, incluye el diseño, implementación, prueba e integración de la solución para los requisitos dentro del contexto de la iteración. En consecuencia, el WBS de este Patrón de Proceso se compone de: Diseñar la solución, Implementar las pruebas de desarrollador, Implementar la solución, Ejecutar pruebas de desarrollador, Integrar y construir, todas bajo la responsabilidad del Desarrollador.

**Actividad: Probar la solución**, desde una perspectiva del sistema, verificando la satisfacción de requerimientos. Incluye las Tareas Implementar las pruebas y Ejecutar las Pruebas, a cargo del Tester.

**Actividad: Actividades continuas**, todas las iteraciones poseen desde la fase de Elaboración una actividad que permite realizar tareas continuas fuera del cronograma, fundamentalmente efectuar Pedidos de Cambio por cualquiera de los roles del proyecto.

En el **Hito de Arquitectura del Ciclo de Vida**, se evalúa el avance hacia el logro de los objetivos de la Fase y se toma una decisión sobre mantener el alcance original, cambiarlo o terminar el proyecto.

#### 4.4.4. Fase de Construcción

El objetivo principal es desarrollar y verificar la solución de manera incremental. Las metas para lograrlo son: desarrollar iterativamente un producto completo que esté listo para ser transferido a la comunidad de usuarios; minimizar los costos de desarrollo y alcanzar algún grado de paralelismo.

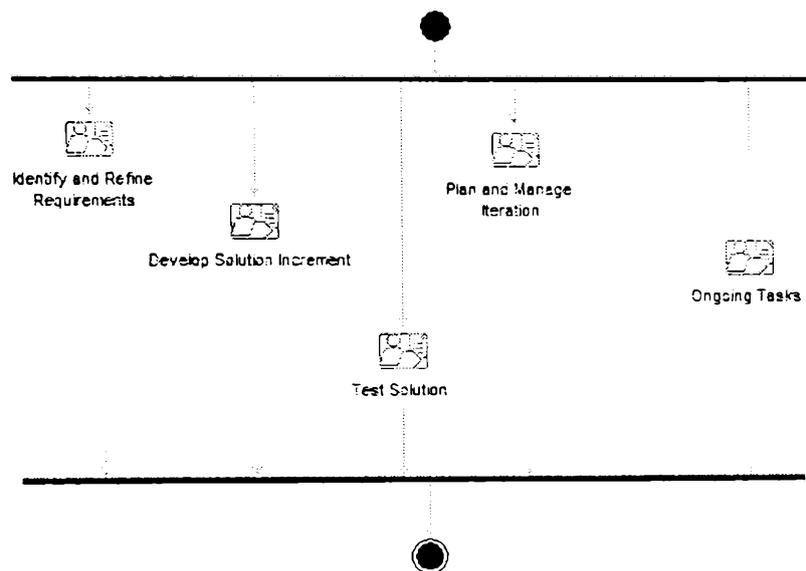


Figura 4.19: Actividades de una iteración en la Fase de Construcción

Como se observa en la figura 4.19 las actividades del WBS del Patrón de Proceso son las mismas que en una iteración de la Fase Elaboración excepto el trabajo sobre la arquitectura. Esto obedece a que en esta parte del proceso se sigue construyendo el sistema, pero ya ha sido superado el Hito de Arquitectura y por lo tanto consideramos que la solución de la misma debe quedar estable para el resto del trabajo.

El Hito de la Fase, **Capacidad Operativa Inicial**, se evalúa el avance hacia los objetivos y se decide desplegar la solución en el entorno operativo.

#### 4.4.5. Fase de Transición

El objetivo es desplegar la solución al entorno operativo y validarla, con: pruebas beta para validar que se satisfacen las expectativas de usuarios; lograr el acuerdo con interesados que el despliegue ha sido completo. Las actividades de una iteración de Fase de Transición se muestran en la figura 4.20

Las actividades de una iteración típica de esta Fase deben conducir a la convergencia hacia el despliegue del sistema, mediante dos actividades centrales: Desarrollar un incremento de solución y Probar la solución, acompañadas como en otras iteraciones por Planificar y gestionar la iteración y Realizar tareas continuas.

En el **Hito de la Release del Producto** se evalúa el logro de los objetivos y se decide liberar el producto.

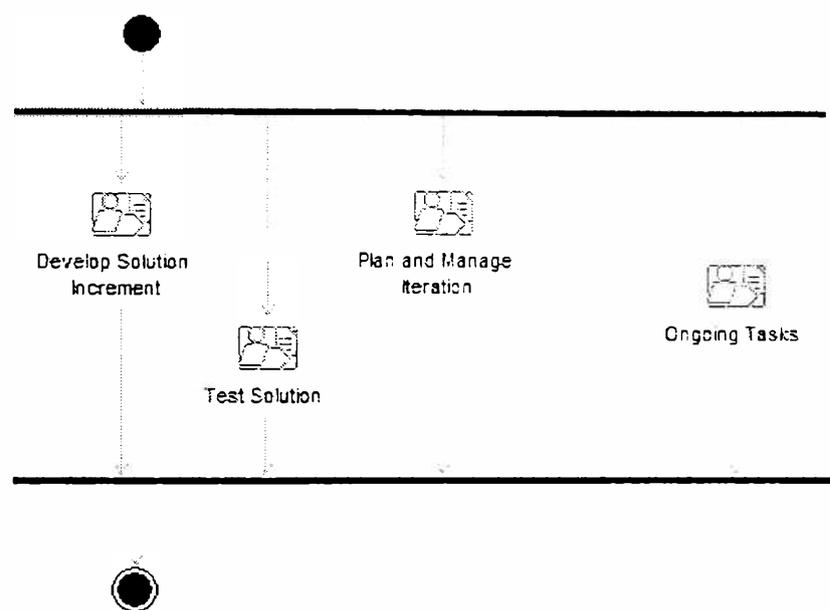


Figura 4.20: Actividades de una iteración en la Fase de Transición

## Capítulo 5

# Capacidad y madurez en usabilidad

### 5.1. Introducción

En el Capítulo 2 presentamos la noción de capacidad y madurez en usabilidad. Vamos a refinar un poco más este concepto y qué herramientas existen para que una organización que desarrolla software puede lograr un incremento de estos indicadores.

Ya mostramos que existe una relación entre un proceso de desarrollo centrado en el usuario y el nivel de usabilidad del producto resultante. Vimos también que la efectividad y eficiencia del proceso DCU está determinada por la capacidad en usabilidad de la organización. Señalamos además que en el mundo de la ingeniería de software, se han introducido los modelos para la evaluación de procesos como herramientas para identificar las debilidades y fortalezas de una organización que desarrolla software y luego utilizar esa información para impulsar acciones de mejora.

De manera similar, desde el campo de la usabilidad se han planteado modelos de capacidad en usabilidad que permitan realizar evaluaciones o assessments y obtener guía para acciones de mejora que levanten el nivel de la capacidad de una organización.

La revisión de la literatura, tanto en el campo de usabilidad como de ingeniería de software, muestra una diversidad interesante de aproximaciones a la evaluación o assessment de usabilidad:

- Nielsen: Modelo de madurez en usabilidad corporativa, descrito en [Nielsen 2006a;b]
- Ehrlich y Rohn: Etapas de aceptación de Diseño Centrado en Usuario [Ehrlich and Rohn 1994]
- Lundmark y Toresson: Modelo de Madurez en Usabilidad de Proyecto, en [Lundmark and Toresson 2007]
- Trillium, por Bell Canada [April and Coallier 1995]
- IBM: Evaluación de liderazgo en usabilidad, [Flanagan and Rauch 1995]
- Schaffer: Institucionalización de usabilidad, presentado en [Schaffer 2004]
- INUSE HCS: Modelo de madurez en usabilidad: escala de centralidad en lo humano, en [Earthy 1998]
- INUSE Processes: Modelo de madurez en usabilidad: procesos, en [Earthy 2000]

- ISO 18529: Modelo de madurez en usabilidad, [ISO/IEC 2000a]
- KESSU, de Timo Jokela Jokela [2001b]

## 5.2. Modelos de madurez en usabilidad

A continuación describimos las principales características de estas aproximaciones en el orden en que han sido mencionadas.

### 5.2.1. Madurez en Usabilidad Corporativa de Nielsen

Jakob Nielsen ha presentado un esquema de madurez evolutiva de usabilidad que atraviesa ocho etapas ([Nielsen 2006a;b]). No se trata de un verdadero modelo de evaluación, pero lo incluimos por tratarse de una de las personalidades más que más ha contribuido a la difusión y aceptación de usabilidad y DCU en los ámbitos corporativos en las últimas tres décadas. Las ocho etapas se describen brevemente a continuación:

**Etapas 1. Hostilidad hacia la usabilidad** La primera etapa es caracterizada por el eslogan “un buen usuario es un usuario muerto”. El equipo de desarrollo no quiere oír sobre los usuarios o sus necesidades. Su único interés radica en construir features y hacerlas funcionar en la computadora.

**Etapas 2. Usabilidad centrada en los desarrolladores** En esta etapa, el abordaje del equipo de diseño consiste en basarse en sus propias intuiciones sobre lo que es bueno para los usuarios. Después de todo, los desarrolladores también son usuarios y saben cuándo un sitio es fácil de usar. Además estos “usuarios” están disponibles para todas las reuniones de trabajo. En esta etapa no existe aporte explícito de recursos para el trabajo de usabilidad.

**Etapas 3. Usabilidad de skunkworks** La empresa comprende que no puede descansar exclusivamente en el buen juicio de sus diseñadores y desarrolladores sobre lo que es fácil de usar. Sin embargo la mayoría de las decisiones todavía se basan en ese juicio, se realizan algunos test de usuario parciales y adhoc. La etapa se caracteriza por la falta de un presupuesto asignado a esta actividad.

**Etapas 4. Presupuesto dedicado a usabilidad** Varios escenarios pueden casar que una compañía pueda invertir más en usabilidad. Un escenario habitual es que algún proyecto pequeño pero de gran impacto en la empresa y con una actividad importante de usabilidad, provoca la creación de un presupuesto dedicado a usabilidad.

**Etapas 5. Usabilidad gestionada** Existe un grupo oficial de usabilidad. El líder del grupo tiene control completo sobre todo el trabajo de usabilidad dentro de la organización.

**Etapas 6. Procesos sistemáticos de usabilidad** En este punto se reemplaza la idea de que la usabilidad es algo así como una poción mágica que se despliega sobre el proyecto por el reconocimiento de un proceso de diseño centrado en usuarios con actividades e hitos a alcanzar.

**Etapas 7. Diseño centrado en usuario integrado** Existen investigaciones de usuario formales, los proyectos se someten a revisiones en todo el ciclo de vida y no se emplean solo estimaciones de usabilidad, sino que existen métricas de usabilidad cuantitativas. Cada proyecto plantea las metas de usabilidad que deben alcanzarse.

**Etapas 8. Corporación conducida por usuarios** En este nivel máximo de madurez en la usabilidad, los datos de usuario no sólo definen proyectos individuales, sino que determinan los proyectos que la compañía debe o no comenzar y apoyar. En la experiencia el usuario se incluyen no solo el software sino toda la interacción con el cliente, tales como soporte y ventas.

### 5.2.2. Ehrlich y Rohn

El trabajo de Ehrlich y Rohn (Ehrlich and Rohn [1994]), al igual que el de Nielsen, no se puede considerar un modelo de evaluación de procesos. Las cuatro etapas que presenta son descritas muy brevemente por los autores. Sin embargo, también al igual que Nielsen, es necesario rescatar el esfuerzo para definir la capacidad en usabilidad. Las etapas que propone esta aproximación son:

**Escepticismo** organizaciones que nunca estuvieron involucradas con DCU. No tienen claro qué beneficios podrían obtener. Si existe algún experto en usabilidad, se lo incorpora tarde en los proyectos o no tiene influencia real.

**Curiosidad** existe apertura hacia los beneficios del DCU, pero sin una formación adecuada en la disciplina. Un experto en DCU influye en algunos aspectos superficiales del sistema

**Aceptación** La gente de DCU es parte del equipo desde el comienzo. Su rol y expertise se aprecian como una parte importante del desarrollo.

**Asociación** el equipo es una entidad unificada. Los sistemas no sólo son más usables sino más útiles. Algunos proyectos o parte de ellos son conducidos por la gente de DCU

### 5.2.3. Lundmark y Toresson

Mattias Lundmark y Johan Toresson presentaron un Modelo de Madurez en Usabilidad para proyectos desarrollado en Suecia y a partir de experiencias en Volvo IT [Lundmark and Toresson 2007]. El nivel de madurez se establece en una escala de 8 etapas. Cada etapa tiene asociados algunos atributos que deben ser reflejados por el proceso en análisis, tal como muestra la Tabla 5.1.

### 5.2.4. Trillium

Trillium [April and Coallier 1995] es un modelo de evaluación de dominio público para el desarrollo de productos de telecomunicaciones, desarrollado por Bell Canadá. Cubre varios procesos de desarrollo, incluyendo uno que denomina *Ingeniería de usabilidad* y que identifica varias prácticas.

Los autores sostienen que Trillium está basado en el Capability Maturity Model (CMM) del Software Engineering Institute [Paulk 1995].

El modelo de Trillium es jerárquico. En el nivel superior, define ocho áreas de capacidad para el desarrollo de producto, cada una de las cuales está compuesta de hojas de ruta. Una de esas áreas es *Soporte del cliente* donde el Ingeniero de usabilidad es una de esas hojas de ruta y contiene varias prácticas de usabilidad asociadas.

Trillium define cinco niveles de capacidad, desde el 1 (el más bajo, *desestructurado*) hasta el 5 (el superior, *completamente integrado*). Diferentes prácticas de ingeniería de usabilidad se categorizan en diferentes niveles de capacidad. La categorización significa que las prácticas en el nivel 2 son consideradas más fundamentales mientras que las ubicadas en el nivel 4 son relevantes sólo después que las de los niveles 2 y 3 se llevaron a cabo. Algunas de las prácticas indicadas para cada nivel son:

Etapa	Atributos
Etapa 0. Desconocimiento	(se desconoce la existencia de problemas de usabilidad)
Etapa 1. Conciencia de usabilidad	Reconocimiento de problemas de usabilidad en el producto desarrollado Actitudes positivas hacia la usabilidad
Etapa 2. Usabilidad considerada	Experto o entusiasta en usabilidad Proyecto influido por un ejecutivo campeón de usabilidad Presupuesto para usabilidad
Etapa 3. Usabilidad estructurada	Métricas de usabilidad Metas de usabilidad Estándares de diseño de interfaces Participación de un miembro del grupo oficial de usabilidad Testing de usabilidad
Etapa 4. Involucramiento de usuarios	Involucramiento de usuarios antes de que el desarrollo comience Involucramiento real de usuarios finales
Etapa 5. Usabilidad integrada	Presupuesto suficiente para un trabajo satisfactorio en usabilidad Seguimiento de la calidad de la experiencia del usuario Integración de usabilidad con otros procesos
Etapa 6. Proyectos centrados en usuarios	Gestión de proyecto centrada en usuario Proyecto integrante de una organización centrada en usuarios Uso de datos de experiencia de usuario recolectados
Etapa 7. Proyectos conducidos por usabilidad	Los datos de usuario son determinantes al definir el financiamiento del proyecto

Tabla 5.1: Modelo de Madurez en Usabilidad de Proyecto

**Nivel 1. Desestructurado** No se indican prácticas en este nivel

**Nivel 2. Repetible** Se evalúan productos competidores. Se visitan puestos de usuario antes de comenzar el desarrollo. Los usuarios se involucran en el proceso de diseño

**Nivel 3. Definido** Análisis comparado de productos competidores, en diferentes etapas del ciclo de vida. Se visita a los usuarios para conocer cómo usan sus sistemas. Se especifican metas de usabilidad con objetivos mensurables. Se utilizan prototipos para desarrollar todas las interfaces de usuario. Se desarrolla formalmente la documentación de usuario. El material de entrenamiento se verifica y valida antes de entregarlo.

**Nivel 4. Gestionado** Se proyecta la evolución de necesidades y habilidades de usuario. Se documenta en forma explícita el rationale para el diseño de las interfaces de usuario

**Nivel 5. Completamente integrado** No hay prácticas de usabilidad indicadas en este nivel

En un assessment con Trillium, se evalúa la capacidad analizando en qué medida se realizan las prácticas de usabilidad definidas en el modelo.

### 5.2.5. IBM, Evaluación de liderazgo en usabilidad

El modelo de IBM, presentado por Flanagan y Rauch en CHI95 ([Flanagan and Rauch 1995],[Rauch and Flanagan 1995]), cubre tanto los aspectos de habilidades y procesos como organizacionales. Define nueve atributos de madurez en la gestión de usabilidad clasificados en tres categorías:

- Organización
  1. Conciencia de la organización en todos los niveles donde la usabilidad desempeña un rol para el desarrollo de productos
  2. Actividades de la organización en todos sus niveles para asegurar un foco importante en usabilidad
  3. Acciones de mejora por parte de los gerentes para impulsar el foco en usabilidad
- Habilidades
  1. Habilidades de HCI del staff responsable de la usabilidad
  2. Recursos de HCI disponibles para las tareas de usabilidad
- Procesos
  1. Foco temprano y continuo en usuarios durante todo el desarrollo
  2. Diseño integrado de todos los elementos externos por equipos multidisciplinarios
  3. Testing de usuario temprano y continuo durante todo el desarrollo
  4. Diseño iterativo, con la habilidad de modificar el diseño sobre la base del feedback recibido de las actividades de usabilidad

Todos estos atributos se evalúan con referencia a un planteo de referencia (benchmarks). Por ejemplo para el atributo Conciencia de organización, podría ser: “la organización entiende el valor de las mediciones de la productividad y satisfacción del usuario” o “la organización valora las habilidades de HCI en el equipo de desarrollo”.

El modelo define cinco niveles de capacidad y benchmarks para cada atributo. La capacidad se evalúa separadamente para cada atributo en una escala de 1 a 5. Lamentablemente la única información públicamente disponible son los trabajos citados y no mencionan las bases teóricas u otros modelos que sirvieron de referencia.

### 5.2.6. El método de Schaffer

Eric Schaffer plantea modelo de madurez en usabilidad como parte de su proceso de “institucionalización de la usabilidad” ([Schaffer 2004]). Utiliza una escala de cinco niveles, donde 1 indica baja madurez y 5 la máxima.

Tal como se muestra en la Tabla 5.2, cada nivel está asociado con una cantidad de Actividades de Usabilidad, cuya realización es la que se toma como parámetro para indicar si el nivel de usabilidad fue alcanzado.

### 5.2.7. INUSE HCS

El Modelo de Madurez en Usabilidad INUSE: Escala de Centralidad en lo humano, [Earthy 1998] está basado principalmente en el Modelo de Madurez de sistemas Sherwood-Jones Sherwood-Jones [1995], en el Modelo IBM Flanagan and Rauch [1995], ISO 13407 ISO/IEC [1999] y las etapas de calidad de Crosby Crosby [1979]. Una de las dimensiones del modelo está dirigida a evaluar la centralidad de los aspectos humanos en la

Actividad	Nivel 1 Usab. inicial	Nivel 2 Campeón	Nivel 3 Infraestructura	Nivel 4 Staffing	Nivel 5 Rutinario
<b>Estrategia</b>					
<b>Estrategia escrita</b>		Completa	Completa	Completa	Completa
<b>Infraestructura</b>					
<b>Proceso de revisión de productos y sitios</b>	Parcial	Parcial	Completa	Completa	Completa
<b>Metodología DCU</b>			Completa	Completa	Completa
<b>Procesos de desarrollo integrados</b>					Completa
<b>Estándares corporativos de diseño</b>			Completa	Completa	Completa
<b>Proyectos de muestra</b>			Completa	Completa	Completa
<b>Educación y entrenamiento</b>					
<b>Entrenamiento continuo</b>					Completa
<b>Staff en usabilidad</b>	Parcial	Parcial	Completa	Completa	Completa
<b>Desarrolladores</b>				Completa	Completa
<b>Gestión</b>				Completa	Completa
<b>Staffing</b>					
<b>Campeón ejecutivo</b>		Completa	Completa	Completa	Completa
<b>Equipo de usabilidad</b>	Parcial	Parcial	Completa	Completa	Completa
<b>Expertos en Staff</b>				Completa	Completa
<b>Porc del staff en usabilidad</b>					Completa

Tabla 5.2: Modelo de Madurez en Usabilidad de Schaffer

organización. Define cinco niveles crecientes de madurez de los procesos cnetrados en las personas, desde *No reconocida* hasta *Insitucionalizada*.

El modelo define además un conjunto de prácticas de gestión que la organización debe implementar en cada nivel. De la misma forma que en Trillium, hay prácticas de ingeniería de usabilidad categorizadas en los niveles de capacidad. La Tabla 5.3 muestra los niveles de capacidad y las prácticas de gestión asociadas. El modelo contiene además varias definiciones de los términos empleados, guías de uso y algunos templates de documentos para instanciarlo.

### 5.2.8. INUSE Procesos

El proyecto INUSE dearrolló un modelo de evaluación de procesos basado en el formato de assesment definido por ISO 15504-2 ISO15504. El modelo define una nueva categoría de procesos: el Diseno Centrado en lo Humano (HCD por Human Centered Design, en inglés) que incluye siete procesos [Earthy 2000]. Este modelo es la base del ISO 18529 que utilizaremos como base para nuestro trabajo. Los detalles comunes entre INUSE Procesos e ISO 18529 serán descriptos más adelante en este mismo capítulo.

### 5.2.9. KESSU

Desde la Universidad de Oulu y con el desarrollo de análisis empíricos de grandes empresas finesas, Timo Jokela propuso en el 2001 el Modelo para procesos de diseno de usabilidad, llamado KESSU [Jokela 2001b].

El modelo se presenta como una interpretación de los modelos de ISO 13407 e ISO 18529, incluyendo según su autor definiciones más explícitas de los procesos de usabilidad y DCU.

KESSU define los procesos a partir de sus resultados, habitualmente entregables, Identifica siete procesos, seis corresponden a la ingeniería de usabilidad y uno al diseno de

<b>Nivel de capacidad</b>	<b>Prácticas de gestión</b>
<b>Nivel X: No reconocido</b>	-
<b>Nivel A: Reconocido</b>	Reconocimiento de problemas. La gerencia y el staff son concientes de la necesidad de mejorar los aspectos del sistema en desarrollo vinculados con su uso
<b>Nivel B: Considerado</b>	Calidad en entrenamiento de usuarios. Métodos de entrenamiento centrados en usuario
<b>Nivel C: Implementado</b>	Involucramiento activo de usuarios Métodos centrados en usuarios Técnicas de calidad en uso
<b>Nivel D: Integrado</b>	Integrar procesos de factores humanos con la calidad del sistema Gestionar la iteración de soluciones de diseño
<b>Nivel E: Institucionalizado</b>	Mejora sistemática de la calidad en uso Aceptación de habilidades centradas en usuario

Tabla 5.3: Prácticas de gestión en INUSE HCS

interacción. Como se observa en la Figura 5.1, KESSU divide el proceso Producir la solución de diseño de ISO 13407 en dos partes: Diseñar la Tarea del usuario y Diseñar la interacción.

Para el assessment, KESSU evalúa los resultados de cada proceso a la luz de atributos de performance.

- Cantidad. Cuanto más extensamente se produce un entregable, mayor es su score.
- Calidad. Examina la calidad y validez de los entregables
- Integración. Brinda un score proporcional al impacto del resultado de un proceso en el diseño y en otros procesos de usabilidad

Cada uno de estos atributos es valorado en una escala de cuatro niveles: No logrado, Logro parcial, Logro amplio y Logro completo.

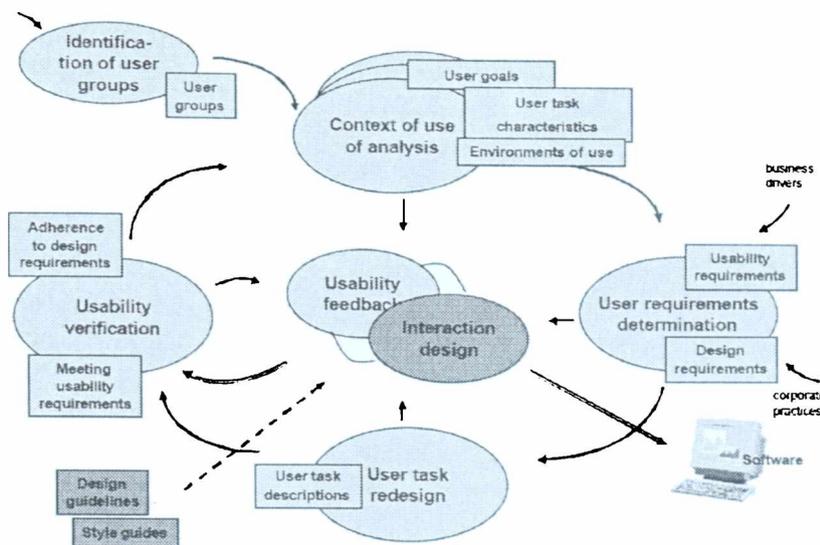


Figura 5.1: Procesos de Diseño en Usabilidad de KESSU

### 5.3. El Modelo de Madurez en Usabilidad de ISO

El Modelo de Madurez en Usabilidad propuesto por la Organización Internacional de Estandarización (MMU-ISO) está contenido en el reporte ISO TR 18529 “Descripciones de Procesos de Ciclo de Vida centrado en las Personas”, que reconoce como base a la Norma ISO 9241:210<sup>1</sup> y al trabajo del Proyecto INUSE[Earthy 2000].

El MMU-ISO, que fue desarrollado de manera iterativa y con revisión de expertos, intenta proveer una base para que quienes planifican sepan qué actividades centradas en los usuarios incluir en un proyecto en particular, así como asistir a quienes desean mejorar la manera en que sus equipos realizan tales actividades.

#### 5.3.1. Modelo de referencia

El MMU-ISO tiene como modelo de referencia al conocido como SPICE (*Software Process Improvement and Capability dEtermination* contenido en la Norma ISO 15504 [ISO/IEC 2006], que puede ser visto como la contraparte europea del CMMI-SEI [SEI 1995]. A su vez, ISO 15504 se derivó del ciclo de vida propuesto en ISO 12207 [ISO/IEC 1995] y algunos modelos de Madurez como Trillium [April and Coallier 1995] y el CMM-SEI.

El modelo ISO 15504 tiene dos dimensiones: **procesos** y **capacidades**. La dimensión de procesos define cinco categorías: Cliente-proveedor, Ingeniería, Soporte, Gestión y Organización. Dentro de cada una de estas categorías, cada proceso puede identificarse con un nivel de capacidad en una escala de seis grados: Incompleto, Realizado, Gestionado, Establecido, Predecible, Optimizado.

La forma de determinar el nivel de capacidad alcanzado en un proceso consiste en analizar cuáles atributos de dicho proceso se verifican según la evidencia recogida sobre:

- Nivel 1
  - 1.1 Performance del proceso
- Nivel 2
  - 2.1 Gestión de la performance

<sup>1</sup>En el momento de elaboración de esta Tesis, fue liberada la Norma 9241:210 que consiste básicamente en la incorporación al cuerpo de la 9241 del contenido de la anterior Norma ISO 13407

- 2.2 Gestión de los productos de trabajo
- Nivel 3
- 3.1 Definición del proceso
- 3.2 Despliegue del proceso
- Nivel 4
- 4.1 Medición del proceso
- 4.2 Control del proceso
- Nivel 5
- 5.1 Innovación del proceso
- 5.2 Optimización del proceso

Cada atributo del proceso implica una o más prácticas genéricas que permiten alcanzarlo. Cada práctica produce sus indicadores para asistir en la realización del assessment. Cada atributo es evaluado en una escala de 4 rangos: No alcanzado (0 - 15 %), Alcanzado parcialmente (16 - 50 %), Alcanzado ampliamente (51 - 85 %), Alcanzado completamente (86 - 100 %).

### 5.3.2. El contenido de MMU-ISO

Con la referencia como estructura del modelo de ISO 15504, MMU-ISO define siete áreas de proceso para el Desarrollo Centrado en el Usuario, de las cuales cinco están tomadas directamente de ISO 13407 (que ahora es ISO 9241-210, como ya se indicó). Las otras dos extienden la incumbencia del DCU del ámbito estricto del proyecto de desarrollo a la organización (*DCU1. Asegurar el contenido de DCU en la estrategia de sistemas*) y a la implantación en campo de los productos (*DCU7. Introducir y operar el sistema*). La capacidad en uno de estos procesos se mide en la escala de seis grados propuesta por ISO15504.

#### Dimensión de los Procesos DCU

Los procesos están descritos como prácticas que es necesario implementar para representar e incluir a los usuarios del sistema durante su ciclo de vida. Cada una está presentado como un Proceso con sus propósitos, actividades a realizar e indicadores de éxito. Presentamos aquí un resumen de cada proceso, ver el detalle completo en el Apéndice A.

El primer proceso, *DCU1. Asegurar el contenido DCU en la estrategia de sistemas*, intenta establecer y mantener el foco de la organización sobre los temas relacionados con los todos los interesados en el sistema final (no sólo los usuarios finales) en cada uno de los sectores de la organización que tenga influencia sobre el concepto, desarrollo, mantenimiento y soporte del software a desarrollar. Incluye actividades tales como: representar al usuario final, reunir información del mercado, definir y planificar las estrategias de sistemas considerando a los usuarios finales.

El proceso *DCU2. Planificar y gestionar el proceso* (DCU2), persigue el propósito de especificar la forma en que las actividades centradas en la persona se adaptan al ciclo de vida completo de procesos y la empresa. Busca asegurar que la planificación de un proyecto concreto ponga en el centro de las actividades a los usuarios finales, con actividades como: consultar a todos los involucrados, planificar la incorporación de usuarios en el desarrollo, identificar, seleccionar y gestionar técnicas de desarrollo centrado en usuarios y proveer todo el soporte posible al proceso DCU.

Los procesos DCU3 y DCU4 constituyen el corazón de una ingeniería de requerimientos centrada en usuarios. **DCU3. Especificar los requerimientos de los involucrados y la organización**, se propone establecer los requerimientos del sistema desde la organización y otros interesados, tomando en cuenta las necesidades, competencias y entorno de trabajo de cada involucrado relevante al sistema, mientras que **DCU4. Entender y especificar el contexto de uso**, tiene por objetivo identificar, clarificar y registrar las características de los involucrados, sus tareas y el entorno físico y organizacional en el que operará el sistema. Entre los dos procesos comprenden el conjunto de prácticas de ingeniería de requerimientos establecidas en la comunidad de HCI, tales como: identificar y definir involucrados, establecer objetivos de calidad de uso, evaluar riesgos de HW y SW para los usuarios, identificar las tareas de los usuarios finales, establecer los atributos de usuario, identificar el entorno técnico, físico y organizacional en que será utilizado el producto a desarrollar.

El ciclo de diseño y evaluación continua, que constituye un rasgo característico de los procesos centrados en usuario, están corporizados en los procesos DCU5 y DCU6. **DCU5. Producir soluciones de diseño**, busca que el equipo de proyecto sea capaz de crear soluciones potenciales de diseño mediante el uso de prácticas del estado del arte, la experiencia y conocimiento de los participantes y los resultados del análisis del contexto de uso, tal como fueron obtenidos de DCU3 Y DCU4. Todas las buenas prácticas de diseño encuentran cabida en las actividades de este proceso: generar un modelo de tareas, distribuir concientemente las funciones entre usuarios y sistemas, explorar alternativas de diseño, seleccionar y refinar alternativas, desarrollar prototipos, trabajar para la capacitación y entrenamiento de usuarios, etc.

En forma iterativa, las soluciones propuestas por DCU5, son revisadas en **DCU6. Evaluar diseños contra los requerimientos**. La intención es reunir feedback sobre el diseño en desarrollo proveniente de los usuarios y otras fuentes representativas, sin agotar la evaluación en aspectos puramente técnicos de performance. Para ello será necesario especificar el contexto de evaluación, evaluar todas la alternativas de diseño pertinentes tanto desde el punto de vista de sistemas como desde las condiciones de uso.

Finalmente, **DCU7. Introducir y operar el sistema**, se incluye para establecer los aspectos centrados en la persona para el soporte e implementación del sistema o producto desarrollado. Para ello será importante, además de gestionar el proceso de cambio, determinar el impacto en las actividades de usuarios, realizar las adaptaciones adecuadas, proveer capacitación, entrenamiento y soporte a los usuarios.

Como vemos, los procesos de ISO 18529 cubren el ciclo completo de la ingeniería de software y pueden analizarse en tres niveles: el de la organización (DCU1 y DCU7), el del desarrollo técnico del proyecto (DCU3, DCU4, DCU5 Y DCU6) y el de la gestión y control del mismo (DCU2), tal como intenta mostrar la Figura 5.2.

El diagrama muestra cómo los procesos DCU3 a 6 constituyen un ciclo técnico sólido en el corazón de todo desarrollo y que se repite varias veces durante un proyecto típico. DCU 2 cubre la gestión y control. Utiliza la información generada por el ciclo DCU 3-6 y conecta el ciclo de vida de los procesos DCU con los otros procesos en el desarrollo de sistemas. DCU 1 actúa como conexión del ciclo de vida DCU con los procesos de gestión y dirección y tiene en cuenta el futuro de los sistemas, al mismo tiempo que establece los límites y metas para los proyectos que ingresan al ciclo DCU 3-6 cuya implementación se realiza en el marco de DCU 7. Este último proceso está relacionado con el uso del sistema, estableciendo una conexión entre los procesos de soporte desde DCU con las otras actividades de soporte en el proyecto.

## Dimensión de los Niveles de Capacidad y Madurez

Ya mencionamos que MMU-ISO utiliza los seis niveles de capacidad de SPICE:

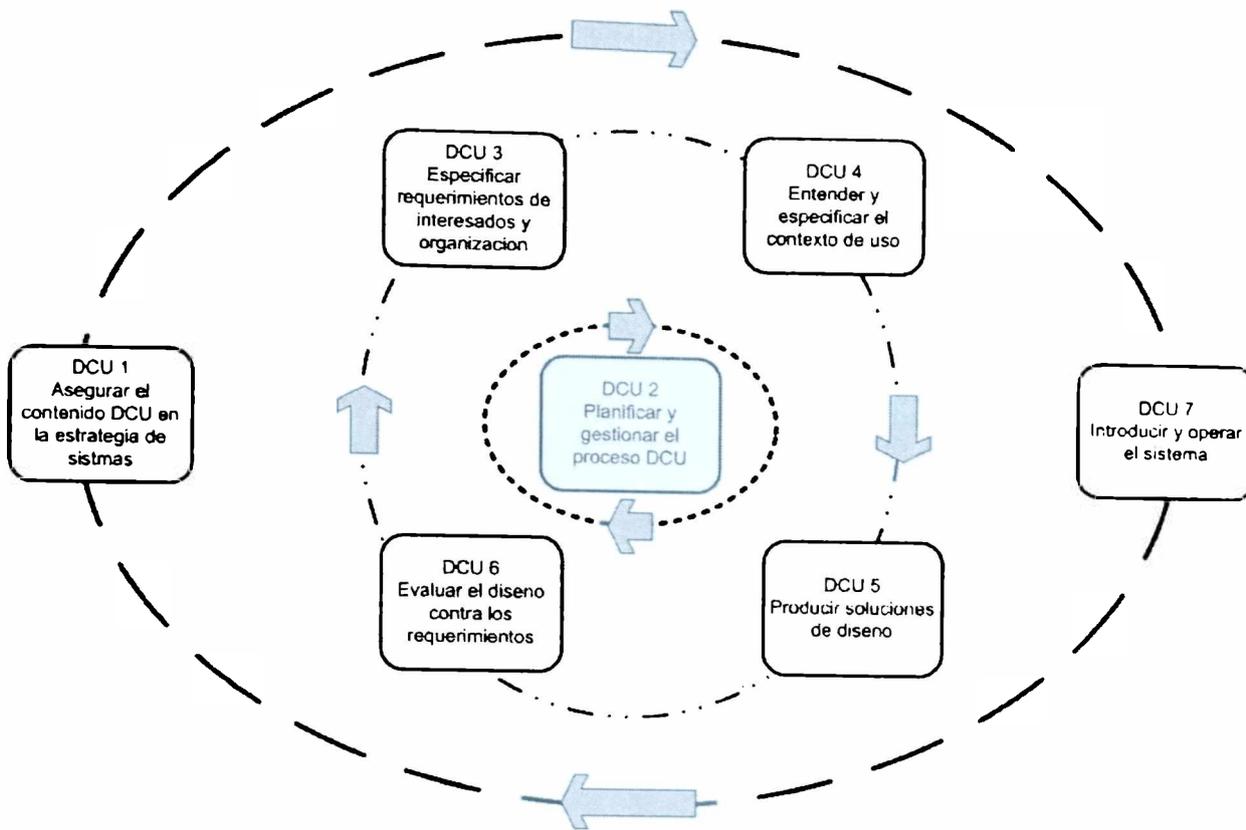


Figura 5.2: Ciclo de vida de procesos DCU en MMU-ISO

- Nivel 0. Incompleto. En este nivel la organización no es capaz de realizar el proceso.
- Nivel 1. Realizado. Los individuos llevan a cabo las actividades y el proceso alcanza los propósitos definidos.
- Nivel 2. Gestionado. Se conocen y controlan los requerimientos de calidad, tiempo y recursos para el proceso. El proceso gestionado entrega productos de trabajo de calidad aceptable dentro de los tiempos definidos y las necesidades de recursos.
- Nivel 3. Establecido. El proceso se desarrolla en una forma especificada por la organización y los recursos están definidos. El proceso establecido asegura el despliegue de un proceso definido, basado en buenos principios de ingeniería de sistemas.
- Nivel 4. Predecible. La performance del proceso está dentro de los límites preestablecidos de recursos y calidad. El proceso predecible se realiza en forma consistente dentro de límites de control para alcanzar sus metas.
- Nivel 5. En optimización. La organización puede ajustar en forma confiable los procesos a requerimientos particulares. El proceso optimizado adapta su performance para satisfacer necesidades actuales y futuras del negocio y cumplir sus metas de negocio confiablemente.

Para ubicar cada uno de los procesos de DCU en uno de los niveles de capacidad, se realiza un análisis que permita caracterizarlos de acuerdo con una serie pre establecida de atributos deseables. Estos atributos de definición de procesos son acumulativos, es decir que en cada nivel de capacidad se espera que se alcancen todos los atributos deseables de los niveles inferiores.

ISO 15504 propone además que en cada uno de los niveles de capacidad, el proceso se acompañe de prácticas de gestión generales. Estas prácticas dan al equipo y a la orga-

nización el grado de control necesario sobre todo el proceso de desarrollo. Los elementos que permiten construir los atributos en este ámbito:

- Las características de la performance de una práctica de gestión que brindan evidencia de su implementación
- Los recursos y elementos de infraestructura que dan soporte a la gestión del proceso
- Los procesos asociados que dan soporte a las prácticas de gestión

Estos atributos intentan establecer una evidencia objetiva de que las prácticas de gestión asociadas con un atributo de proceso se están llevando a cabo.

El detalle completo de atributos de producto y de gestión para cada nivel de capacidad se incluye en el Apéndice B. Planteamos aquí los atributos de los Niveles 1, 2 y 3, que constituyen el foco de esta tesis. En el Nivel 0-Incompleto, obviamente no existen atributos a identificar, mientras que los niveles 4 y 5 requieren un assessment sobre aspectos de la organización que exceden y no pueden ser cubiertos sólo con la configuración de un determinado método de desarrollo.

**Nivel 1: Proceso realizado** El logro de este nivel se demuestra con la identificación del siguiente atributo de performance del proceso: el grado en el cual los productos de trabajo resultantes son producidos a partir de los productos de input mediante la realización de las prácticas que comprende este proceso. La práctica de gestión para lograr este atributo del proceso es Asegurar que las prácticas básicas son realizadas para satisfacer el propósito del proceso.

**Nivel 2: El proceso gestionado** En este nivel debemos identificar tanto atributos de la gestión de la performance como de la gestión de productos de trabajo.

En cuanto a la gestión del proceso, se valora el grado con el cual el proceso se gestiona para producir productos de trabajo dentro del tiempo establecido y los requerimientos de recursos. Las prácticas de gestión relacionadas son:

- Identificar requerimientos de recursos para permitir la planificación y el control del proceso
- Planificar la performance del proceso identificando las actividades y los recursos alocados de acuerdo con los requerimientos
- Implementar las actividades definidas para lograr el propósito del proceso
- Gestionar la ejecución de las actividades para producir los productos dentro del tiempo establecido y los requerimientos de recursos.

Respecto de los productos de trabajo, la evaluación se enfoca en el grado con el cual los productos de trabajo son documentados y controlados para satisfacer sus requerimientos funcionales y no funcionales. Para ello será importante identificar la ejecución de las siguientes prácticas de gestión:

- Identificar requerimientos para la integridad y calidad de los productos de trabajo
- Identificar las actividades necesarias para alcanzar los requerimientos de integridad y calidad de los productos de trabajo.
- Gestionar la configuración de los productos de trabajo para asegurar la integridad.
- Gestionar la calidad de los productos de trabajo para asegurar que satisfacen sus requerimientos funcionales y no funcionales.

**Nivel 3: El proceso establecido** El proceso establecido asegura el despliegue de un proceso definido basado en buenos principios de ingeniería de sistemas.

En cuanto a la definición de los procesos, la evaluación analizará el grado con el cual un proceso dado se define con un estándar adecuado para contribuir a las metas de la organización. Las prácticas de gestión relacionadas son:

- Identificar entre las disponibles en la organización la definición de proceso estándar que es apropiada al propósito del proceso y las metas de negocio de la organización
- Adaptar el proceso estándar para obtener un proceso definido apropiado al contexto
- Implementar el proceso definido para lograr su propósito consistente y repetidamente y soportar la meta de negocio definida para la organización
- Proveer feedback en el proceso estándar desde la experiencia de usar el proceso definido

Se deberá analizar también el proceso utiliza recursos humanos calificados e infraestructura adecuada para contribuir a las metas definidas para la organización. Las prácticas de gestión relacionadas son:

- Definir las competencias de recursos humanos requeridas para soportar la implementación del proceso definido
- Definir los requerimientos de infraestructura del proceso para soportar la implementación del proceso definido
- Proveer recursos humanos calificados adecuadamente para satisfacer las competencias definidas
- Proveer infraestructura adecuada para el proceso de acuerdo con las necesidades definidas.

En resumen para evaluar el logro del Nivel 1, deberemos asegurar que las prácticas básicas planteadas en cada uno de los siete procesos DCU de MMU-ISO encuentran algún grado de realización tal que permita la obtención del propósito del proceso respectivo.

Para evaluar el logro del Nivel 2, no alcanzará con que la configuración del método de desarrollo proponga las prácticas básicas y alguna gestión que asegure su realización. También deberán verificarse los atributos de gestión del proceso que permiten obtener los resultados esperados con los tiempos y recursos previstos, así como la gestión de calidad de los productos de trabajo para asegurar que satisfacen los requerimientos acordados para el proyecto.

Para calificar un logro de Nivel 3, será necesario asegurar que el proceso resulta adecuado a las necesidades de cada proyecto, dentro de la organización. Esta adaptación se consigue mediante actividades de tailoring que consistirán en instanciaciones de la configuración del proceso en cada proyecto que lo amerite.

## Capítulo 6

# Conformidad de OpenUP/Basic al MMU-ISO

### 6.1. Introducción

Tal como se ha manifestado en capítulos anteriores, la contribución que esta tesis se propone es dar soporte para un proceso de desarrollo de software basado en OpenUP que logre conformidad con el MMU-ISO.

Ya hemos visto la descripción de OpenUP y también hemos analizado los objetivos, contenido y niveles del MMU-ISO. Es necesario establecer ahora el grado de conformidad que la versión actualmente disponible denominada OpenUP/Basic alcanza para los procesos y actividades MMU-ISO.

Para conseguirlo hemos diseñado y puesto en práctica un assessment informal de OpenUP/Basic que presentamos en este capítulo.

El carácter de informal del assessment no está dado por la relajación de los criterios de evaluación o el recorte del universo de atributos evaluables, sino por la simplificación de los componentes burocrático-administrativos de un proceso formal de evaluación y certificación.

Asumimos para esta tarea un escenario ideal en el que una organización o equipo de proyecto configura e instancia un proceso consistente en la implementación completa de OpenUP/Basic, con todas las prácticas y contenido tanto de Método como de Proceso. Ciertamente se trata de una situación idealizada que difícilmente se lleve a cabo en la práctica, pero nos permite realizar un assessment que refleje el mayor grado de capacidad en MMU-ISO alcanzable con OpenUP en su versión Basic.

En las secciones que siguen describimos el método de assessment utilizado, los resultados obtenidos y las conclusiones sobre el nivel de conformidad MMU-ISO que entendemos es posible alcanzar con OpenUP/Basic. Para cerrar el capítulo incluimos las necesidades de ampliación y modificación que detectamos y que servirán de insumo para la propuesta de extensión que permita obtener un proceso basado en OpenUP que conforme al MMU-ISO en los niveles de Capacidad 1, 2 y 3.

### 6.2. Metodología de assesment

Un proceso típico de assesment comienza por definir los objetivos o el alcance del assesment, que proveen la base para la planificación del proceso de evaluación.

El escenario del assesment comienza con la información a los involucrados en la organización sobre los objetivos y propósitos del assesment. Luego se obtienen datos sobre

los temas relacionados, en nuestro caso con el diseño centrado en usuario, mediante la lectura de documentos y reportes y se realizan entrevistas. De las entrevistas habitualmente se obtiene el status de las características de una organización que pueden impactar en los procesos DCU. Para determinar qué datos son relevantes y qué características de la organización investigar se utiliza la estructura del modelo de assessment elegido. Luego se analizan los datos obtenidos y se interpretan a la luz de los criterios planteados por el modelo de assessment. Finalmente se elaboran perfiles de capacidad (como el que muestra la Figura 6.1) y se informa los resultados a la organización.

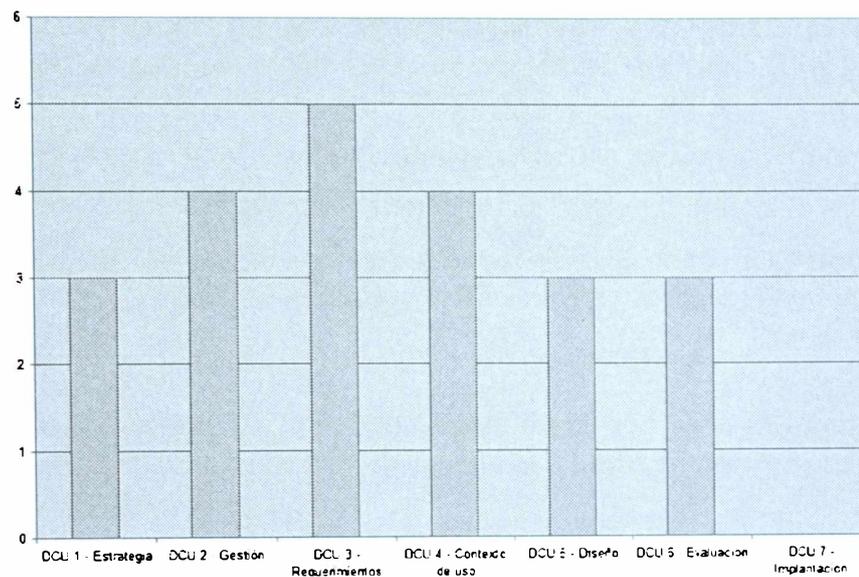


Figura 6.1: Perfil de Capacidad para un Nivel

El método de análisis consistirá entonces en identificar para cada uno de los atributos de Capacidad que se definen en los procesos de MMU-ISO el nivel de logro con el que OpenUP/Basic permite satisfacerlos, dado el supuesto anterior de una instanciación completa de la configuración.

Como input para recolectar evidencia de logro de cada atributo utilizamos la especificación completa de OpenUP/Basic Release 1.5.0.1 [Eclipse 2008c]. Al decir completa indicamos que cualquier contenido, ya sea de método o proceso, que permita interpretar que de ser empleado en un proyecto concreto indicaría la satisfacción de algún atributo de MMU-ISO, será considerado evidencia suficiente de tal capacidad.

Estamos intentando determinar la Capacidad de OpenUP/Basic en los Niveles 1 a 3 del MMU-ISO, eso implica:

- Para el Nivel 1, analizar si cada práctica base está cubierta
- Para el Nivel 2, analizar el grado de capacidad que alcanza OpenUP para los atributos de gestión de la performance del proceso y de los productos
- Para el Nivel 3, analizar el grado de capacidad mostrado respecto de los atributos de definición de procesos y recursos.

En consecuencia, el ciclo de evaluación es el siguiente:

1. Tomamos un proceso DCU del MMU-ISO
2. Tomamos un nivel a evaluar

3. Para cada atributo del Proceso en el Nivel elegido analizamos la especificación de OpenUP/Basic y buscamos Contenidos o Procesos que permitan satisfacer el atributo. Determinamos el grado de logro del atributo MMU-ISO en un ranking de 4 niveles:
  - N: **No hay evidencia** del logro de la práctica definida (Score numérico: 0)
  - P: El desarrollo de las actividades incluidas en OpenUP permite alcanzar un **logro parcial** del atributo (Score numérico mayor que 0 y menor o igual que 0.3)
  - A: El desarrollo de las actividades incluidas en OpenUP permite alcanzar un **logro amplio** del atributo (Score numérico mayor que 0.3 y menor o igual que 0.7)
  - C: El desarrollo de las actividades incluidas en OpenUP permite alcanzar un **logro completo** del atributo (Score numérico mayor que 0.7)
4. En los casos de duda sobre el logro entre dos niveles para una determinada práctica, aplicamos el beneficio de la duda y calificamos en el nivel superior
5. El proceso se repite para el próximo atributo del nivel.
6. El proceso se repite para el próximo nivel, hasta que no se encuentra evidencia de la realización de ninguna práctica en este nivel o hasta que se alcanza el límite superior establecido para la evaluación.
7. El ciclo 2-6 se repite para el siguiente proceso DCU de MMU-ISO

El formato de la hoja de evaluación empleada (basada en [Earthy 2000]) se muestra en la tabla 6.1. Se utiliza una hoja para cada Proceso DCU de MMU-ISO.

MMU-ISO Atributo a medir	Evidencia OpenUP Práctica EPF	Evidencia OpenUP Contenido de Proceso	Evidencia OpenUP Contenido de Método	Score
<i>Nivel 1</i>				
Atributo 1.1				
Atributo 1.n				
Score combinado de atributos 1.1 a 1.n				
Atributo 2.1				
Atributo 2.n				
Score combinado atributos 2.1 a 2.n				
Score combinado de Nivel 1				
<i>siguiente nivel</i>				
...	...	...	...	...

Tabla 6.1: Esquema de hoja de assesment

En las celdas correspondientes a la columnas **Evidencia de OpenUP** indicamos los elementos identificados como evidencia del logro del atributo MMU-ISO a medir.

La columna **Práctica EPF**, nos sirve para identificar cuál o cuáles de las Prácticas de la Librería del Eclipse Proces Framework contiene suficiente evidencia para alcanzar algún nivel de logro del atributo. La correspondiente a **Contenido de Proceso** nos sirve para precisar las *Actividades*, *Patrones de proceso* o *Procesos para despliegue* implicados en tal logro, mientras que en la correspondiente a **Contenido de Método** cargamos para cada uno de los Procesos señalados los *Roles*, *Artefactos* o *Tareas* particularmente involucrados en ese logro. Para cada uno de los atributos de MMU-ISO se indica el grado de logro en la escala mencionada (traducida a valores numéricos para facilitar los cálculos de promedios). Luego realizamos un promedio del logro de todos los atributos para una

clase (Performance, Gestión de performance, Gestión de Producto, Definición de Proceso, Definición de Recursos de Proceso) y con estos promedios se establece la medición de logro del nivel de capacidad para el proceso en cuestión.

En el Apéndice B se incluye la hoja de evaluación completa con todos los atributos utilizados para los niveles 1, 2 y 3 evaluados y los resultados para OpenUP/Basic. Como ejemplo, la figura C.1 muestra el sector correspondiente a la evaluación del Proceso DCU1.

DCU 1 Asegurar el contenido DCU en la estrategia de sistemas				
Prácticas y procesos	Prácticas	Contenidos de proceso	Contenidos de Método	Score
<b>Nivel 1 Proceso realizado</b>				
<b>AP1.1 Atributo de performance del proceso</b>				
DCU 1.1 Representar los stakeholders	Equipo integrado. Visión compartida. Planificación en dos niveles. Desarrollo conducido por UC	Iniciar el proyecto. Identificar y refinar requerimientos	Definir visión. Planificar proyecto. Encontrar y bocetar requerimientos	0.3
DCU 1.2 Analizar el mercado				
DCU 1.3 Definir y planificar la estrategia de sistemas				
DCU 1.4 Reunir respuesta de mercado				
DCU 1.5 Analizar tendencias de usuarios				
Score combinado para los atributos (DCU 1.1 to 1.5)				0.3
Score combinado para este nivel				0.3
<b>Nivel 2 Proceso Gestionado</b>				
<b>AP2.1 Atributo de gestión de la performance</b>				
PG2.1.1 Identificar requerimientos de recursos	Equipo integrado. Visión compartida. Planificación en dos niveles. Desarrollo conducido por UC	Iniciar el proyecto. Identificar y refinar requerimientos	Definir visión. Planificar proyecto. Encontrar y bocetar requerimientos	0.3
PG2.1.2 Planificar la performance del proceso	Equipo integrado. Visión compartida. Planificación en dos niveles. Desarrollo conducido por UC	Iniciar el proyecto. Identificar y refinar requerimientos	Definir visión. Planificar proyecto. Encontrar y bocetar requerimientos	0.3
PG2.1.3 Implementar las actividades definidas	Equipo integrado. Visión compartida. Planificación en dos niveles. Desarrollo conducido por UC	Iniciar el proyecto. Identificar y refinar requerimientos	Definir visión. Planificar proyecto. Encontrar y bocetar requerimientos	0.3
PG2.1.4 Gestionar la ejecución de las actividades	Equipo integrado. Visión compartida. Planificación en dos niveles. Desarrollo conducido por UC	Iniciar el proyecto. Identificar y refinar requerimientos	Definir visión. Planificar proyecto. Encontrar y bocetar requerimientos	0.3
Score combinado para los atributos (PG2.1.1 to 2.1.4)				0.3
<b>AP 2.2 Atributo de Gestión de producto de trabajo</b>				
PG2.2.1 Identificar requisitos	Equipo integrado. Visión compartida. Planificación en dos niveles. Desarrollo conducido por UC	Iniciar el proyecto. Identificar y refinar requerimientos	Definir visión. Planificar proyecto. Encontrar y bocetar requerimientos	0.3
PG2.2.2 Identificar las actividades necesarias	Equipo integrado. Visión compartida. Planificación en dos niveles. Desarrollo conducido por UC	Iniciar el proyecto. Identificar y refinar requerimientos	Definir visión. Planificar proyecto. Encontrar y bocetar requerimientos	0.3
PG2.2.3 Gestionar la configuración de productos de trabajo	Equipo integrado. Visión compartida. Planificación en dos niveles. Desarrollo conducido por UC	Iniciar el proyecto. Identificar y refinar requerimientos	Definir visión. Planificar proyecto. Encontrar y bocetar requerimientos	0.3
PG2.2.4 Gestionar la calidad de productos de trabajo	Equipo integrado. Visión compartida. Planificación en dos niveles. Desarrollo conducido por UC	Iniciar el proyecto. Identificar y refinar requerimientos	Definir visión. Planificar proyecto. Encontrar y bocetar requerimientos	0.3
Score combinado para los atributos (PG2.2.1 to 2.2.4)				0.3
Score combinado para este nivel				0.3
<b>Nivel 3 Proceso establecido</b>				
<b>AP3.1 Atributo de definición de proceso</b>				
PG3.1.1 Identificar la definición del proceso standard				
PG3.1.2 Adaptar el proceso standard				
PG3.1.3 Implementar el proceso definido				
PG3.1.4 Proveer feedback				
Score combinado para los atributos (PG3.1.1 to 3.1.4)				0
<b>AP 3.2 Atributo de recursos de proceso</b>				
PG3.2.1 Definir las competencias de rhh				
PG3.2.2 Definir los requisitos de infraestructura del proceso				
PG3.2.3 Proveer rhh cualificados				
PG3.2.4 Proveer infraestructura de proceso adecuada				
Score combinado para los atributos (PG3.2.1 to 3.2.4)				0
Score combinado para este nivel				0

Figura 6.2: Resultados para el Proceso DCU 1

### 6.3. Resultados

A continuación, mostramos con mayor detalle el tipo y grado de logro alcanzado por cada Proceso DCU en cada uno de los Niveles. El listado completo de elementos de proceso y método de OpenUP/Basic involucrados se incluye en el Apéndice C.

### 6.3.1. Capacidad en DCU1 (Asegurar el contenido de DCU en la estrategia de sistemas)

En este proceso debemos encontrar evidencia de que OpenUP/Basic asegura el foco en temas relacionados con usuarios (y stakeholders en general) en cada una de las áreas relacionadas con la idea, desarrollo, mantenimiento y soporte del sistema.

#### Nivel 1 - Proceso realizado

En este nivel sólo encontramos evidencia del primer atributo *Representar a los stakeholders*. Las tareas de *Definir la visión*, *Planificar el proyecto* y *Encontrar y bocetar requerimientos* incluyen entre los roles participantes al *Stakeholder*. Sin embargo, la especificación no da pautas concretas sobre la manera de identificar a todos los stakeholders pertinentes del proyecto. Por otra parte, es importante marcar que estas tareas están incluidas en Patrones de Proceso sólo para las Fases de Inicio y Elaboración, lo que podría dificultar el trabajo de “abogado de los usuarios” a lo largo de todo el proyecto. Por esa razón es que consideramos que el atributo está sólo parcialmente cubierto.

No existe evidencia de que la sola aplicación de OpenUP/Basic permita el logro de los otros cuatro atributos de este nivel, ni la realización o utilización de los productos de trabajo que normalmente están asociados a las prácticas requeridas por ellos, tales como informes del mercado, análisis de tendencias y planteo de estrategias DCU.

En consecuencia, la capacidad manifestada por OpenUP/Basic para el Proceso DCU1 en el Nivel 1 es **Parcial**.

#### Nivel 2 - Proceso gestionado

Los mismos elementos de método que permiten satisfacer el atributo DCU1.1. brindan evidencia de logro de las prácticas de gestión tanto de la performance del proceso como del producto. Sin embargo, al estar restringido a las tareas relacionadas con la identificación de los stakeholders no podemos considerar que la capacidad sea siquiera amplia.

El contexto de las prácticas OpenUP *Equipo integrado*, *Visión compartida*, *Planificación en dos niveles* y *Desarrollo conducido por Casos de uso* permite abordar de manera adecuada la planificación, gestión y control de los productos de trabajo vinculados a la participación de los stakeholders.

La capacidad manifestada en este Nivel es **Parcial**.

#### Nivel 3 - Proceso establecido

No hay evidencia de ningún tipo Prácticas de Gestión vinculadas a la definición de procesos o de recursos a incluir en lo concerniente con el Diseño Centrado en el Usuario.

La capacidad manifestada en este Nivel es **Nula**.

### 6.3.2. Capacidad en DCU2 (Planificar y gestionar el proceso DCU)

Buscamos aquí evidencia sobre la forma en que las actividades centradas en usuario están incluidas en todos los planes del proyecto y en las actividades de gestión.

#### Nivel 1 - Proceso realizado

Encontramos evidencias de logro de todos los atributos de este nivel, aunque con un grado dispar.

Los dos primeros (*Consultar a los involucrados* e *Identificar y planificar la participación de usuarios*) alcanzan un logro amplio, fundamentalmente gracias a la inclusión

de Prácticas tales como *Equipo integrado*, *Visión compartida*, *Planificación en dos niveles* y *Desarrollo iterativo*. Las tareas que incluyen estas prácticas (por ejemplo *Planificar el proyecto*, *Planificar la iteración*, *Definir la visión técnica*) permiten mantener a los involucrados en el loop de consultas y decisiones.

Sin embargo, a la hora de profundizar el enfoque DCU con la selección de métodos y técnicas adecuadas, la planificación de actividades DCU específicas y el soporte a tales procesos, OpenUP no dice nada explícito y a menos que en el equipo existan especialistas vinculados con estas áreas el logro por parte de la especificación del método sólo puede considerarse como parcial.

Por ejemplo, la única referencia de OpenUP respecto de la incorporación de usuarios y stakeholders en el equipo es un párrafo dentro de la *Orientación Designar recursos para un proyecto* (traducción propia):

*Incluya stakeholders. Los stakeholders, incluyendo los orientados al negocio como los usuarios finales y los técnicos como el staff de operaciones, pueden agregar valor significativo al equipo. En lugar de sólo entrevistarlos para obtener información o pedirles que revisen su trabajo, por qué no incluirlos como participantes activos del equipo?*

Y lo mismo para la posible inclusión de especialistas en interacción y factores humanos

*Incluya especialistas para tareas de corto alcance y específicas. Los especialistas pueden agregar valor aún en un equipo de desarrollo ágil, particularmente cuando tienen habilidades y experiencia muy específica que los miembros del equipo carecen. A menudo puede ser muy efectivo traer un especialista al equipo por un período breve para ayudar con una tarea específica (como la instalación y el setup de un servidor de aplicaciones o para tomar parte en una revisión simplemente*

Con una especificación tan somera, es difícil garantizar que dadas ciertas condiciones de proyecto, el equipo se asegurará que todas las instancias del proyecto serán adecuadamente planificadas y gestionadas desde un punto de vista centrado en los involucrados, que en todas las fases se abogue por el DCU y se provea el soporte adecuado a los restantes integrantes del equipo.

Algunos de los productos de trabajo que darían cuenta de estas tareas y que no están apropiadamente incluidos en OpenUP/Basic son los procedimientos para establecer actividades específicas de DCU e integrarlas con otras actividades de diseño, cómo garantizar que el conocimiento del contexto de uso (ver DCU4) y los requerimientos de usuarios y organizaciones conducen el proceso de diseño, que se emplea el prototipado para mejorar y refinar el diseño, que un número adecuado de usuarios y sus representantes son consultados para identificar contexto de uso.

En consecuencia, en el análisis global del Nivel encontramos que la capacidad alcanzable es de **Amplia**.

## Nivel 2 - Proceso gestionado

Algo similar ocurre con el Nivel 2. Existe evidencia de todas las prácticas de gestión requeridas para el nivel, pero concentradas en las actividades relacionadas con algún tipo de inclusión de involucrados en la planificación del proyecto y de cada iteración. Esto no implica necesariamente la evaluación de diferentes técnicas y la selección de la más apropiada para alguna fase del proyecto, así como tampoco el control necesario para asegurar que un número y calificación adecuada de usuarios y otros involucrados prestan el feedback necesario en cada etapa del proceso (y que ese feedback es convenientemente utilizado para guiar el diseño).

Por estas razones encontramos que la capacidad alcanzable para Nivel 2 en el Proceso DCU2 es **Parcial**.

### Nivel 3 - Proceso establecido

Otorgando el beneficio de la duda, podríamos considerar que las prácticas de gestión del nivel vinculadas con la identificación y definición de recursos tienen algún grado de logro (en particular en lo referido a identificar las necesidades de recursos humanos para involucrar a usuarios en el proyecto). Lamentablemente no hay evidencia de requisitos de OpenUP referidos a los diferentes tipos de usuarios o stakeholders a incluir en el proyecto.

La capacidad en este Nivel es **Nula**.

#### 6.3.3. Capacidad en DCU3 (Especificar los requerimientos de stakeholders y organización)

Se trata en este proceso de determinar la capacidad de OpenUP/Basic para alcanzar una identificación y especificación apropiadas de los requerimientos para el sistema, tanto de la organización como de los usuarios finales.

### Nivel 1 - Proceso realizado

Todos los atributos del Nivel 1 están cubiertos en OpenUP. Tres de ellos alcanzan un logro amplio: *Clarificar y documentar las metas del sistema*, *Evaluar los riesgos para los stakeholders*, *Generar requerimientos de stakeholders y organización*. Las actividades vinculadas con la definición de una *Visión compartida* y la *Identificación iterativa de requerimientos y soluciones* contribuyen a ese logro.

En cambio, los restantes atributos son cubiertos en una medida mucho menor. No hay evidencia en OpenUP/Basic que asegure una Definición completa y detallada de los stakeholders.

Tampoco encontramos evidencia sobre una definición integral del sistema, que considere el contexto completo de la situación, no sólo del software involucrado.

Finalmente, respecto de la fijación de objetivos de calidad de uso. OpenUP es demasiado escueto, apenas en la Orientación *Desarrollo de especificación de requerimientos de sistema* se indica como todo método para la identificación y especificación de requisitos de usabilidad, lo siguiente (traducción propia):

1. *Identifique los temas claves de usabilidad observando las tareas críticas, los perfiles de usuario, las metas de sistema y los problemas de usabilidad anteriores.*
2. *Elija el estilo adecuado para expresar los requerimientos:*
  - *Estilo de performance: especifique con qué rapidez los usuarios pueden aprender varias tareas y cómo pueden realizarla después del entrenamiento.*
  - *Estilo de defectos: más que medir los tiempos de tareas, identifique los defectos de usabilidad y especifique con qué frecuencia pueden ocurrir.*
  - *Estilo de guía: especifique la apariencia general y el tiempo de respuesta de la interfaz de usuario con referencia a algún estándar bien definido.*
3. *Escriba los requerimientos reales,*

La ponderación de los diferentes grados de logro y utilizando el beneficio de la duda, consideramos que el logro de Capacidad DCU3 en nivel es **Amplio**.

### Nivel 2 - Proceso gestionado

Los cuatro atributos de gestión vinculados con la performance del proceso están cubiertos parcialmente por métodos de OpenUP. En particular, la administración de las prácticas que evidencian logro amplio en el Nivel 1. Lo mismo ocurre con otros tantos atributos de gestión de productos de trabajo.

Lamentablemente, no hay evidencia de que un proceso OpenUP/Basic incluya prácticas de gestión de performance y producto con la misma amplitud para el resto de los atributos.

La evaluación global del Nivel 2 para este proceso es **Parcial**.

### Nivel 3 - Proceso establecido

En este nivel sólo podemos advertir cierta evidencia de logro para los atributos vinculados con los recursos del proceso, pero no con la definición misma de procesos (consideración de alternativas, adaptaciones al proyecto, etc.).

En consecuencia OpenUP/Basic resulta de capacidad **Nula** para DCU3 en el Nivel 3.

### 6.3.4. Capacidad en DCU4 (Entender y especificar el contexto de uso)

Pretendemos encontrar aquí evidencia de que OpenUp asegura una buena especificación de contexto de uso del sistema, incluyendo los involucrados, sus tareas, el entorno físico y el organizacional.

#### Nivel 1 - Proceso realizado

Este proceso complementa a DCU3 en la visión centrada en usuarios de la ingeniería de requerimientos. Por tanto, es lógico que al menos en el Nivel 1 todos los atributos encuentren algún grado de logro.

Lamentablemente, sin embargo, sólo es posible hallar buena evidencia de capacidad OpenUP/Basic respecto de la identificación y documentación de las tareas del usuario, particularmente gracias al enfoque de Desarrollo conducido por Casos de Uso.

La consideración de aspectos significativos de los usuarios (características físicas, culturales, educativas, etc. no relacionadas exclusivamente con su rol respecto del sistema) o las condiciones del contexto de la organización, el ambiente físico y técnico, apenas requieren alguna descripción en el documento de Vision técnica.

En resumen, la capacidad manifestada respecto de Nivel 1 en DCU4 por OpenUP/Basic es **Parcialmente cubierta**.

#### Nivel 2 - Proceso gestionado

Los defectos de logro en el Nivel 1 se propagan al 2. La especificación de OpenUP muestra evidencia de prácticas de gestión de performance y producto en lo relacionado con las tareas del usuario vinculadas directamente al sistema. Es obvio que aquellas especificaciones que apenas son descritas en su versión básica, no tendrán asociadas prácticas para gestionar la performance del proceso que permita obtenerlas en tiempo y con la calidad necesaria, como tampoco habrá indicación de tareas para validar la calidad de los artefactos de trabajo.

En este sentido, es notable la carencia de indicaciones para gestionar actividades y productos de trabajo que especifiquen de manera adecuada los elementos del entorno de

organización, técnico y físico donde será empleado el sistema y que pueden derivar en requerimientos no identificados directamente por los usuarios.

La capacidad demostrada en este Nivel es **Parcial**.

### Nivel 3 - Proceso establecido

Como hemos visto para procesos anteriores, la evidencia para cubrir este Nivel debería asegurar que la organización o el grupo de desarrollo se plantea un paso previo a la planificación del proyecto y de cada iteración, decidiendo las mejores implementaciones de tareas y artefactos para cubrir las necesidades del proyecto.

En este sentido, sólo encontramos evidencia parcial en las prácticas OpenUP Visión integradora y Desarrollo conducido por Casos de uso para las fases de Inicio y elaboración respecto, lo cual en cierto modo es lógico (el pase a la Fase de Construcción ocurre cuando el Hito de Arquitectura es alcanzado). Sin embargo, el proceso de entrega de la Fase de Construcción admite iteraciones sobre la identificación y refinamiento de requerimientos, lo cual debería requerir el tailoring del proceso antes de asumir cualquier versión estandarizada como correcta.

La capacidad de este Nivel es **Nula**.

#### 6.3.5. Capacidad en DCU5 (Producir soluciones de diseño)

En este proceso buscamos evidencia de que OpenUP/Basic garantiza la creación de soluciones de diseño mediante el uso de las prácticas que son estado del arte en DCU, aprovechando la experiencia y conocimiento de todos los participantes e involucrados y tomando en cuenta los resultados del análisis de contexto de uso realizado en DCU4.

### Nivel 1 - Proceso realizado

Encontramos evidencia de logro en cuatro de los ocho atributos del nivel. OpenUP/Basic contiene elementos de proceso y método que aseguran el enfoque sobre la distribución adecuada de funciones entre usuarios y sistemas y la generación iterativa e incremental de propuestas de diseño. Por ejemplo en las tareas *Bocetar la arquitectura* o *Diseñar una solución*.

Lamentablemente, no ocurre lo mismo respecto de un abordaje integral del diseño del sistema, incluyendo las tareas y puestos de trabajo del usuario. Tampoco brinda OpenUP/Basic evidencia suficiente de un enfoque de diseño basado en prototipos, modelos o maquetas, característica distintiva de los procesos centrados en usuario.

Finalmente no existe ningún tipo de evidencia respecto de los atributos vinculados con la producción de elementos de entrenamiento y soporte a los usuarios.

Por lo tanto, a pesar de tener un logro razonable de los primeros atributos mencionados, la carencia respecto del resto lleva la capacidad de este proceso en el Nivel 1 a **Parcial**.

### Nivel 2 - Proceso gestionado

Al igual que en procesos anteriores, para este nivel se puede encontrar evidencia de prácticas de gestión para la performance y los items de trabajo mencionados en el Nivel 1. En particular, para la creación y evolución incremental de la arquitectura y el diseño de soluciones de sistema (Prácticas OpenUP *Arquitectura evolutiva* y *Diseño evolutivo*).

La capacidad demostrada es de logro **parcial**.

### Nivel 3 - Proceso establecido

No encontramos evidencia de logro para este nivel. La capacidad es **Nula**.

### 6.3.6. Capacidad en DCU6 (Evaluar los diseños contra los requerimientos)

Buscamos evidencia de que OpenUP asegure la prueba y recolección de feedback de los usuarios e involucrados en general sobre el diseño en desarrollo.

#### Nivel 1 - Proceso realizado

Como es lógico, la evidencia recogida para este proceso se concentra en la práctica OpenUP *Testing concurrente*. Allí aparece el tipo de prueba que puede estar vinculada con usuarios (a diferencia de la Prueba de desarrollador, incluida en la Práctica Desarrollo conducido por pruebas o TDD). Los productos de trabajo involucrados son el *Caso de Prueba*, el *Log de Prueba* y el *Guión de la Prueba*.

Los elementos de Proceso y Método relacionados con ellos permiten cubrir de alguna forma tres de los seis atributos del Nivel. En efecto, podemos reconocer evidencia para satisfacer parcialmente:

- Especificar y validar el contexto de evaluación, particularmente con las tareas de *Crear casos de Prueba e Implementar guiones de prueba*.
- Evaluar prototipos para mejorar el diseño, en la *Actividad Testear la solución* (que subsume las Tareas *Implementar pruebas y Ejecutar pruebas*). Aunque las especificaciones OpenUP no hablan específicamente de prototipos, podemos considerar que la realización de esta actividad durante todas las iteraciones, en particular de la Fase elaboración, tienen el mismo efecto y, como señala la descripción de esa Actividad “Stakeholders and end-users also may also be involved in performing tests to accept the release.”
- Evaluar el sistema para chequear que se satisfacen los requerimientos, la evidencia del ítem anterior aplica también en este.

Lamentablemente, OpenUP/Basic no proporciona evidencia respecto de:

- Evaluar prototipos iniciales para definir requerimientos de sistema, por cuanto esta forma de elicitación no constituye parte mandatoria en la práctica Desarrollo conducido por Casos de uso
- Evaluar el sistema para verificar que se siguieron las prácticas requeridas, debido a que como vimos en los procesos DCU1 y DCU2, OpenUP no asegura la adhesión del proyecto a estándares y sugerencias de usabilidad y calidad de uso en general
- Evaluar el sistema para asegurar que continuamente se cumplen metas organizacionales y necesidades de usuario, por cuanto, como veremos en DCU7, OpenUP no aborda tareas de proyecto concernientes a la implantación del sistema en la Fase de Transición.

Algunos de los productos de trabajo que un proceso de evaluación centrado en usuario generaría y de los cuales no se encuentra evidencia en OpenUP son: listado de defectos de calidad de uso, Métricas de ergonomía empleadas, Registro en audio y video de pruebas, Logs de observación de usuario, Transcripción de entrevistas.

Por lo expuesto, consideramos que la capacidad demostrada por OpenUP en este Nivel es **Parcial**.

### Nivel 2 - Proceso gestionado

Encontramos aquí evidencia de Prácticas de gestión, tanto para el proceso como para los productos, respecto de los artefactos y tareas que mencionamos en el Nivel 1 y que permitieron cubrir la mitad de los atributos.

En las cuatro fases se llevan adelante las Tareas mencionadas para el Nivel 1, aunque ubicadas lógicamente en diferentes patrones de proceso. En Inicio, *Crear pruebas* constituye el método de aproximación al testing dentro de la *Actividad Identificar y Refinar requerimientos*, para pasar en las Fases de Elaboración y Construcción a constituir casi con entidad propia la *Actividad Testear la solución*.

Como resulta lógico no hay ningún tipo de práctica de gestión prescripta para aquellos atributos que no han sido cubiertos en el nivel 1. Por lo tanto, al igual que en el anterior, asociamos una **logro Parcial** al actual Nivel 2.

### Nivel 3 - Proceso establecido

No encontramos evidencia de logro en este Nivel.

#### 6.3.7. Capacidad en DCU7 (Presentar y operar el sistema)

El propósito del proceso es garantizar que los aspectos centrados en usuario son considerados al implantar el sistema desarrollado y dar soporte posterior a todos los involucrados.

Hemos visto al describir OpenUP/Basic que todos los aspectos posteriores a la construcción y testing del sistema no son abordados por el método, asumiendo que el resto de la organización donde se desarrolla el proyecto tiene establecidos actividades, tareas y artefactos que garanticen una buena transición del producto.

### Nivel 1 - Proceso realizado

Con este planteo inicial es obvio que no encontraremos evidencia de logro en ninguno de los niveles de MMU-ISO. La capacidad de OpenUP/Basic para este proceso es **Nula** en el Nivel 1 y, en función del mecanismo de assessment no es procedente continuar con los restantes niveles.

### Nivel 2 - Proceso gestionado

No aplica por falta de logro de Nivel anterior.

### Nivel 3 - Proceso establecido

No aplica por falta de logro de Nivel anterior.

## 6.4. Perfiles de capacidad y posibilidades de extensión

El análisis cuantitativo de los resultados globales del assessment muestra que OpenUP/Basic no alcanza un logro completo de ninguno de los atributos que MMU-ISO establece para los niveles 1 a 3. Incluso son muy pocos los que pueden mostrar un logro amplio (apenas en 11 atributos sobre un total de 156 revisados).

Las Figuras 6.3, 6.4 y 6.5 muestran la distribución de atributos para cada proceso DCU el tipo de logro alcanzado por los atributos del proceso en cada nivel de capacidad.

Al comenzar a analizar los resultados por cada uno de los Niveles, podemos señalar que OpenUP/Basic alcanza un aceptable grado de conformidad MMU-ISO para el Nivel 1 (Proceso realizado) en las actividades centrales del desarrollo centrado en usuario: requerimientos, diseño, construcción y evaluación. En el Nivel 2 (Proceso gestionado), solo

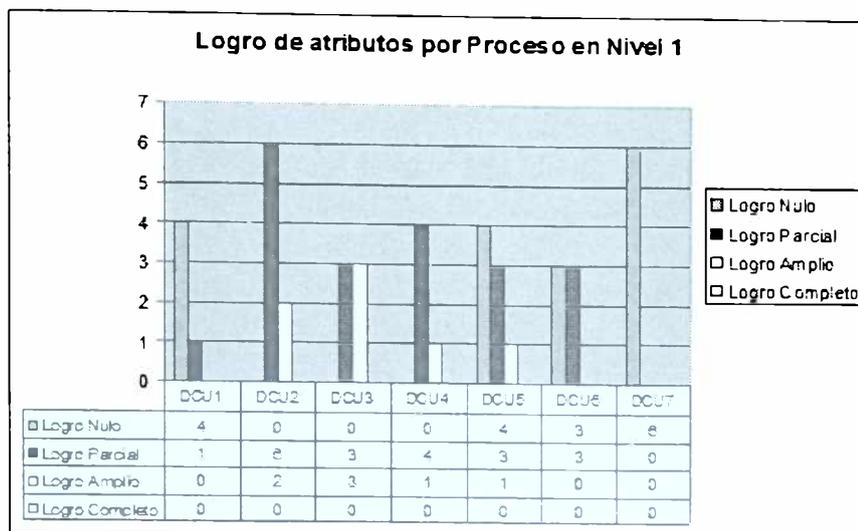


Figura 6.3: Cantidad de atributos por logro alcanzado para el Nivel 1

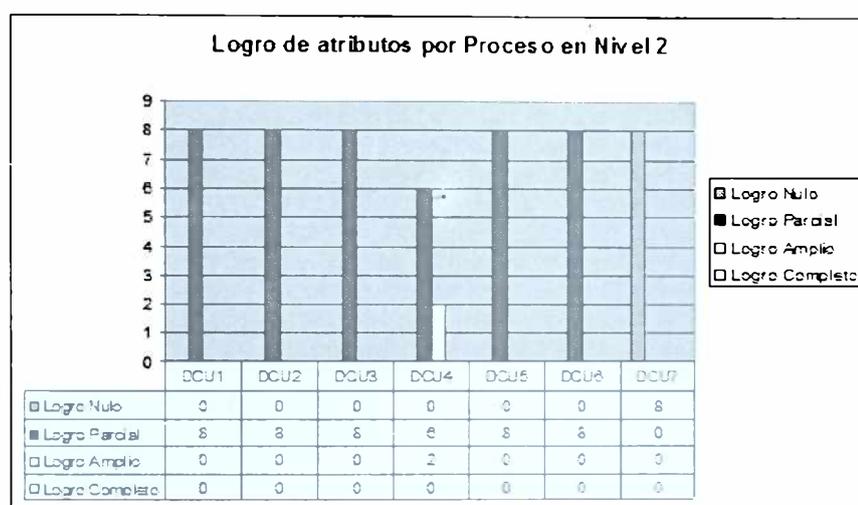


Figura 6.4: Cantidad de atributos por logro alcanzado para el Nivel 2

DCU4 (Comprender el contexto de uso) alcanza un logro aceptable, es algo menor para los DCU 1,2,3,5 y 6 y nula para DCU7. En el Nivel 3 prácticamente no existe logro por parte de OpenUP/Basic.

En la Figura 6.6 mostramos el perfil de capacidad de cada uno de los procesos DCU en los tres niveles del MMU-ISO que nos interesan.

Surge claramente que las únicas áreas que logran algún grado de conformidad son las del núcleo central indicado por la figura 5.2 (página 67): DCU3 a DCU6 más el proceso de planificar y gestionar el desarrollo. Es razonable que en virtud de los objetivos de OpenUP, los procesos relacionados con la implantación en la empresa u organización de un desarrollo centrado en usuario alcancen una logro mínima cuando se trata de pensar estrategias (DCU1) nula, a la hora de efectuar transiciones e instalaciones de sistemas (DCU7).

Todas las prácticas de gestión (*Planificación en dos niveles, Equipo integrado, Desarrollo iterativo, Ciclo de vida riesgo-valor, Gestión grupal del cambio*) muestran alguna evidencia de logro DCU, aunque ninguna alcanza para poder calificar con una C (completa) o A (amplia). En todos los casos, como detallaremos más adelante, se trata de algunos elementos de método (o partes de los mismos) que, de ser llevados a cabo por recursos con background en HCI y diseño de interacción alcanzarían un logro parcial de procesos

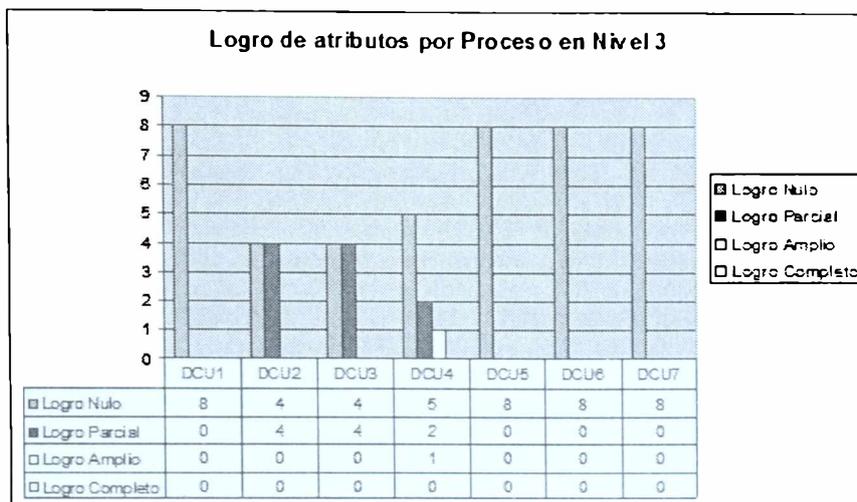


Figura 6.5: Cantidad de atributos por logro alcanzado para el Nivel 3

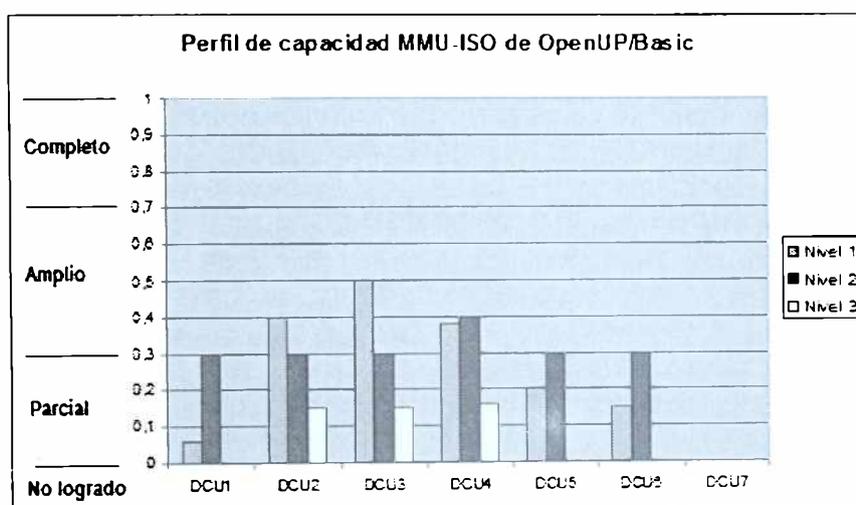


Figura 6.6: El perfil de capacidad por proceso DCU en niveles 1 a 3 de MMU-ISO

DCU MMU-ISO.

Las prácticas técnicas, con excepción de *Integración continua* y *Desarrollo conducido por pruebas (TDD)*, también exhiben un logro parcial, particularmente los objetivos de *Visión compartida* y *Arquitectura y diseño evolutivos*. En el caso del *Desarrollo conducido por Casos de Uso*, se da la misma situación respecto de la necesidad de contar con recursos experimentados en HCI y DCU (lo cual no constituye un requerimiento en OpenUP/Basic) ya que, por ejemplo, la identificación y especificación de stakeholders podría resultar incompleta o parcial para la elaboración posterior de especificaciones de tareas y contextos de uso.

El planteo de esta tesis es proveer un proceso de desarrollo basado en OpenUP que permita alcanzar una conformidad con MMU-ISO en los Niveles de capacidad 1, 2 y 3. Para ello consideramos necesario alcanzar una especificación que cubra completamente el Nivel 1 (al menos en los procesos DCU2 a DCU6) y alcance al menos una Capacidad Amplia en los Niveles 2 y 3, para todos los procesos.

Para alcanzar un perfil de capacidad de cobertura completa para el nivel 1, dependiendo del resultado que mostramos en el assessment necesitaremos tres tipos de acciones para cubrir las brechas identificadas:

- Completar: agregar o modificar contenidos de método y proceso para pasar de la

cobertura amplia a la completa.

- Ampliar: extender OpenUP/Basic para que los procesos en los que se evaluó cobertura parcial se alcance al menos el nivel siguiente
- Incluir: generar el contenido de método y los procesos necesarios para que los procesos que no están cubiertos en OpenUP/Basic según el assessment realizado, puedan alcanzar un nivel Amplio de logro en un nuevo assessment.

La Tabla 6.2 lista los atributos MMU-ISO cuya capacidad debe ser mejorada por OpenUP para alcanzar una Capacidad Completa en el Nivel 1, indicando qué tipo de acción es necesaria en cada uno de ellos.

Procesos DCU y actividades incluidas	Evaluated	Completar	Ampliar	Incluir
<b>DCU.1 Asegurar el contenido DCU en la estrategia de sistemas</b>				
DCU.1.1 Representar los stakeholders	0.3		X	
DCU.1.2 Analizar el mercado				X
DCU.1.3 Definir y planificar la estrategia de sistemas				X
DCU.1.4 Reunir respuesta de mercado				X
DCU.1.5 Analizar tendencias de usuarios				X
<b>DCU.2 Planificar y Gestionar el proceso DCU</b>				
DCU.2.1 Consultar a los stakeholders	0.7	X		
DCU.2.2 Planificar la participacion de usuarios	0.7	X		
DCU.2.3 Seleccionar metodos y tecnicas centradas en usuarios	0.3		X	
DCU.2.4 Asegurar el enfoque centrado en usuario en el equipo	0.3		X	
DCU.2.5 Planificar el proceso de diseo centrado en usuario	0.3		X	
DCU.2.6 Gestionar el proceso centrado en usuario	0.3		X	
DCU.2.7 Abogar por las actividades centradas en usuario	0.3		X	
DCU.2.8 Proveer soporte para el diseo centrado en usuario	0.3		X	
<b>DCU.3 Especificar los requerimientos de usuario y de organizacion</b>				
DCU.3.1 Clarificar y documentar las metas del sistema	0.7	X		
DCU.3.2 Definir los stakeholders	0.3		X	
DCU.3.3 Evaluar los riesgos para stakeholders	0.7	X		
DCU.3.4 Definir el sistema	0.3		X	
DCU.3.5 Generar requerimientos de stakeholders y de la organizacion	0.7	X		
DCU.3.6 Establecer la calidad en objetivos de uso	0.3		X	
<b>DCU.4 Entender y especificar el contexto de uso</b>				
DCU.4.1 Identificar y documentar tareas de usuario	0.7	X		
DCU.4.2 Identificar y documentar atributos significativos de usuario	0.3		X	
DCU.4.3 Identificar y documentar entorno organizacional	0.3		X	
DCU.4.4 Identificar y documentar el entorno técnico	0.3		X	
DCU.4.5 Identificar y documentar entorno fisico	0.3		X	
<b>DCU.5 Producir soluciones de diseo</b>				
DCU.5.1 Alocar funciones	0.3		X	
DCU.5.2 Producir modelo de tareas compuesto		X		X
DCU.5.3 Explorar el diseo del sistema	0.7			
DCU.5.4 Desarrollar soluciones de diseo	0.3		X	
DCU.5.5 Especificar el sistema				X
DCU.5.6 Desarrollar prototipos	0.3		X	
DCU.5.7 Desarrollar entrenamiento de usuarios				X
DCU.5.8 Desarrollar soporte de usuarios				X
<b>DCU.6 Evaluar el diseo contra los requerimientos</b>				
DCU.6.1 Especificar y validar el contexto de evaluacion	0.3		X	
DCU.6.2 Evaluar prototipos tempranos para definir los requerimientos de diseo				X
DCU.6.3 Evaluar prototipos para mejorar el diseo	0.3		X	
DCU.6.4 Evaluar el sistema para chequear que se satisfacen los requerimientos	0.3		X	
DCU.6.5 Evaluar el sistema para chequear que se siguieron las practicas requeridas				X
DCU.6.6 Evaluar el sistema en uso ...				X
<b>DCU.7 Introducir y operar el sistema</b>				
DCU.7.1 Gestionar el cambio				X
DCU.7.2 Determinar el impacto sobre la organizacion y los stakeholders				X
DCU.7.3 Customisacion y diseo local				X
DCU.7.4 Proveer entrenamiento a usuarios				X
DCU.7.5 Dar soporte a usuarios en las actividades planificadas				X
DCU.7.6 Asegurar la conformidad del lugar de trabajo a la legislación de ergonomía				X

Tabla 6.2: Atributos a mejorar para dar conformidad a MMU-ISO en Nivel 1

Para generar un proceso que pueda ser configurado para dar cumplimiento a MMU-ISO en niveles de Capacidad y Madurez 2 y 3, será necesario que todos los atributos de esos niveles puedan mostrar evidencia de logro con cobertura amplia.

En los capítulos siguientes presentamos la propuesta de extensión de OpenUP para alcanzar estos objetivos.

## Parte III

# Soporte para obtener Capacidad y Madurez en Usabilidad con OpenUP

## Capítulo 7

# Una extensión de EPL y OpenUP para conformar al MMU-ISO

### 7.1. Introducción

En el Capítulo 6 hemos visto que el gap entre OpenUP/Basic y MMU-ISO puede ser caracterizado en tres aspectos:

- Los procesos DCU 1 (Estrategia de sistemas) y DCU 7 (Implantación) casi no están cubiertos
- El proceso DCU 2 (Gestión) tiene una cobertura superficial
- En los restantes procesos, que constituyen el núcleo duro de un desarrollo centrado en usuarios, tienen una cobertura dispar, extendida en varios de sus atributos, pero incompleta.

Cuando presentamos el MMU-ISO en el Capítulo 5 vimos que el Nivel 1 de capacidad consiste en la cobertura de todas las prácticas de base de los siete procesos DCU y que los Niveles 2 y 3 de Capacidad se alcanzan con la incorporación de atributos de gestión de procesos y gestión de productos de trabajo. Por tanto, para poder obtener un proceso con base en OpenUP que pueda alcanzar la conformidad MMU-ISO al menos en los Niveles mencionados, será necesario realizar una tarea en dos etapas:

1. Para lograr el Nivel 1 de Capacidad en todos los Procesos DCU, deberemos rellenar con Contenido de Métodos todos los huecos detectados en las prácticas de base.
2. Para alcanzar la Capacidad de los Niveles 2 y 3, deberemos incorporar los Contenidos de Procesos que cubran los atributos de gestión de la performance de procesos y productos sobre las prácticas de base.

#### 7.1.1. La creación de Métodos para la Biblioteca de Prácticas del EPF

El Método de Autoría de Métodos (MAM) para la EPL [Eclipse 2008b] es una aproximación a la creación de métodos centrada en arquitectura, orientada a la calidad y basada en prácticas. Provee las guías necesarias para generar métodos dentro de la Biblioteca de Prácticas del EPF (EPL) y documentarlos adecuadamente.

El MAM está compuesto de varias Orientaciones del tipo Hoja de Ruta y del tipo Guía que, a partir de la denominada *La autoría en la EPL*, conducen al ingeniero de procesos a través del EPF para generar los cambios, extensiones, configuraciones, etc que sean necesarias para obtener un proceso de desarrollo adecuado al contexto de un proyecto u organización específicos.

La primera tarea es establecer el entorno de autoría adecuado, que implica tanto el proceso de autoría como las herramientas necesarias para ejecutarlo. El propio MAM es el proceso que utilizaremos en este trabajo. Respecto de las herramientas, emplearemos las versiones publicadas de la EPL, OpenUP/Basic y MMU-ISO (que hemos descripto extensamente en capítulos anteriores) con su metamodelo y lenguaje de modelado, SPEM 2.0 y UML, respectivamente. Para la implementación de los procesos resultantes de esta autoría y su distribución a un equipo de desarrollo nos valdremos del EPF Composer, herramienta de software desarrollada por la Fundación Eclipse para la autoría en la EPL (esta parte de la tarea constituye el objeto del siguiente capítulo).

Una vez establecido el entorno de autoría existen tres áreas de concern: la arquitectura del Framework de prácticas (el EPF), la autoría (o personalización) de una práctica y la autoría (o personalización) de configuraciones de prácticas.

La primera implica tareas tales como agregar un nuevo Slot de Producto de Trabajo, incorporar una nueva disciplina, insertar nuevos elementos en el Framework, etc. Las guías del MAM en este concern nos servirán de referencia en el próximo capítulo cuando mostremos las extensiones desarrolladas como plugins elaborados con el EPF Composer y su inserción en la arquitectura del EPF.

Para decidir cómo se verán involucradas en este trabajo de autoría las otras dos, MAM propone una guía de preguntas que conducen al escenario de trabajo que permite satisfacer el objetivo planteado. Reproducimos a continuación la parte de esta guía que resulta significativa para nuestro trabajo (la versión completa se puede consultar en Eclipse [2008d]).

- **¿Se quiere agregar un nuevo rol, tarea o producto de trabajo?**
  - **Sí. ¿Esos elementos reflejan una nueva técnica o aproximación al trabajo?**
    - **Sí. Crear una nueva práctica que incluya los nuevos elementos y procesos específicos que los articulen. Seguir *Hoja de Ruta: Autoría de nueva práctica*.**
  
- **¿Se quiere personalizar un proceso existente agregando elementos de una práctica disponible?**
  - **Sí. ¿La configuración por defecto del proceso ya incluye la práctica de la que se agregan elementos?**
    - **No. Personalizar los procesos transversales a varias prácticas usando los elementos que no están en la actual configuración. Seguir *Hoja de Ruta: Personalizar una configuración de prácticas existente*.**

La **Autoría de una nueva práctica** incluye las tareas de definir cómo se integra en el framework, cuáles son sus elementos y cómo se estructura, tanto como proveer descripciones para cada uno de los componentes incluidos. La sugerencia del MAM para la EPL es crear prácticas en forma iterativa e incremental, comenzando por un “boceto” que identifique los inputs y outputs y una organización general, pasando luego a una

estructuración más formal, para finalizar con el detalle de todos los componentes de la práctica. El cierre del proceso de creación de una práctica es su publicación dentro de EPL.

El otro escenario que nos interesa es el de la **Personalización de una configuración de prácticas existente**. Una configuración de prácticas consiste en un conjunto de prácticas vinculadas pero reconocibles individualmente que aborda una necesidad específica e incluye contenido de soporte para describir la forma en que las prácticas de ensamblan. OpenUP/Basic es una configuración que utiliza todas las prácticas de la EPL para presentar un proceso liviano y completo, como vimos antes.

En general, la customización de una configuración de prácticas involucra la creación de nuevas configuraciones que incluyan los elementos deseados de la original más las adaptaciones.

### 7.1.2. La contribución de esta Tesis: una Práctica y tres Configuraciones de método

Para cumplir con el objetivo de conseguir instanciaciones de procesos basados en OpenUP que puedan satisfacer los requerimientos del MMU-ISO, utilizando el MAM para la EPL como guía, abordamos los dos escenarios descritos en los párrafos anteriores mediante las siguientes tareas:

- Agregar nuevos elementos a la EPL. Todos los componentes específicos del desarrollo centrado en usuario identificados en el MMU-ISO deberán ser incorporados en la EPL.
- Agregar y modificar los procesos transversales a varias prácticas. En el Nivel 1 de capacidad del MMU-ISO, para garantizar la convergencia de la arquitectura evolutiva, el diseño evolutivo y el desarrollo centrado en el usuario. En los niveles 2 y 3, para incluir gestión de procesos y productos de DCU.
- El uso de un conjunto alternativo de asignaciones de roles, para pasar la responsabilidad de algunos procesos a los roles que requiere un proceso DCU.

Como resultado de estas tareas, nuestra propuesta consiste en el agregado a la EPL de la **Práctica: Desarrollo Centrado en el Usuario** (DesCU) y la personalización de la configuración OpenUP/Basic mediante su extensión para generar tres configuraciones nuevas.

La nueva **Práctica: Desarrollo Centrado en el Usuario**, contiene y articula los elementos de Contenido de Método que componen el núcleo duro y específico de un proceso DCU en línea con lo establecido por MMU-ISO.

Las tres configuraciones, que llamaremos **OpenUP/MMU-ISO Nx**, implementan los Niveles de Capacidad 1, 2 y 3 de MMU-ISO. El Nivel 1 se podrá alcanzar instanciando un proceso basado en OpenUP/Basic con el agregado de la Práctica DCU y la asignación correcta de los nuevos roles en tareas de otras prácticas. La capacidad en los siguientes niveles se logrará con extensiones de tareas de gestión y nuevas asignaciones, de las que se ofrecen Patrones de Capacidad en las configuraciones propuestas.

En lo que resta de esta Tesis describiremos estos nuevos elementos de método (resto del presente capítulo) y su implementación como plugins en el EPF (Capítulo 8).

### 7.1.3. Trabajos relacionados y antecedentes

Antes de describir en forma detallada el contenido de las contribuciones mencionadas, describimos aspectos pertinentes de algunos de los trabajos relacionados con la incorporación de elementos de diseño centrado en usuario en el Proceso Unificado (algunos fueron

presentados en el Capítulo 3 que pueden señalarse como antecedentes para la concepción de la Práctica DesCU y las configuraciones OpenUP/MMU-ISO.

### El plugin: OpenUP/DSDM

En el Capítulo 3 presentamos al framework DSDM como uno de los intentos desde la ingeniería de software de incluir actividades de DCU y usabilidad. Cuando la Fundación Eclipse liberó OpenUP como la versión open source del Proceso Unificado, el consorcio DSDM contribuyó con un plugin [DSDM-Consortium 2007] que extiende el rol de Stakeholder incluido en OpenUP, agregando cuatro roles de negocio y asignándolos a responsabilidades específicas para aspectos claves del proyecto. Los roles son:

- **Sponsor ejecutivo.** Representa a los grupos de interés cuyas necesidades deben satisfacerse con el proyecto. Este rol puede ser desempeñado por cualquiera que es afectado materialmente por el resultado del proyecto.
- **Visionario.** Tiene la visión del cambio de negocio que se necesita y es responsable para iniciar y conducir el proyecto a través de la entrega de los objetivos del negocio.
- **Usuario embajador.** Habitualmente proviene del área de negocios afectada y se convierte en una parte integral del equipo de desarrollo.
- **Usuario consultor.** Este rol tiene el conocimiento directo del trabajo diario que se automatiza con el sistema. Se trata de un potencial usuario final del sistema en desarrollo.

El Plugin OpenUP/DSDM no agrega ningún otro elemento nuevo de método. Los nuevos roles son asignados a las Tareas y Productos existentes. La intención es promover un abordaje explícito de colaboración entre las comunidades de negocios y técnica, ambas responsables del desarrollo de software. Se sostiene que “es la experiencia de los practicantes de DSDM y la comunidad Agile en general la que mejora significativamente el potencial para un proyecto exitoso” [DSDM-Consortium 2007].

### El plugin: Icon Web enabled UCD (WEUCD)

Consiste en una mejora al Proceso Unificado para el desarrollo centrado en usuario de proyectos de sitios web. Ha sido liberada como plugin para el EPF para poder incorporarlo en una configuración de OpenUP [ATG 2010], aunque también puede ser utilizado con el Rational Method Composer de IBM.

Basado en el ciclo de vida del Proceso Unificado, con foco en las disciplinas Requerimientos, Análisis y Diseño, Implementación y Test, agrega varios elementos al método: 11 nuevos roles, 12 nuevos productos de trabajo, 6 nuevas Actividades que agrupan a tareas nuevas y existentes en UP. La figura 7.1 muestra el proceso de despliegue básico del plugin WEUCD.

Dado el foco en el desarrollo de aplicaciones web, los roles agregados se vinculan con el manejo de contenidos (Content Editor, Content Manager y Content Organizer) y la creación conceptual del sitio (Creative Concept Director, Information architect). Pero también se agregan otros propios de un proceso que tiene en cuenta la usabilidad y la interacción de usuarios, como el diseño de pantallas y de navegación (Graphic Designer, User Interface Prototyper, Interaction designer), las evaluaciones de usabilidad (Usability Evaluator, Usability Specialist) y la investigación de mercado (User Researcher).

Los productos que se incorporan y las tareas que los manejan pueden agruparse en manejo de conceptos y especificaciones (Content Inventory, Information Architecture, User Experience Concept, Interaction Specification, User Research Report, Writing Style

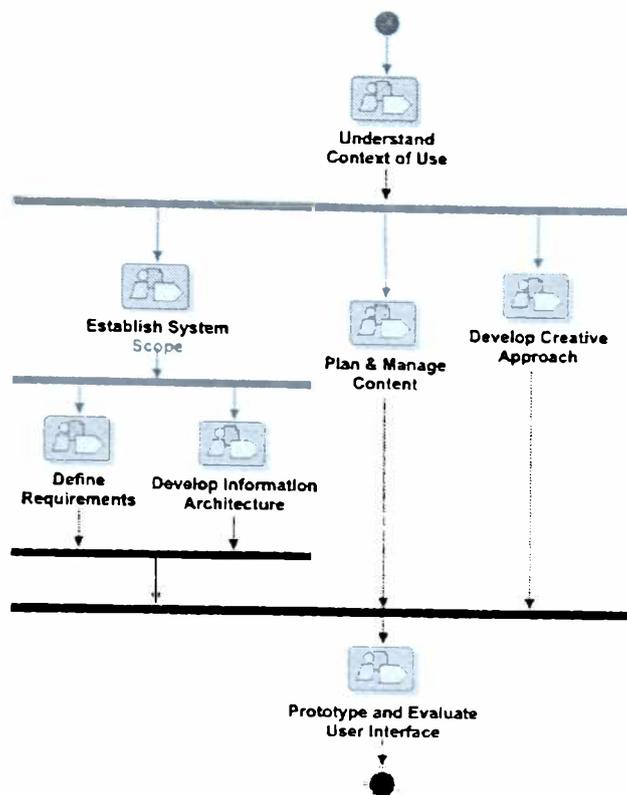


Figura 7.1: Patrón de entrega de plugin WEUCD, de [ATG 2010]

Guide), prototipado y diseño de interfaces y navegación (Site Map, User Interface Prototype, Wireframes) y evaluaciones de usabilidad (Usability Benchmarks, Usability Test Plan, Usability Test Results).

Aunque el proceso de despliegue presenta el ciclo completo de requerimientos, diseño, construcción y prueba desde el punto de vista de usuario, el plugin no muestra ni sugiere la forma de incorporar las actividades agregadas en el ciclo de vida del Proceso Unificado.

### El plugin de la experiencia de usuario para RUP

En el Capítulo 3 hablamos de RUP y la extensión publicada por IBM como Plugin de la Experiencia del Usuario [IBM 2005b]. El plugin presenta el abordaje de la UX en tres áreas:

- En la arquitectura, agrega al Documento de Arquitectura una vista de la experiencia de usuario. Esta vista es opcional y contiene los elementos del Modelo de la UX que tienen impacto en la arquitectura del sistema.
- En la capa de presentación, incorpora todos los elementos de método y proceso vinculados con el diseño y desarrollo de la interfaz de usuario. Toma preeminencia en esta vista el Modelo de la UX y, en consecuencia, los roles encargados de su realización (el Diseñador de Interfaz de Usuario) y las tareas de las que es output (Diseñar la Interfaz de Usuario) o para las que es input (Analizar los Casos de Uso, Prototipar la IU, Revisar el diseño, Identificar elementos de diseño). El plugin UX-RUP establece que el Modelo de UX se represente como un modelo UML con el estereotipo «user-experience model».
- En la capa de negocio, se intenta que exista consistencia entre el Modelo de la UX y los elementos del método que soportan la representación de los aspectos del negocio.

## La disciplina Usabilidad de Göransson

Bengt Göransson ha propuesto la extensión de RUP con la integración de una disciplina de usabilidad [Göransson et al. 2003]. El trabajo identifica obstáculos en RUP para la aplicación de diseño centrado en usuario (la centralidad de la arquitectura, el predominio de los casos de uso, la concentración de actividades de usabilidad sólo en la disciplina de Requerimientos y en la fase de Elaboración). Para sortear estos obstáculos propone crear una nueva disciplina, el Diseño de Usabilidad que, si bien tiene un mayor nivel de actividad en las fases de Inicio y Elaboración, aporta al ciclo de vida en todas las fases.

La disciplina de Diseño de Usabilidad introduce cinco nuevos roles y un flujo de tareas con diez actividades.

Los roles introducidos por DU son:

- Diseñador de usabilidad, como el campeón y líder de usabilidad
- Especialista de estudios de campo, que planifica y realiza estudios de usuarios y analiza los resultados.
- Diseñador de interacción, que se ocupa de la creación y refinamiento de todo el esquema de interacción de usuario.
- Diseñador gráfico, responsable de dar forma visual a la interfaz de usuario.
- Especialista en evaluaciones de usabilidad, responsable de planificar, preparar, ejecutar y reportar las pruebas de usabilidad.

La figura 7.2 muestra el flujo básico de actividades para la disciplina Diseño de Usabilidad. Se puede advertir que el flujo incorpora tres actividades propias sólo de las fases Inicio y Elaboración tales como Crear/Modificar el Plan de Diseño de Usabilidad, Realizar análisis de competidores y Conducir estudios de usuario. El resto de las actividades, que se describen brevemente a continuación se pueden efectuar en cualquiera de las fases del ciclo de vida del UP. Lamentablemente, en las publicaciones de Göransson y sus colegas, no indican referencias exhaustivas sobre la integración de este flujo en los Patrones de Entrega del Proceso Unificado en general.

- Planificar el Diseño de Usabilidad, al comienzo del proyecto y refinarlo antes de cada iteración.
- Realizar estudios de usuario, para comprender a los potenciales usuarios, sus necesidades y el contexto de uso.
- Estudiar productos competidores, para comprender el estado del arte en el dominio del problema.
- Diseño conceptual, se plantea como la estructura general de la interfaz de usuario, apoyado por escenarios.
- Diseño de interacción, que consiste en desarrollar e ilustrar la forma en que el usuario realizará la interacción con el sistema, navegación, uso de funcionalidad, etc.
- Diseño detallado, con el refinamiento en detalle de las partes individuales de la interfaz de usuario.
- Desarrollar asistencia de usuario, focaliza en desarrollar material y actividades para dar soporte al usuario en el uso del sistema.
- Monitorear el trabajo de usabilidad, que focaliza en mantener el control sobre las decisiones de usabilidad que se tomen <sup>a</sup>al vuelo.<sup>en</sup> la fase de Construcción, cuando habitualmente ya todas las decisiones de diseño se han tomado.

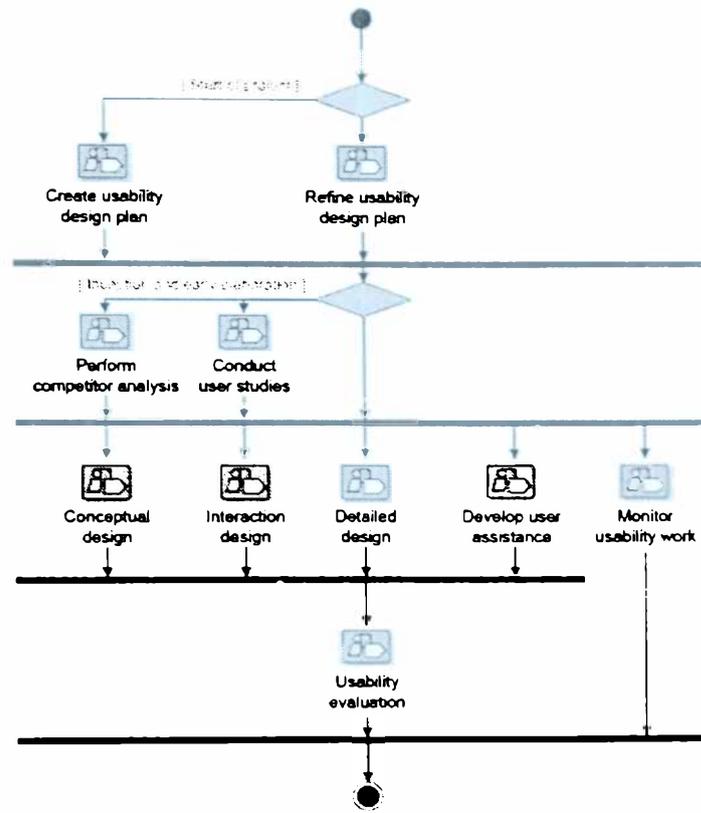


Figura 7.2: Flujo de actividades de Diseño de Usabilidad. según [Goransson et al. 2003]

- Realizar evaluaciones de usabilidad, que consiste en probar el sistema (en su etapa de prototipos iniciales o en sus versiones para liberar) contra los objetivos y metas de usabilidad planteadas en el plan inicial.

## 7.2. Práctica: DesCU - Desarrollo Centrado en el Usuario

La práctica que proponemos incorporar a la EPL articula los elementos metodológicos necesarios para constituir un proceso de desarrollo de sistemas basados en software que involucre activamente a todas las partes afectadas por el sistema final, particularmente a los usuarios.

A continuación listamos los elementos de contenido de método que son articulados por la práctica que llamamos **DesCU o Desarrollo Centrado en el Usuario**. Este listado sólo pretende ser ilustrativo de la diversidad de aspectos que se ponen en juego y la magnitud de esfuerzo que puede demandar la implementación de la práctica. Como se verá más adelante en las configuraciones OpenUP/MMU-ISO cada proyecto podrá requerir la implementación completa o parcial de la Práctica o incluso su extensión.

### ▪ Roles

- Stakeholder: Patrocinador
- Stakeholder: Líder técnico del dominio
- Stakeholder: Representante de Usuarios
- Stakeholder: Usuario final
- Analista: Especialista en Experiencia de Usuario
- Desarrollador: Diseñador de Experiencia de Usuario

-  Tester: Tester de Experiencia de Usuario
- Productos de trabajo
  -  Modelo de usuario
  -  Modelo de tareas
  -  Meta de usabilidad
  -  Concepto de Experiencia de Usuario
  -  Prototipo de Experiencia de Usuario
  -  Storyboard de Experiencia de Usuario
  -  Mapa de navegación
  -  Documento de usuario y entrenamiento
- Tareas
  -  Adaptar el proceso de Desarrollo Centrado en Usuario
  -  Establecer objetivos de la Experiencia de Usuario
  -  Entender y especificar el contexto de uso
  -  Diseñar la Experiencia de Usuario
  -  Diseñar componentes de interacción
  -  Revisar el diseño para la experiencia de usuario
  -  Preparar pruebas de usabilidad
  -  Ejecutar pruebas de usabilidad
  -  Producir material de entrenamiento y soporte al usuario
  -  Proveer entrenamiento y soporte al usuario
- Orientaciones
  -  Definiciones de procesos DCU
  -  Modelo ISO de Madurez en Usabilidad

### 7.2.1. Roles

OpenUP/Basic, como configuración de las prácticas base de la EPL ofrece tres definiciones genéricas que podrían ser asignadas a procesos DCU: Stakeholder, Analista y Desarrollador.

En la presentación del MMU-ISO (Capítulo 4) y en el Assessment de OpenUP (Capítulo 6) vimos con claridad que las diferentes prácticas que componen los procesos DCU requieren de un conjunto de habilidades específicas, en el caso de los miembros técnicos del equipo, y de conocimientos del dominio y capacidades de decisión apropiadas, en lo que hace a los Stakeholders involucrados.

Por esta razón, consideramos que las definiciones genéricas de roles no siempre aseguran que luego se puedan identificar claramente los perfiles necesarios y, sobre todo, encontrar profesionales que posean el suficiente conjunto de habilidades necesarias para satisfacer los requerimientos del proceso.

Hemos visto que algunas extensiones de UP y OpenUP ya han identificado esta situación proponiendo diversidad de roles a incluir para asegurar la consideración de la experiencia de usuario en el desarrollo. OpenUP/DDSMM propone la inclusión de cuatro roles nuevos, dos con responsabilidades de tareas y otros dos de consulta y apoyo [DSDM-Consortium 2007]. El plugin denominado “Web-enabled UCD” (DCU para la web) [ATG 2010] propone once nuevos roles, sin embargo extrañamente no hay ningún rol nuevo de usuarios o clientes en este plugin. El plugin de Experiencia de Usuario para RUP [IBM 2005b] propone la inclusión de un Diseñador de Interfaz de Usuario diferenciado de los diseñadores de software.

Para nuestra propuesta, tomamos los roles habitualmente definidos en la literatura sobre Usabilidad y Diseño de interacción además de los casos mencionados antes, pero priorizamos el compromiso por mantener OpenUP como un conjunto de prácticas que permitan configurar procesos ágiles y de bajo nivel de burocracia y ceremonia. Hemos identificado la necesidad de contar con seis roles que no están incluidos en OpenUP: cuatro son *Actores de Negocio* (pueden definirse como extensiones de Stakeholder), los otros dos corresponden a extensiones de roles de *Analista* y *Desarrollador*.

- Actor de negocio
  - Stakeholder: Patrocinador
  - Stakeholder: Líder técnico del dominio
  - Stakeholder: Representante de usuarios
  - Stakeholder: Usuario final
- Analista
  - Analista: Especialista en Experiencia de usuario
- Desarrollador
  - Desarrollador: Diseñador de la Experiencia de usuario
- Tester
  - Tester: Tester de la Experiencia de usuario

Con estos nuevos roles buscamos asegurar por un lado que se identifica claramente el tipo de participación que los stakeholders deben tener en un proceso DCU y por otro que existe una adecuada disponibilidad de las habilidades y conocimientos necesarios para la viabilidad y factibilidad de las prácticas básicas que prescribe MMU-ISO.

A continuación presentamos en conjunto las Definiciones de los Roles y sus Asignaciones en el marco de la Práctica DesCU<sup>1</sup>.

### Stakeholder: Patrocinador

Este rol es el representante del grupo de interesados en que el desarrollo se lleve adelante con éxito. Es quien impulsa su realización internamente en la organización afectada.

Corresponde en parte con el rol denominado Sponsor en OpenUP/DSDM [DSDM-Consortium 2007], aunque en la propuesta OpenUP/MMU-ISO este rol opera en conjunto con el Gerente del Proyecto. Debería formar parte de un grupo o comité de avance que vele por el buen desarrollo del proyecto, adoptando todas las decisiones necesarias sobre recursos y riesgos.

<sup>1</sup>Dado que se trata de roles específicos para la Práctica DesCU, haremos uso de la opción que brinda EPL para evitar LA *asignación diferida*. La asignación diferida es el mecanismo que propone EPL para facilitar el reuso de roles en otras prácticas (ver Concept: Delayed assignment en [Eclipse 2008b])

Este rol será desempeñado por alguien con cargo alto en el negocio, ya que sus funciones de cara al proyecto muchas veces consisten en resolver temas que implican “abrir puertas” y tomar decisiones financieras. Es un rol crucial para asegurar y permitir el avance del proyecto sin interrupciones, evitando los obstáculos que suelen imponer las burocracias o niveles políticos.



Figura 7.3: Asignaciones del Rol Patrocinador

Las habilidades de este rol comprenden la capacidad de comprometer fondos y recursos apropiados, tener pensamiento crítico, capacidad de decisión, conciencia de la situación política y conocimiento de negocio.

A lo largo de todo el proyecto, las responsabilidades del Patrocinador consisten en:

- Monitorear la governancia del proyecto
- Responder a los temas que puedan exceder en las capacidades de decisión del equipo del proyecto
- Revisar y aprobar la Lista de Riesgos
- Revisar y acordar el Plan de Proyecto
- Comprometer los recursos necesarios para la siguiente fase del proyecto.

Además, debe hacerse cargo de las siguientes responsabilidades en algunos momentos específicos del proceso:

- **En Fase de Inicio:** Revisar y aceptar el Plan de Proyecto o detener el proyecto si no hay una solución viable
- **En Fase de Elaboración:** Aprobar el Caso de Negocio para el proyecto se extienda más allá de la fase de Elaboración para obtener los recursos necesarios.
- **En Fase de Transición:** Aprobar el sistema entregado y Determinar si es necesario un nuevo incremento del sistema o se detiene el proyecto.

Para cumplir estas responsabilidades, en lo referido a la Experiencia del Usuario participa en forma adicional de la Tarea Adaptar el proceso de Desarrollo Centrado en Usuario.

#### Stakeholder: Líder técnico de dominio

El Líder técnico de dominio es la persona responsable de conseguir que el proyecto se inicie focalizado en las metas del negocio y se mantenga alineado con ellas. Permanece involucrado a lo largo de todo el proceso de diseño y desarrollo para asegurar que se alcanzan los objetivos originales. Se convierte en el nexo con los altos niveles de dirección cuando surgen aspectos de negocio que requieren de decisiones sobre metas del proyecto, recursos financieros, etc.

Debe asegurar que el proyecto transcurra en senderos de excelencia desde el punto de vista del negocio (el Gerente del Proyecto hará lo propio desde el punto de vista de la ingeniería del software). Su vinculación con el Patrocinador en estos aspectos es de vital importancia.

Sus principales responsabilidades son:

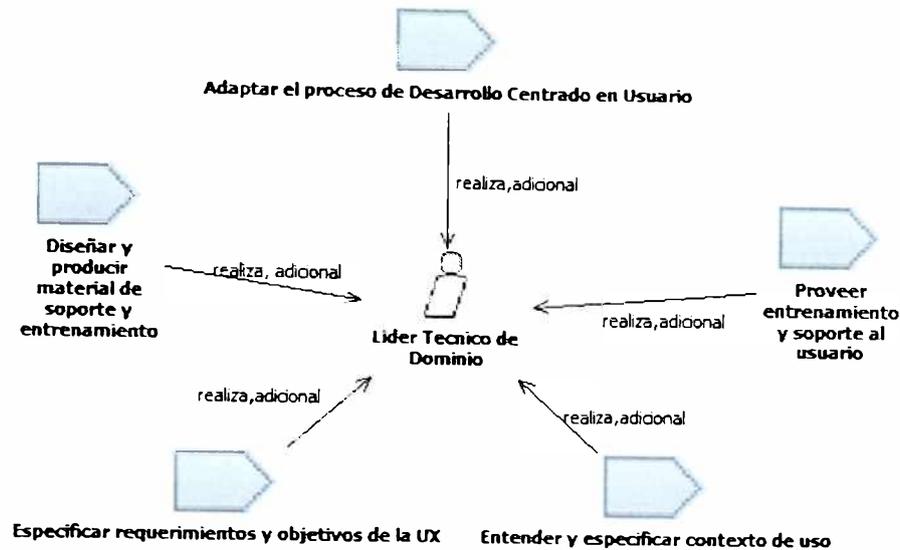


Figura 7.4: Asignaciones del Rol Líder Técnico de Dominio

- Facilitar la implementación práctica y operativa del contenido de la Visión
- Aportar un punto de vista comprensivo sobre el proceso de negocio completo
- Participar y contribuir a las sesiones claves de búsqueda de requerimientos
- Participar y contribuir en las sesiones claves de diseño
- Resolver conflictos entre las áreas de negocio
- Asegurar la disponibilidad de recursos de usuario
- Monitorear el progreso en relación con la Visión original
- Aportar compromiso y disponibilidad durante todo el ciclo de desarrollo

Para cumplir estas responsabilidades, básicamente participa en forma adicional de las siguientes Tareas:

- Adaptar el proceso de Desarrollo Centrado en Usuario
- Entender y especificar el contexto de uso
- Establecer objetivos de la experiencia del usuario
- Diseñar y producir material de entrenamiento y soporte al usuario
- Guiar y dar soporte al usuario

### Stakeholder: Representante de Usuarios

El Representante de usuarios es parte fundamental del equipo de trabajo, conoce el dominio y la tarea diaria, compartiendo con el equipo ideas y generando requerimientos al tiempo que mantiene a otros usuarios (fundamentalmente usuarios finales) informados del estado de situación.

Quien desempeñe este rol debe tener además de interés y deseos de participar, algún nivel de autoridad, responsabilidad y conocimiento que asegure que los requerimientos e ideas obtenidas sobre el dominio resultan correctas. Por ejemplo, en los casos de sistemas

de gestión este rol lo desempeñan Jefes de Sector o de Area Usuaria. Aunque no se trata necesariamente de que sea alguien con poder formal, sino con habilitación durante el proyecto.

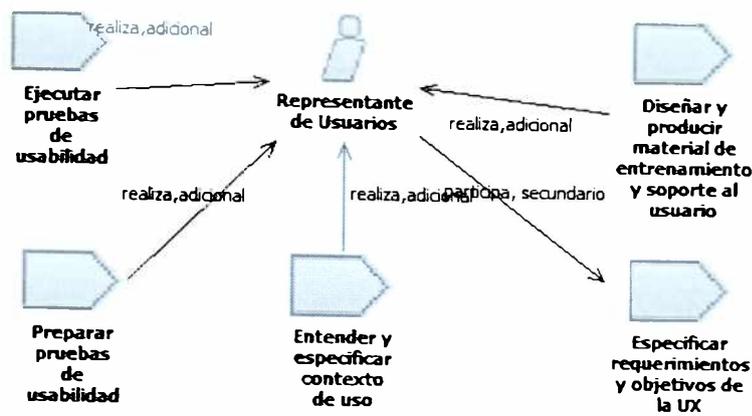


Figura 7.5: Asignaciones del Rol Representante de Usuarios

Es importante que la persona asignada al rol tenga el tiempo y los recursos necesarios para participar en el proyecto activamente. Entre sus responsabilidades se pueden mencionar:

- Proveer input de requerimientos de negocio y decisiones de diseño
- Comunicar con otros usuarios y obtener su acuerdo
- Proveer input en las sesiones de prototipado
- Revisar documentación
- Revisar y aceptar el software entregado
- Proveer documentación de usuario
- Asegurar entrenamiento adecuado
- Organizar y controla el test de usuario

Para cubrir estas responsabilidades, debe participar en forma primaria o adicional de:

- Entender y especificar el contexto de uso
- Especificar metas y requerimientos de UX
- Diseñar y producir material de soporte
- Preparar pruebas de usabilidad, Ejecutar pruebas de usabilidad

#### Stakeholder: Usuario final

Es el que operará diariamente con el sistema a construir. Posee un conocimiento esencialmente práctico del tema o área de negocio y habilidad para comunicarlo.

Habitualmente no está integrado en los equipos de proyecto, sino que participa en algunos procesos de revisión y pruebas de usabilidad. Sin embargo es imprescindible su participación en el ciclo de vida del proyecto, ya que es la persona que se verá directamente

afectada en su quehacer cotidiano con el sistema y quien puede aportar detalles de la tarea que muchas veces quedan escondidos o relegados en relevamientos de requerimientos.

Todas las tareas que requieran la voz del usuario final (elaboración de modelos mentales de usuario, pruebas de usabilidad, aceptación de usuarios, etc) deberían ser configuradas con la participación de este rol.

Su nexa natural con los miembros técnicos del equipo de proyecto es el Representante de usuarios.

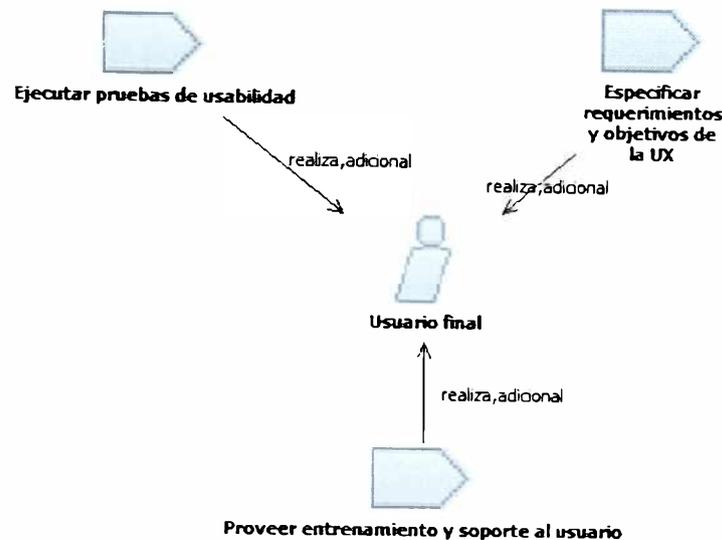


Figura 7.6: Rol de Usuario final

Sus responsabilidades consisten en:

- Proveer el conocimiento de la tarea diaria involucrada en el sistema.
- Participar en las actividades de prototipado y revisión, especialmente en lo referido a los aspectos prácticos de las tareas que el sistema intenta automatizar.
- Aprobar los diseños prototipados desde el punto de vista de su uso práctico.
- Colaborar en las pruebas de usabilidad y aceptación.

Para cubrir estas responsabilidades, debe participar como partícipe secundario de:

- Especificar requerimientos de UX
- Ejecutar pruebas de usabilidad
- Proveer entrenamiento y soporte al usuario

### Analista: Especialista en UX

Este Analista participa activamente como principal responsable de la extracción, definición y especificación de requerimientos de interacción y experiencia del usuario. Adicionalmente colabora en las mismas actividades que el resto de los analistas.

Este rol debe funcionar como el “campeón” y responsable general de la UX. En algunos contextos de desarrollo, por ejemplo para las aplicaciones web, este rol se asocia con el arquitecto de la información. Por una parte, realiza tareas de Analista para identificar y especificar requerimientos para la UX del sistema, por otra, colabora en el planteo del

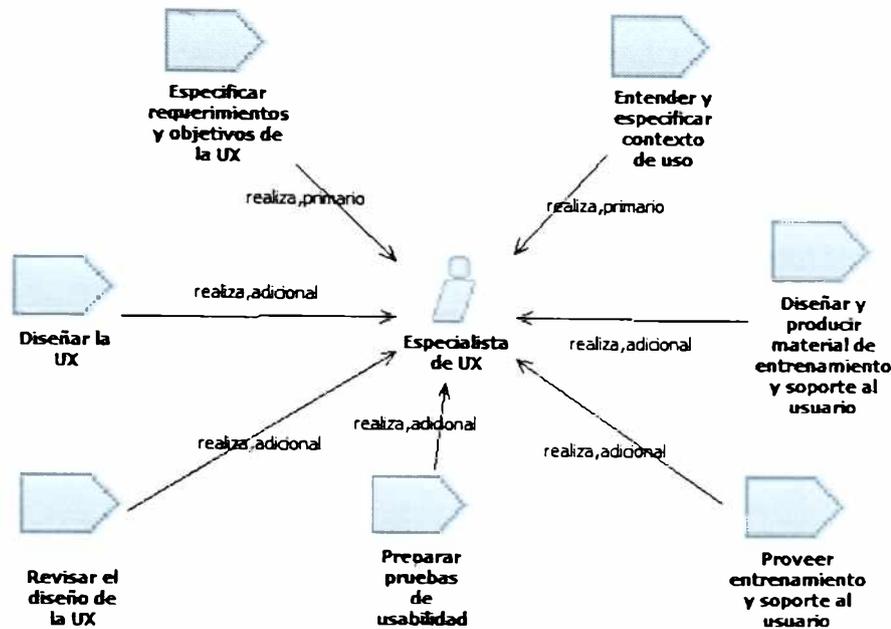


Figura 7.7: Rol de Especialista en Experiencia de Usuario

diseño de la interacción. En cualquier caso, debe ser el encargado de planificar y gestionar las actividades relacionadas con la UX y ejecutar algunas.

Es de esperar que los recursos cumpliendo este rol dispongan de un buen background tanto en HCI y procesos DCU como en los recursos tecnológicos adecuados. La experiencia en la utilización de diferentes métodos y técnicas de usabilidad les permitirá categorizar usuarios y entender sus necesidades. Debe ser capaz también de planificar, realizar y analizar estudios de usuarios.

Será responsable de:

- Elicitar y especificar los requerimientos de UX
- Categorizar usuarios, entender sus necesidades y especificar el contexto de uso
- Proponer y colaborar en la definición de objetivos de UX

Para cubrir sus responsabilidades, el Especialista de UX participará de las siguientes tareas:

- Establecer objetivos de la UX
- Entender y especificar el Contexto de Uso
- Diseñar la UX
- Revisar el diseño de la UX
- Preparar pruebas de usabilidad
- Diseñar y producir material de soporte y entrenamiento al usuario

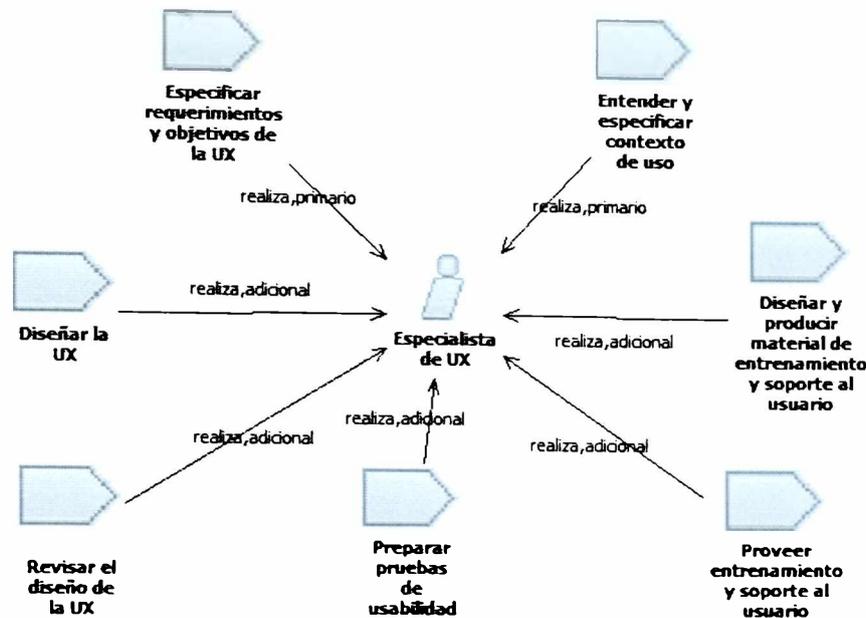


Figura 7.7: Rol de Especialista en Experiencia de Usuario

diseño de la interacción. En cualquier caso, debe ser el encargado de planificar y gestionar las actividades relacionadas con la UX y ejecutar algunas.

Es de esperar que los recursos cumpliendo este rol dispongan de un buen background tanto en HCI y procesos DCU como en los recursos tecnológicos adecuados. La experiencia en la utilización de diferentes métodos y técnicas de usabilidad les permitirá categorizar usuarios y entender sus necesidades. Debe ser capaz también de planificar, realizar y analizar estudios de usuarios.

Será responsable de:

- Elicitar y especificar los requerimientos de UX
- Categorizar usuarios, entender sus necesidades y especificar el contexto de uso
- Proponer y colaborar en la definición de objetivos de UX

Para cubrir sus responsabilidades, el Especialista de UX participará de las siguientes tareas:

- Establecer objetivos de la UX
- Entender y especificar el Contexto de Uso
- Diseñar la UX
- Revisar el diseño de la UX
- Preparar pruebas de usabilidad
- Diseñar y producir material de soporte y entrenamiento al usuario

## Desarrollador: Diseñador de UX

Es necesario incorporar una especialización del Rol Desarrollador: el Diseñador de Experiencia de Usuario, que se encargará de plasmar los requerimientos de UX en el sistema en desarrollo.

Este rol no es responsable de la implementación de la IU, sino que su trabajo se orienta hacia el diseño y organización visual de la IU. Debe participar activamente en la captura de requerimientos que impacten en la IU, construir los prototipos necesarios tanto para la elicitación de requerimientos como para la presentación de ideas de diseño, involucrar a los roles Representante de usuario, Usuario final en las revisiones de usabilidad y pruebas de aceptación, revisar las implementaciones llevadas adelante por los desarrolladores.

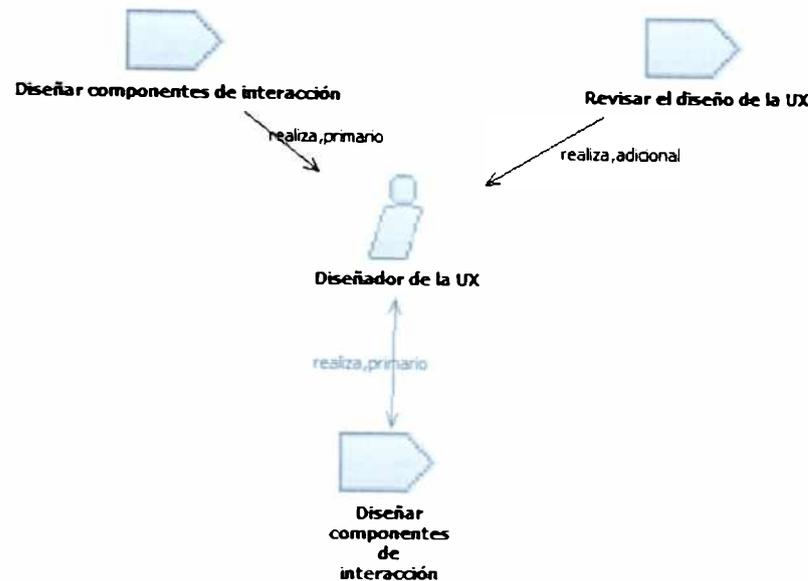


Figura 7.8: Asignaciones del Rol Diseñador de Experiencia de Usuario

Habitualmente, este rol es desempeñado por profesionales creativos con background en artes visuales, más que en temas de negocio o tecnología. Entre las razones que justifican la asignación de este rol (y no asignar el trabajo de IU simplemente a los desarrolladores más afines a aspectos visuales) se encuentra la necesidad de disponer de un conjunto de habilidades orientadas a la usabilidad que puedan ser incrementadas y optimizadas en el proyecto, lo que obliga entre otras cosas a que quien/es estén a cargo de la UX se vean más influidos por consideraciones vinculadas a la experiencia del usuario que a los aspectos de implementación.

El rol Diseñador de la UX será responsable de:

- Realizar propuestas de diseño de todos los elementos del sistema que proveen la experiencia del usuario: mapa de navegación, interfaces de usuario, etc. asegurando que satisfacen los requerimientos de calidad de uso especificados.
- Realizar los prototipos de elementos de UX para facilitar las pruebas de usabilidad y aceptación con usuarios

Participa en, participa activamente de:

- Diseñar la experiencia del usuario
- Revisar el diseño de la experiencia del usuario
- Diseñar componentes de interacción

### Tester: Tester de Experiencia de Usuario (TUX)

El Tester de UX es el responsable de evaluar la usabilidad del sistema, sea mediante pruebas con usuarios o mediante inspecciones. Las pruebas de usabilidad se basan en la exposición del sistema ante muestras representativas de los usuarios. Por otra parte, un experto en usabilidad puede efectuar evaluaciones heurísticas sobre la base de criterios aceptados en la comunidad de usabilidad, sin contar necesariamente con inputs de usuarios.

La tarea del Tester de UX consistirá por tanto en determinar los métodos de evaluación apropiados, diseñar las pruebas, reclutar participantes, conducir las evaluaciones, analizar los datos y presentar los reportes que muestran los resultados y recomendaciones.

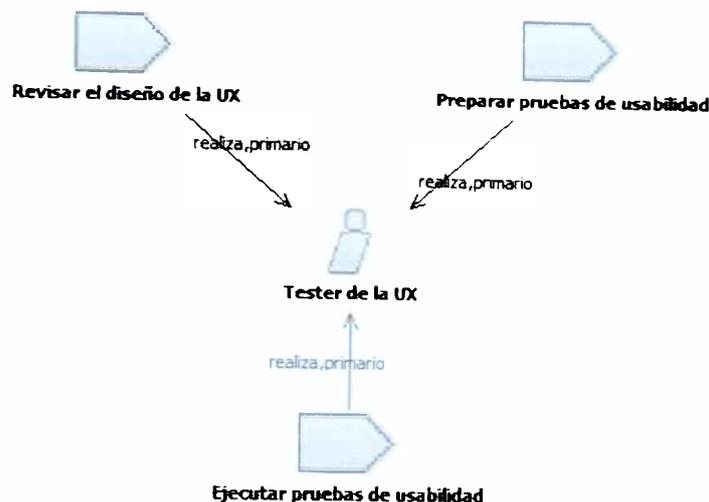


Figura 7.9: Asignaciones del Rol Tester de Experiencia de Usuario

La persona que cumpla el rol de Tester de UX debe estar entrenado en los métodos para realizar pruebas de usabilidad e inspecciones. Debería poseer habilidades en la evaluación de software sobre la base de criterios de usabilidad y en el diseño, ejecución y evaluación de pruebas empíricas.

Es probable que la misma persona pueda cubrir los roles de Tester de UX y Especialista en UX, pero es importante notar que el enfoque de la tarea es diferente. En el caso del Tester buscará encontrar los defectos de usabilidad del software ya construido (aunque sea en estado de prototipo), mientras el Especialista deberá encontrar y especificar los requerimientos de la aplicación a ser construida.

Será responsable primario de las tareas Revisar el diseño de la UX, Preparar pruebas de usabilidad y Ejecutar pruebas de usabilidad.

#### 7.2.2. Productos de trabajo

Con el Assessment de OpenUP descrito en el Capítulo 6 identificamos las carencias de este método, entre otras cosas, en términos de Productos de Trabajo según el MMU-ISO. Para cubrir esa carencia sólo será necesario incluir los artefactos mencionados en el MMU-ISO que no figuran en OpenUP (una lista de estos puede verse en la Tabla 7.1). Sin embargo, hemos planteado la conveniencia de mantener el bajo nivel de burocracia y ceremonial que propugna OpenUP, lo que nos obliga a incorporar sólo aquellos productos que no pueden ser cubiertos por los ya existentes en la configuración básica del método, ya sea mediante extensiones o adaptaciones ad-hoc.

Procesos DCU	PT de MMU-ISO	PT a incluir en OpenUP	PT de OpenUP a extender
<b>DCU.1 Asegurar el contenido DCU en la estrategia de sistemas</b>			
DCU.1.1 Representar los stakeholders.	Análisis de mercado - Análisis de tendencias	Usability benchmarking	
DCU.1.2 Analizar el mercado		Plan de UX	
DCU.1.3 Definir y planificar la estrategia de sistemas			
DCU.1.4 Reunir respuesta de mercado			
DCU.1.5 Analizar tendencias de usuarios			
<b>DCU.2 Planificar y Gestionar el proceso DCU</b>			
DCU.2.1 Consultar a los stakeholders	Planes de integración HCI en otras actividades de desarrollo - Planes de comunicación entre desarrollo y hci - Hitos de diseño y desarrollo de hci - Política de hci - Reportes de auditoría		Plan de proyecto
DCU.2.2 Planificar la participación de usuarios			Plan de iteración
DCU.2.3 Seleccionar métodos y técnicas centradas en usuarios			
DCU.2.4 Asegurar el enfoque centrado en usuario en el equipo			
DCU.2.5 Planificar el proceso de diseño centrado en usuario			
DCU.2.6 Gestionar el proceso centrado en usuario			
DCU.2.7 Abogar por las actividades centradas en usuario			
DCU.2.8 Proveer soporte para el diseño centrado en usuario			
<b>DCU.3 Especificar los requerimientos de usuario y de organización</b>			
DCU.3.1 Clarificar y documentar las metas del sistema	Rango y relevancia de usuarios - Especificación detallada de reqs de usuario y stakeholder - Especificación de reqs de organización - Metas de usabilidad específicas y medibles - Lista de legislación y normas aplicables - Fuentes de reqs	Requerimiento de usuario	Visión
DCU.3.2 Definir los stakeholders		Modelo de análisis	Modelo casos de uso
DCU.3.3 Evaluar los riesgos para stakeholders			
DCU.3.4 Definir el sistema			
DCU.3.5 Generar requerimientos de stakeholders y de la organización		usability benchmarking	
DCU.3.6 Establecer la calidad en objetivos de uso			
<b>DCU.4 Entender y especificar el contexto de uso</b>			
DCU.4.1 Identificar y documentar tareas de usuario	Especificación de rango de usuarios objetivo, de tareas y entorno - Información de usuario y de stakeholder - Información de tareas - Análisis de organización -	Modelo de tareas	Modelo casos de uso
DCU.4.2 Identificar y documentar atributos significativos de usuario		Modelo de análisis	
DCU.4.3 Identificar y documentar entorno organizacional		Reporte de investigación de usuarios	Requerimiento de apoyo de usuarios
DCU.4.4 Identificar y documentar el entorno técnico			
DCU.4.5 Identificar y documentar entorno físico			
<b>DCU.5 Producir soluciones de diseño</b>			
DCU.5.1 Alocar funciones	Especificación de interacción de usuario - Detalle de diálogos- Look and feel - Layout y detalles de UI - Modelos de tareas - Prototipos - Evidencia de revisión de acuerdo con resultados de evaluaciones - Lista de normas usadas y cómo se aplicaron - Rational	Prototipo	Cuaderno arquitectura
DCU.5.2 Producir modelo de tareas compuesto		Storyboard	
DCU.5.3 Explorar el diseño del sistema		Mapa de navegación	
DCU.5.4 Desarrollar soluciones de diseño			
DCU.5.5 Especificar el sistema			
DCU.5.6 Desarrollar prototipos		Documentación de usuario y mantenimiento	
DCU.5.7 Desarrollar entrenamiento de usuarios		Documentación de usuario y mantenimiento	
DCU.5.8 Desarrollar soporte de usuarios			
<b>DCU.6 Evaluar el diseño contra los requerimientos</b>			
DCU.6.1 Especificar y validar el contexto de evaluación	Contexto de evaluación - Descripción de sistema testeado y estado - Elección de métodos y datos para testing - Defectos de usabilidad y ergonomía - Registro audiovisual de pruebas - Logs de observación de usuario -	Prototipo	
DCU.6.2 Evaluar prototipos tempranos para definir los requerimientos de diseño		Storyboard	
DCU.6.3 Evaluar prototipos para mejorar el diseño		Mapa de navegación	
DCU.6.4 Evaluar el sistema para chequear que se satisfacen los requerimientos		Test de UX	
DCU.6.5 Evaluar el sistema para chequear que se siguieron las prácticas requeridas		Plan de test de UX	
DCU.6.6 Evaluar el sistema en uso			
<b>DCU.7 Introducir y operar el sistema</b>			
DCU.7.1 Gestionar el cambio	Plan de implantación - Estructura afectada de la organización - Instructivos de trabajo - Plan de entrenamiento - Material de entrenamiento - Monitoreo de uso - Reportes de impacto - Recomendaciones de mejoras para futuros desarrollos	Test de aceptación	
DCU.7.2 Determinar el impacto sobre la organización y los stakeholders		Documentación de usuario y mantenimiento	
DCU.7.3 Customización y diseño local			
DCU.7.4 Proveer entrenamiento a usuarios			
DCU.7.5 Dar soporte a usuarios en las actividades planificadas			
DCU.7.6 Asegurar la conformidad del lugar de trabajo a la legislación de ergonomía			

Tabla 7.1: Productos de Trabajo a incorporar

El primer elemento que debe ser incluido explícitamente es central en cualquier proceso iterativo, particularmente uno centrado en usuarios: el *Prototipo de Experiencia de Usuario* en cualquiera de sus versiones, que facilita la intervención activa de los usuarios en el diseño y la evaluación del sistema [Sharp et al. 2007],[Gulliksen et al. 2003]. También incluimos otros dos artefactos que tendrán una fuerte relación con los prototipos: el *Storyboard de UX* y el *Mapa de Navegación*. El Storyboard permitirá analizar la versión dinámica de una parte del prototipo de UX del sistema, mientras que el Mapa de navegación dejará claramente estable la relación entre todos los Storyboards. En esta parte de la selección de productos de trabajo puede reconocerse una influencia importante del Plugin RUP para la UX. Sin embargo, vale destacar que haremos foco en el uso de estas herramientas como lugar compartido de elaboración entre los roles de analista, desarrollador y las diferentes versiones de usuarios.

La lista completa de las nuevas definiciones de productos de trabajo que incluimos en la Práctica DesCU es la siguiente:

- Modelo de usuario
- Meta de usabilidad
- Modelo de Tareas
- Concepto de la Experiencia de Usuario
- Prototipo de Experiencia de Usuario
- Storyboard de Experiencia de usuario
- Mapa de navegación
- Documento de usuario y entrenamiento

A continuación mostramos las definiciones de estos productos utilizando una estructura basada en la sugerida en la Orientación: Detallando una Tarea de la versión 1.5.0.1 de OpenUP [Eclipse 2008c]:

- Propósito: ¿porqué hay que incluir este producto? ¿Cuál es su meta en el proceso?
- Descripción y relaciones: ¿en qué consiste?, ¿cuáles roles se vinculan con él (quién es responsable de realizarlo, quién puede modificarlo, quienes colaboran), de qué tareas es input y de cuáles es output?
- Ejemplos: incluir muestras completas o parciales que ejemplifiquen el producto.
- Personalización posible: detallar las posibilidades de adaptar el proceso para tener o no el Producto y cómo adaptarlo al contexto específico. Enumerar las posibles razones para no incluirlo y el impacto de esa decisión. Diferentes formas de representación que el Producto permite.
- Otra información: puede consistir en Orientaciones o Guías para brindar detalles sobre la realización del producto, Conceptos involucrados, etc.

El orden de presentación está relacionado con una secuencia posible de su aparición a lo largo del ciclo de vida de un proceso típico que tal como sugiere ISO 13407 según ya hemos visto avanza en iteraciones de valor creciente por la identificación del contexto de uso, la especificación de los requerimientos de usuario, las propuestas de diseño y su evaluación.

## Modelo de usuario

### *Propósito*

Para poder optimizar el diseño de la interacción y la experiencia del usuario es necesario conocer y especificar las características relevantes de las personas que utilizarán el producto o sistema en desarrollo. Una interacción que es apropiada para el usuario infrecuente o casual que necesita interfaces fáciles de aprender y de recordar, puede resultar desmotivadora y hasta improductiva para un usuario experto, que demandará interfaces altamente eficientes, poderosas y flexibles. Como lo plantea MMU-ISO es necesario identificar y documentar atributos del usuario como conocimientos previos, capacidades, experiencia, etc. que exceden la definición de del Actor en un Caso de Uso.

### *Descripción y relaciones*

**Descripción** Contiene la colección de atributos para un usuario “típico”. Un sistema o producto puede necesitar la especificación de más de un modelo o perfil de usuario. Los atributos que pueden resultar relevantes de un usuario son sus habilidades y destrezas, capacidades y limitaciones físicas y mentales, experiencia en la tarea, frecuencia de uso prevista, formación o educación previa, nacionalidad, identidad cultural, etc.

Para realizar un Perfil de usuario se pueden utilizar diferentes técnicas. En todos los casos será importante priorizar aquellos métodos y técnicas que permitan comprender y especificar el mejor conocimiento posible de los usuarios futuros del sistema o producto en desarrollo.

En [Hackos and Redish 1998] se presenta el método para construir *Perfiles de usuario*. Utilizan tanto un formato basado en listas como uno más personalizado que incluye narraciones e ilustraciones. La construcción de un Perfil de Usuario según estos autores sigue tres pasos: reunir información relevante, comprender y categorizar los tipos de usuarios, construir los perfiles. Para la recolección de información se utilizan técnicas etnográficas, entrevistas, cuestionarios, etc. Las características a incluir en los perfiles incluyen: demográficas (edad, género, nivel socio económico), experiencia en la tarea, educación, experiencia con computadoras, conocimiento de dominio, etc. En la Figura 7.10 se muestra un ejemplo de Perfil generado con esta técnica.

La técnica denominada *Personajes* [Cooper 2004, Pruitt and Adlin 2006] consiste en elaborar una descripción rica de usuarios típicos del sistema en desarrollo. No describen gente real, sino que son sintetizadas de usuarios reales involucrados en ejercicios de recolección de datos. Los personajes se describen con rigor y detalle y quedan definidas por un conjunto de metas. El método asume que basar la descripción en el cargo o trabajo no es apropiado, ya que personas con roles o cargos muy diferentes pueden tener las mismas metas para el uso de un producto o sistema. Además de las metas, el Personaje incluye la descripción de habilidades, actitudes, tareas y entorno del usuario representado. Cada Personaje tiene un nombre, una foto y algunos detalles personales. Ver las Figuras 7.11 y 7.12.

Los Roles de Usuario de [Constantine and Lockwood 1999] son “parientes cercanos” de los Personajes y los Actores de los Casos de Uso. Un Rol de Usuario no intenta representar una persona real sino que modela en forma abstracta una relación entre un usuario y el sistema. Contiene el conjunto de necesidades, comportamientos y expectativas de un usuario. En su forma más compacta se define por tres “Cs”: Contexto (las responsabilidades que el rol asume en el contexto dentro del cual se desempeña), Características (patrones de interacción, comportamientos y actitudes

que se esperan del rol) y Criterios (objetivos de diseño relacionados con el soporte efectivo del rol). Constantine y Lockwood indican que los roles, actores y personajes están fuertemente interrelacionados y promueven la construcción de los tres. Según estos autores, un sistema tendrá varios actores; para cada actor se podrán encontrar uno o más roles; para algunos roles clave será conveniente crear personajes detallados. La Figura 7.13 muestra una hoja de control con los criterios a seleccionar para las tres Cs.

**Slot que completa** [Especificación técnica]

**Rol responsable** Especialista de UX

**Roles que pueden modificarlo** Diseñador de UX, Analista

**Es input para las tareas** Entender y especificar el contexto de uso. Diseñar la experiencia de usuario, Revisar el diseño de la experiencia de usuario, Preparar pruebas de usabilidad,

**Es output de las tareas** Especificar requerimientos y objetivos de experiencia de usuario, entender y especificar el contexto de uso

### *Ejemplo*

Las Figuras 7.10, 7.11, 7.12 y 7.13 muestran algunos ejemplos las técnicas indicadas para la construcción del Perfil de Usuario.

Travel Agent (primary) Characteristic Ranges	
Age:	25–40 years (Average: 32 years)
Gender:	80% female
Job Titles:	Travel agent, Travel specialist, Travel associate
Experience Level:	0–10 years (Typical: 3 years)
Work Hours:	40 hours per week; days and times depend on the company
Education:	High school to Bachelors degree (Typical: some college)
Location:	Anywhere in the U.S. (Predominantly mid-west)
Income:	\$25,000–\$50,000/year; depends on experience level and location (Average: \$35,000/year)
Technology:	Some computer experience; high speed internet connection
Disabilities:	No specific limitations
Family:	Single or married (Predominantly married with 1 child)

Figura 7.10: Perfil de Usuario según Hackos para un agente de viajes

## **Meta de usabilidad**

### *Propósito*

Establecer metas de usabilidad persigue dos propósitos. En primer lugar, un conjunto articulado de metas de usabilidad ayuda a los diseñadores a focalizar los esfuerzos durante el proceso con un objetivo claro y algún tipo de parámetro contra el cual evaluar sus logros. Si los diseñadores y usuarios acuerdan que la facilidad de uso para usuarios expertos es más importante que la facilidad de aprendizaje para novatos, los esfuerzos de diseño se enfocarán sobre las alternativas que provean velocidad y eficiencia en el uso.

En segundo lugar, las metas de usabilidad servirán como criterios de aceptación durante las pruebas de usabilidad. El ciclo de diseño y pruebas de usabilidad se itera hasta que las evaluaciones indican que los criterios de usabilidad han sido satisfechos. Estos criterios responden a las metas de usabilidad planteadas para el proyecto. Además estas

- **Role:** Senior designer on project, go-to person for team members, other users in a large organization.
- **Experience:** 15-20+ years on the job; has used every tool out there.
- **Measured on:** Managing project scope, hitting deadlines, team performance, manufacturing yields.
- **Typical Day:** Answers a lot of questions, tries to keep up on her own design work as well. Integrates various tools in the flow and works on harder import/export issues.
- **Likes:** Being the expert, solving hard problems.
- **Dreads:** Scope creep; any surprises during the last week of a design project.
- **Uses SupportNet for:** checking out error messages, local version information, finding related files, methodology and first party tool information, advanced email, paper information tips/tricks.

## The Superuser



Improve SupportNet with: expert-level technical content, acknowledgement of her advanced expertise by giving her preferred access to backline technical help, no beginners please.

Figura 7.11: Ejemplo de Personaje, de [Pruitt and Adlin 2006]



**zylom**  
Have fun.

**Meet Maria (49)**

Thursday, October 26, 2006

### *Maria (49)*

Maria lives outside of a big city, in a self-owned home and enjoys the benefits of having a broadband internet connection. She plays a couple of games on the internet every day. She prefers to play in the evening after dinner when her husband is watching TV. The computer resides in the living room, and while she is playing she chats a bit with her husband or kids.

Maria is a homemaker and mother with an above average income family. Her husband has a few more years to go before he can start enjoying his retirement and pension. Maria is a little addicted to playing online games but she tells herself and her family that playing games helps her to train her brain.

The available leisure time and financial freedom that the family currently enjoys is being used to travel and live life in a comfortable way. Maria has an open mind and enjoys her freedom and her comfortable life.



Figura 7.12: Ejemplo de Personaje, de [Pruitt and Adlin 2006]

**Constantine & Lockwood, Ltd.**

### User Role Checklist for Agile Modeling

A user role is a relationship with a system. Tasks are performed by users within roles. Tasks are about what users do, roles are about how they do it. The key to succinct characterization of user roles is differential description. How is this role not like other roles? What is distinctive or salient about it in comparison to other roles? Listed below are typical factors for consideration as potentially relevant and useful for characterizing a user role. They may or may not apply to a given role.

#### Context (within which role is played)

Is there anything special or distinguishing about this role in terms of

<b>ENVIRONMENT</b>	<input type="checkbox"/> overall job, workflow, or activity within which role is played	Examples "Follow-up of prior purchase."
	<input type="checkbox"/> physical environment in which role is played	"Typical noisy office."
	<input type="checkbox"/> social situation in which role is played	"With field research partners."
	<input type="checkbox"/> relationships with indirect users in role	"Customer on telephone."
	<input type="checkbox"/> external sources of information, such as paper forms, telephone, visual observation, in-person interview	"Phone review of packing slip."
<b>INCUMBENTS</b>	<input type="checkbox"/> background of role incumbents in terms of training, education, or experience	"Cursorry OTJ training."
	<input type="checkbox"/> system knowledge expected or required within role	"Fully familiar from long use."
	<input type="checkbox"/> domain knowledge expected or required within role	"No retail management knowledge."
	<input type="checkbox"/> distribution of user skills in terms of novice, intermediate, or expert usage patterns	"Mostly perpetual novices."
	<input type="checkbox"/> required or discretionary nature of role	"Part of regular job."

#### Characteristics (of performance of role)

Is there anything special or distinguishing about this role in terms of

<input type="checkbox"/> orientation, attitude, or emotional state typical within role	"Harried, under pressure."
<input type="checkbox"/> frequency with which role is played	"Less than once a month."
<input type="checkbox"/> regularity with which role is played	"Impulse buy after infermercial runs."
<input type="checkbox"/> intensity of interaction in the role	"Sporadic bursts of calls."
<input type="checkbox"/> duration of interaction in the role	"Full 8-hour shift."
<input type="checkbox"/> complexity of interaction in the role	"No-brainer."
<input type="checkbox"/> predictability of interaction in the role	"Scripted sales protocol."
<input type="checkbox"/> volume of information handled in the role	"Limited items available."
<input type="checkbox"/> direction of information flow to or from system	"Data entry."

#### Criteria (for support of role)

Are there any design objectives that are particularly important for this role, such as

<input type="checkbox"/> ease of learning	<input type="checkbox"/> enhancement of proficiency
<input type="checkbox"/> retention of learning	<input type="checkbox"/> user convenience
<input type="checkbox"/> efficiency of interaction	<input type="checkbox"/> accuracy of input
<input type="checkbox"/> reliability of interaction	<input type="checkbox"/> clarity of presentation
<input type="checkbox"/> user satisfaction	<input type="checkbox"/> comprehensibility of presentation

Are there any specific functions, features, facilities, capabilities, or content that are particularly important for this role to be performed effectively?

Contact: [constantine@toruse.com](mailto:constantine@toruse.com) © 2004, Constantine & Lockwood Ltd

Figura 7.13: Lista de control para definir las tres Cs del Rol de Uso de [Constantine and Lockwood 1999]

metas ayudan a priorizar los esfuerzos para reparar los problemas de usabilidad revelados por las pruebas. Cuando los recursos son limitados, los problemas que se vinculan directamente con las metas identificadas para el proyecto deben tener la máxima prioridad de resolución.

De esta manera, se puede decir que las metas de usabilidad conducen el proceso de diseño y de evaluación de la interacción y la experiencia del usuario [Mayhew 1999].

### *Descripción y relaciones*

**Descripción** Las metas de usabilidad se basan en la elaboración de los Modelos/perfiles de usuario y en el Análisis de Tareas (que genera el Modelo de Tareas) y en las metas y reglas del negocio específico. Las metas de usabilidad pueden ser cualitativas o cuantitativas y pueden ser categorizadas empleando la definición de usabilidad como vimos en el Capítulo 2:

- **efectividad:** relaciona las metas o sub-metas del usuario con la precisión y completitud con la cual pueden ser alcanzadas utilizando el sistema. Por ejemplo, si la meta deseada es reproducir un documento de dos páginas en un formato especificado, entonces la precisión podría especificarse y medirse por el número de errores de ortografía y el número de desviaciones del formato especificado y la completitud por el número de palabras del documento transcrito dividido el número de palabras del documento original.
- **eficiencia:** relaciona el nivel de efectividad alcanzado en función de los recursos utilizados. Recursos relevantes pueden incluir el esfuerzo mental o físico, tiempo, materiales, dinero. Si la meta es imprimir copias de un reporte, la eficiencia podría especificarse y medirse por el número de copias impresas usables del reporte dividido por recursos empleados como las horas de trabajo, costo del proceso o materiales consumidos.
- **satisfacción:** describe el confort y aceptabilidad en el uso. Se puede especificar y medir con escalas de medición de actitudes o proporciones de comentarios negativos y positivos en el uso. Las medidas de satisfacción pueden evaluar actitudes en el uso del producto o percepciones del usuario en temas de utilidad, facilidad de aprendizaje.

**Slot que completa** [Especificación técnica]

**Rol responsable** Especialista en experiencia de usuario

**Roles que pueden modificarlo** Analista, Diseñador de experiencia de usuario

**Es input para las tareas** Diseñar la experiencia de usuario, Revisar el diseño de experiencia de usuario, Preparar pruebas de usabilidad

**Es output de las tareas** Especificar requerimientos y metas de experiencia de usuario, Entender y especificar el Contexto de Uso

### *Ejemplo*

Las tablas 7.2 y 7.3, tomadas de ISO 9241-11 [ISO/IEC 1998] muestran ejemplos de las metas de usabilidad organizadas en las tres categorías básica.



Medidas de efectividad	Medidas de eficiencia	Medidas de satisfacción
Porcentaje de metas logradas	Tiempo para completar una tarea	Escala de satisfacción
Porcentaje de usuarios que completan una tarea	Tareas completadas por unidad de tiempo	Tasa de uso en el tiempo
Precisión promedio de las tareas completadas	Costo monetario de realizar una tarea	Frecuencia de quejas

Tabla 7.2: Criterios generales de usabilidad

## Modelo de tareas

### *Propósito*

El propósito básico de un Modelo de Tareas es el especificar el Contexto de Uso de la aplicación, particularmente en los aspectos operativos y facilitar que el diseño de la interacción pueda mantener una postura de diseño “desde cero” utilizando ese contexto y los conocimientos de las condiciones cognitivas y sociales del usuario para establecer la mejor asignación de responsabilidades entre el sistema y las personas que lo utilizarán. Se trata de no dar por sentada la distribución actual de responsabilidades, sino de al menos validarla como paso del proyecto.

### *Descripción y relaciones*

**Descripción** Un modelo de tareas (resultado del Paso:Identificar y documentar las tareas del usuario, Tarea:Identificar y especificar el Contexto de Uso) se define como una descripción de una tarea interactiva a realizar por el usuario de una aplicación utilizando su interfaz de usuario [Limbourg and Vanderdonckt 2004]. Los elementos individuales de un modelo de tareas representan acciones específicas que el usuario puede realizar. También debe incluirse en el modelo la información en el ordenamiento de subtareas y las condiciones para la ejecución de tareas.

Para obtener un Modelo de Tareas se pueden usar diferentes métodos que se diferencian en el grado de formalismo de su notación, poder de expresividad y finalidad. Si bien todos ellos representan las tareas del usuario con el sistema, la finalidad del modelo puede ser diferente [Lorés 2001] y ello se ve reflejado en las formas de representación y la profundidad del análisis:

- **Modelos cognitivos:** identifican secuencias de comportamiento correctas, representan el conocimiento que debe poseer un usuario sobre el uso del sistema; parten de la descripción de las tareas para especificar el conocimiento del usuario
- **Modelos predictivos:** para la evaluación del rendimiento humano; describen secuencias de comportamiento y el conocimiento que necesita el usuario para su ejecución; el análisis se enfoca en las rutinas del comportamiento.
- **Modelos descriptivos:** permiten obtener una descripción más o menos completa del sistema a partir de la información obtenida de las tareas

La Tabla 7.4 muestra algunos métodos para realizar los tipos de modelos que se mencionaron.

**Slot que completa** [Especificación técnica]

**Rol responsable** Especialista en UX

<b>Criterios</b>	<b>Efectividad</b>	<b>Eficiencia</b>	<b>Satisfacción</b>
Uso de expertos	Número de tareas realizadas  Porcentaje de funciones relevantes usadas	Eficiencia relativa a un usuario experto	Escala de satisfacción en funciones avanzadas
Uso de novatos	Porcentaje de tareas completamente exitosas en el primer intento	Tiempo insumido en el primer intento  Eficiencia relativa del primer intento	Tasa de uso voluntario
Uso infrecuente o intermitente		Tiempo insumido en aprender nuevamente una función Número de errores persistentes	Frecuencia de reuso
Disminuir requerimientos de soporte	Número de referencias a la documentación Número de llamadas a soporte Número de accesos a ayuda online	Tiempo productivo  Tiempo para aprender	
Facilidad de aprendizaje	Número de funciones aprendidas Porcentaje de usuarios capaces de aprender a criterio	Tiempo para aprender Tiempo para reaprender  Eficiencia relativa mientras se aprende	Escala de facilidad de uso
Tolerancia a errores	Porcentaje de errores corregidos Número de errores de usuario tolerados	Tiempo insumido en corregir errores	Escala de manejo de errores
Legibilidad	Porcentaje de palabras leídas correctamente a una distancia dada		

Tabla 7.3: Medidas para objetivos de usabilidad

Método	Tipo	Notación	Comentarios
HTA	Cognitivo	Grafico	Modelo de descomposición del conocimiento
GOMS	Cognitivo	Textual	Familia de lenguajes para describir el conocimiento
UAN	Cognitivo	Grafico	Notacion para el estilo de manipulación directa
KLM	Predictivo	Textual	Medición del rendimiento humano
TAG	Predictivo	Textual	Medida de la consistencia
CTT	Descriptivo	Grafico	Herramientas de soporte al análisis y verificación

Tabla 7.4: Métodos para elaborar Modelos de Tareas

**Roles que pueden modificarlo** Diseñador de UX, Analista, Representante de Usuarios

**Es input para las tareas** Diseñar la UX

**Es output de las tareas** Entender y especificar el Contexto de uso

### Ejemplos

El **Análisis Jerárquico de Tareas (HTA)** fue desarrollado por Annet y Duncan [Annet and Duncan 1967]. Realiza una descripción de tareas en términos de operaciones y planes. Las operaciones (descomposición en subtareas) son actividades que realizan las personas para lograr un objetivo y los planes son una descripción de las condiciones que se tienen que dar cuando se realiza cada una de las actividades. Las operaciones se pueden descomponer de forma jerárquica y se adigna un plan a cada una de las subtareas que aparecen. Se define un objetivo como un estado determinado del sistema que desea alcanzar un usuario. HTA implica tres etapas: recopilar información (revisión de documentos, entrevistas, cuestionarios, etc.), diagramar y analizar proveyendo sugerencias y guías para la tarea de diseño.

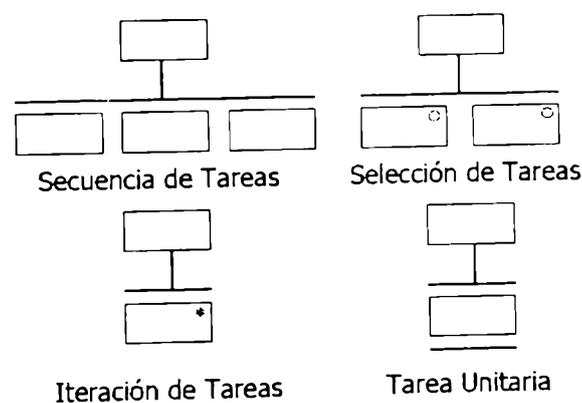


Figura 7.14: Notación en HTA, de [Lorés 2001]

El formato gráfico se parece a un árbol con ramas y subramas en función de las necesidades (ver Figura 7.14). La Figura 7.15 muestra un ejemplo de HTA representado por un diagrama de árbol y una tabla (tomado de [Lorés 2001]).

**GOMS**, propuesto por Card, Moran y Newell [Card et al. 2000] es una familia de lenguajes que se basan en la visión de un usuario como un sistema procesador de información. Para cada tarea se describe el objetivo a satisfacer (*Goal*), el conjunto de operaciones (*Operations*) que el sistema pone a disposición del usuario para la interacción, los métodos disponibles para llevar a cabo esas operaciones (*Methods*) y un conjunto de reglas de selección (*Selection*) para determinar la alternativa más conveniente para cada caso.

Los objetivos son las metas que se propone conseguir el usuario. Un objetivo contiene información de la intención del usuario. Para ello, debe realizar una serie de operaciones

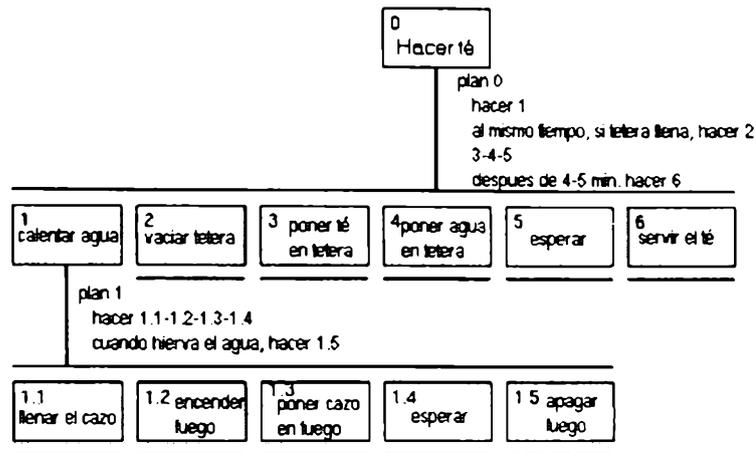


Figura 6 Descripción en HTA de la preparación del té

0. Hacer té

1. Calentar el agua

1.1 Llenar cazo

1.2 Encender fuego

1.3 Poner cazo en fuego

1.4 Esperar

1.5 Apagar fuego

2. Vaciar tetera

3. Poner hojas de té en tetera

4. Verter el agua

5. Esperar

6. Servir el té

Plan 0: hacer 1.

si tetera está llena,

entonces hacer 2 al mismo tiempo

hacer 3-4-5

Cuando el té ha reposado, hacer 6

Plan 1: hacer 1.1-1.2-1.3-1.4

Cuando el agua está hirviendo, hacer 1.5

Figura 7.15: Ejemplo de HTA, de [Lorés 2001]

básicas, que GOMS toma como las unidades elementales de percepción, motoras o actos cognitivos cuya ejecución es necesaria para cambiar algún aspecto del modelo mental del usuario, o bien, para modificar el entorno. Este tipo de acciones puede afectar al sistema (pulsar una tecla) o bien, sólo al estado mental del usuario (leer el cuadro de diálogo). Existe un grado de flexibilidad acerca de la granularidad de las operaciones (amplitud de cada operación). Para llevar a cabo estas operaciones, existen varias posibilidades de descomposición de una tarea en subtareas. Por ejemplo, en un gestor de ventanas, se puede cerrar la ventana mediante ratón en un menú o teclado (atajo). Cada una de estas posibilidades será un método.

GOAL: CERRAR-VENTANA

```
[select GOAL: USAR-METODO-RATON
      MOVER-RATON-A-MENU-VENTANA
      ABRIR- MENU
      CLICK-SOBRE-OPCION-CERRAR
GOAL: USAR-METODO-TECLADO
      PULSAR-TECLAS-ALT-F4]
```

Cuando hay más de una alternativa, podemos indicar una serie de condiciones y reglas para tomar la mejor alternativa (método):

```
METHODS: IF (USUARIO-EXPERTO)USAR-METODO-TECLADO
           ELSE USAR-METODO-RATON]
```

Podemos descomponer los objetivos en subobjetivos.

GOAL: EDITAR-DOCUMENTO

GOAL: ABRIR-DOCUMENTO

La descomposición de tareas permite comprender las estrategias para resolver problemas del dominio de la aplicación. El objetivo del análisis jerárquico de tareas es la de producir una descomposición de tareas, de modo que se pueda seguir paso a paso el método de resolución. GOMS puede servir también para medir rendimientos. La profundidad de subtareas se puede usar para estimar los requerimientos de la memoria de corto plazo e incluso para estimar tiempo de respuesta.

**KLM o Keystroke Level Mode**, es una de las variantes de GOMS, que reduce el conjunto de operaciones disponibles a la pulsación de teclas y movimiento del mouse. Esta simplificación permite obtener predicciones del tiempo que una persona empleará para la realización de una tarea. Estas mediciones parten de valores experimentales que indican la duración de actividades elementales. Por ejemplo, el tiempo de planificación de una tarea se puede estimar en 2 a 3 segundos si ya está definida y de 5 a 30 seg. si hay que pensarla, la pulsación de teclado para un usuario normal es de 0.28 seg, apuntar con el mouse dura 1,10 seg. etc.

**CTT o ConcurTaskTrees** es una notación desarrollada por Fabio Paternó [Paternó 1999, Paternó et al. 1997] cuyo principal finalidad es la de poder representar las relaciones temporales existentes entre las actividades y usuarios que son necesarios para llevar a cabo en las tareas. Esta notación es especialmente útil para aplicaciones CSCW. Una de las principales ventajas de esta notación es su facilidad de uso, lo que hace que sea aplicable a proyectos reales con aplicaciones de un tamaño mediano y que con llevan especificaciones de cierta complejidad. La notación genera una representación gráfica en forma de árbol de la descomposición jerárquica de las tareas existentes en el sistema. Se permite la utilización de un conjunto de operadores, sacados de la notación de LOTOS, para describir las relaciones temporales entre tareas (secuencialidad, concurrencia, recursión, iteración...). La Figura 7.16 muestra un ejemplo de modelo de tareas construido con CTT.

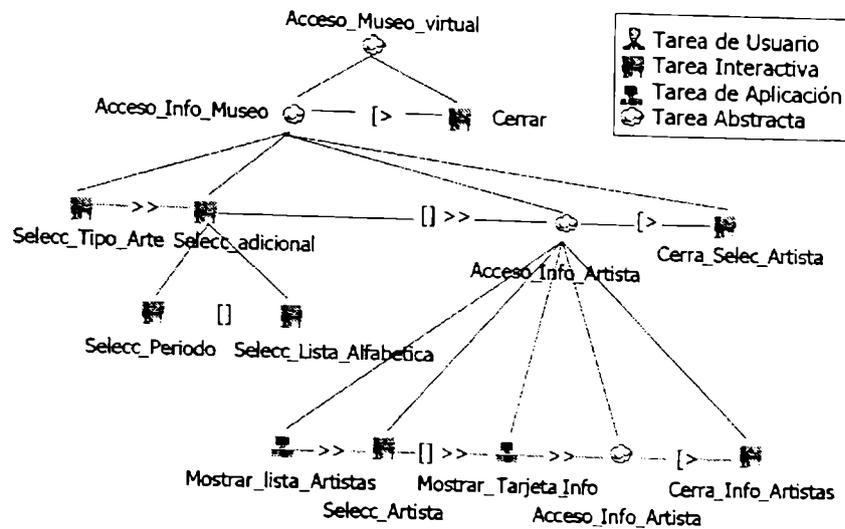


Figura 7.16: Uso de CTT en la elaboración de un Modelo de Tareas

En su metodología **Diseño Centrado en el Uso**, Larry Constantine [Constantine and Lockwood 1999] presenta un **Modelo de Tareas basado en Casos de Tareas**, una adaptación de Casos de Uso. La modelización de tareas está en el corazón del método con foco en la performance del usuario. Un Caso de Tarea representa una intención singular llevada a cabo por un usuario en algún rol. Constantine los llama también Casos de uso esenciales, porque intentan describir la esencia de una tarea. Cada uno es expresado como un diálogo en el cual la intención del usuario y las responsabilidades del sistema son abstractas, simplificadas y desprovistas de cualquier asunción sobre la tecnología o su implementación. Esta forma de descripción intenta aproximarse mejor a la esencia de la tarea desde la perspectiva del usuario en un rol y evitar la consolidación prematura sobre el diseño de la interfaz de usuario.

### Concepto de la Experiencia de Usuario

El Concepto o Modelo Conceptual de la UX es una descripción de alto nivel del modo en que el sistema está organizado y opera de cara a usuario.

#### *Propósito*

Una vez formulado y consensuado, el concepto de la interacción se convierte en una hoja de ruta compartida por todo el equipo de proyecto sobre el camino a recorrer para brindar la experiencia de usuario deseada.

Este concepto se utiliza por todo el equipo de diseño como la base sobre la cual desarrollar en forma detallada y concreta cada componente de interacción que se necesite sin perder de vista el panorama general de la UX.

#### *Descripción y relaciones*

**Descripción** Se trata de una abstracción que esquematiza lo que la gente puede hacer con un producto y los conceptos necesarios para entender cómo interactuar con él [Johnson and Henderson 2002]. No es una descripción de la interfaz de usuario sino una estructura que esboza los conceptos y las relaciones entre ellos que forman la base del producto o sistema y su relación con los usuarios. De acuerdo con el tipo de sistema o producto en desarrollo, el Concepto de UX puede contener las principales metáforas y analogías a utilizar en el sistema, los conceptos del dominio que el sistema expone al usuario, la relación entre tales conceptos y el mapeo entre

ellos y las tareas que el usuario realiza. La importancia de contar con un concepto de la UX previo al diseño de la IU se basa en el hecho de que durante el uso de un sistema interactivo los usuarios construyen en sus mentes un modelo del sistema y su forma de funcionamiento. Esto les permite predecir su comportamiento y generalizar a otras tareas a partir de lo aprendido. Contar con un concepto bien diseñado de la UX facilitará esa acción de “descifrado” por parte del usuario y en consecuencia aumentará la usabilidad en términos de facilidad de aprendizaje.

**Slot que completa** [Diseño técnico]

**Rol responsable** El Diseñador de la UX

**Roles que pueden modificarlo** También puede intervenir en su desarrollo el Especialista de UX

**Es input para las tareas** El Concepto de UX es el input principal de la tarea Diseñar componentes de interacción

**Es output de las tareas** Constituye el resultado de la tarea Diseñar la UX.

### Prototipo de Experiencia de Usuario

El prototipo es un modelo restringido de la interacción de usuario, que se utiliza tanto para extraer requerimientos como para las tareas de diseño (para explorar y para validar ideas).

#### *Propósito*

El principal propósito para la utilización de este producto de trabajo consiste en reducir el riesgo de avanzar en una dirección para el desarrollo hasta un momento en que cualquier cambio resulte tanto o más costoso que devolver el proyecto a su estado inicial.

Una de las ventajas más importantes de un prototipo es la de poder testear una propuesta de UX mucho tiempo antes de que el desarrollo real del sistema comience, lo que permite asegurar en alguna medida que el sistema a construir será el adecuado.

#### *Descripción y relaciones*

**Descripción** Un prototipo puede ser tan formal o informal como sea necesario. Existe abundante literatura en el campo de HCI, Diseño de interacción y usabilidad sobre las posibilidades y características de los prototipos (ejecutable o no ejecutable, alta o baja fidelidad, profundo o extenso, etc.); por ejemplo [Arnowitz et al. 2006], [Sharp et al. 2007], Buxton [2007].

**Slot que completa** [Diseño técnico]

**Rol responsable** El diseñador de UX lo emplea para explorar y/o validar ideas de diseño antes consolidar decisiones que luego resulte muy caro modificar. Los analistas y especialistas de UX para entender los requerimientos de usuarios y luego planificar las actividades de testing

**Roles que pueden modificarlo** Cualquiera de los Stakeholder, Tester de UX

**Es input para las tareas** Revisar el diseño de la UX, Preparar pruebas de usabilidad

**Es output de las tareas** Entender y especificar el contexto de uso, diseñar la UX, Ejecutar pruebas de usabilidad

#### *Ejemplo*

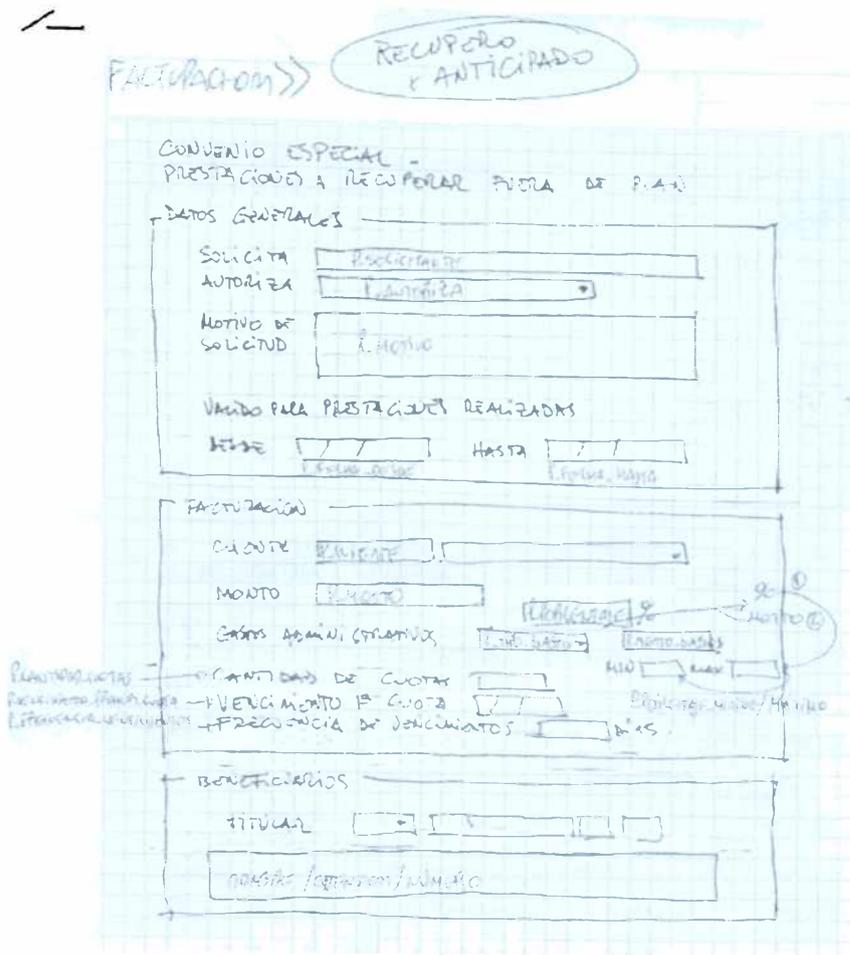


Figura 7.17: Prototipo de IU de baja fidelidad

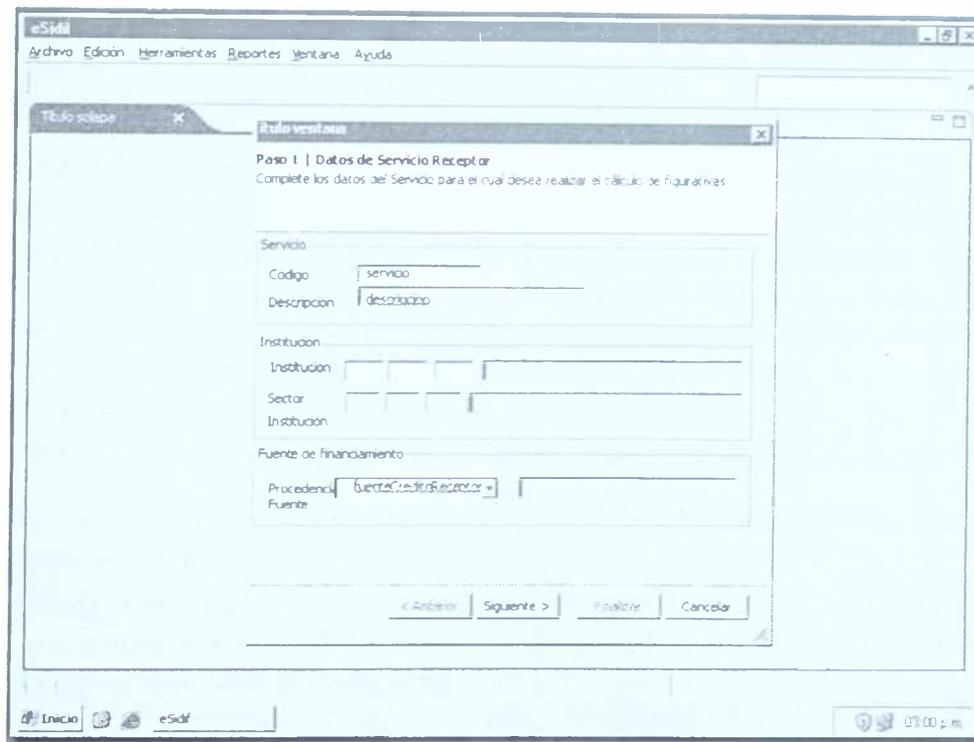


Figura 7.18: Prototipo de IU de alta fidelidad

En las figuras 7.17 y 7.18 se muestran dos ejemplos de prototipos de interfaz de usuario de baja y alta fidelidad, respectivamente.

### ***Adaptaciones***

El formato y técnica empleada para la realización de prototipos no constituye un aspecto relevante en sí mismo. Deberá elegirse el que resulte adecuado al proyecto y el propósito buscado. El mismo criterio se aplicará respecto de la extensión de la porción del sistema a prototipar (alguna interacción específica o todo el conjunto).

Lo importante es comprender que para lograr el objetivo de probar tempranamente las interfaces de usuario, el prototipo debe ser más económico de construir que el sistema real, pero tener las suficientes capacidades para permitir una prueba con usuarios que resulte significativa.

### **Storyboard de Experiencia de Usuario**

Un Storyboard de UX consiste en la descripción de la interacción del usuario con el sistema para un escenario específico.

#### ***Propósito***

Puede considerarse como lo plantean [Sharp et al. 2007] que un storyboard es una clase de prototipo. Especialmente, uno de baja fidelidad. Sin embargo, el rol decisivo que desempeña para describir la dinámica de una interacción nos inclina a ubicarlo entre la lista de productos de trabajo que OpenUP/MMU-ISO debe proveer.

Este producto de trabajo es uno de los componentes de diseño indicado en el MMU-ISO cuya carencia detectamos en el assessment. Para la propuesta nos basamos, además de los artefactos asociados a procesos DCU que sugiere el MMU-ISO, en las prácticas reconocidas por la comunidad de Experiencia de usuario, por ejemplo [Mayhew 1999], [Beyer and Holtzblatt 1998]. También reconocemos como antecedente al artefacto: Storyboard del plugin UX de RUP [IBM 2005b]

#### ***Descripción y relaciones***

**Descripción** Un Storyboard de UX puede contener tres elementos:

- Una descripción del flujo de eventos entre el usuario y el sistema
- Diagramas y gráficos que muestran la interacción del usuario. Pueden ser tan formales como los diagramas UML de colaboración y secuencia, que sugiere el Plugin UX de RUP o simples bocetos a la manera utilizada en la programación de películas para describir la serie de pasos que se dan para utilizar la funcionalidad del sistema durante un escenario específico.
- Referencias explícitas a las metas de usabilidad vinculadas con el escenario (tiempos de ejecución previstos, máxima tasa de error aceptable, etc.)

Es habitual que en la medida que se han construido Prototipos de UX un Storyboard haga referencia a ellos.

Debe construirse un Storyboard por cada escenario de utilización del sistema previsto, cualquiera sea el método de modelización adoptado para las situaciones de uso (Casos de Uso, User stories, etc.).

**Slot que completa** [Diseño técnico]

**Rol responsable** Los Storyboards de UX deben construirse y ser mantenidos por el Diseñador de UX.



El propósito de este artefacto es representar la navegación del usuario por el sistema y contribuir a conformar el Modelo de la UX. El alcance de este mapa es obviamente todo el sistema (esto lo distingue de los storyboards, que focalizan en un escenario en particular).

El Mapa de Navegación ayuda a unificar la relación entre los diferentes storyboards de la aplicación y trasmite la estructura de un Modelo de la UX.

Los mapas de navegación son habituales en proyectos de aplicaciones web, sin embargo resultan de utilidad en el diseño de cualquier tipo sistema interactivo. En particular, dentro de la familia de procesos basados en UP podemos reconocer su presencia en las orientaciones Usability engineering en RUP y Modeling user experience (UX Plugin) [IBM 2005b].

### *Descripción y relaciones*

**Descripción** Muestra los componentes de interacción y los caminos de navegación entre ellos. Puede representarse con diagramas UML de clase, como muestra la figura 7.21, ya sea en un único diagrama o en varios, ajustando el nivel de detalle incluido.

**Slot que completa** [Diseño técnico]

**Rol responsable** El Diseñador de UX

**Roles que pueden modificarlo** Pueden participar en su redefinición y mantenimiento el Analista de UX, cualquier otro Analista y cualquiera de los roles de Usuario.

**Es input para las tareas** Diseñar la UX, Revisar el diseño de la UX, Preparar pruebas de usabilidad

**Es output de las tareas** Diseñar la UX

### *Ejemplo*

La Figura 7.21 muestra un mapa de navegación presentado por el Plugin para la UX de RUP.

### **Documento de usuario y entrenamiento**

#### *Propósito*

Este producto intenta proveer guía y soporte al usuario para el uso del producto.

#### *Descripción y relaciones*

**Descripción** Este producto comprende todos los materiales que brinden asistencia al usuario final para el aprendizaje, utilización, operación y mantenimiento del sistema.

Algunos de los formatos que puede adoptar este producto son:

- Manual de usuario
- Guías de operación
- Guías de mantenimiento
- Ayudas en línea
- Notas de release

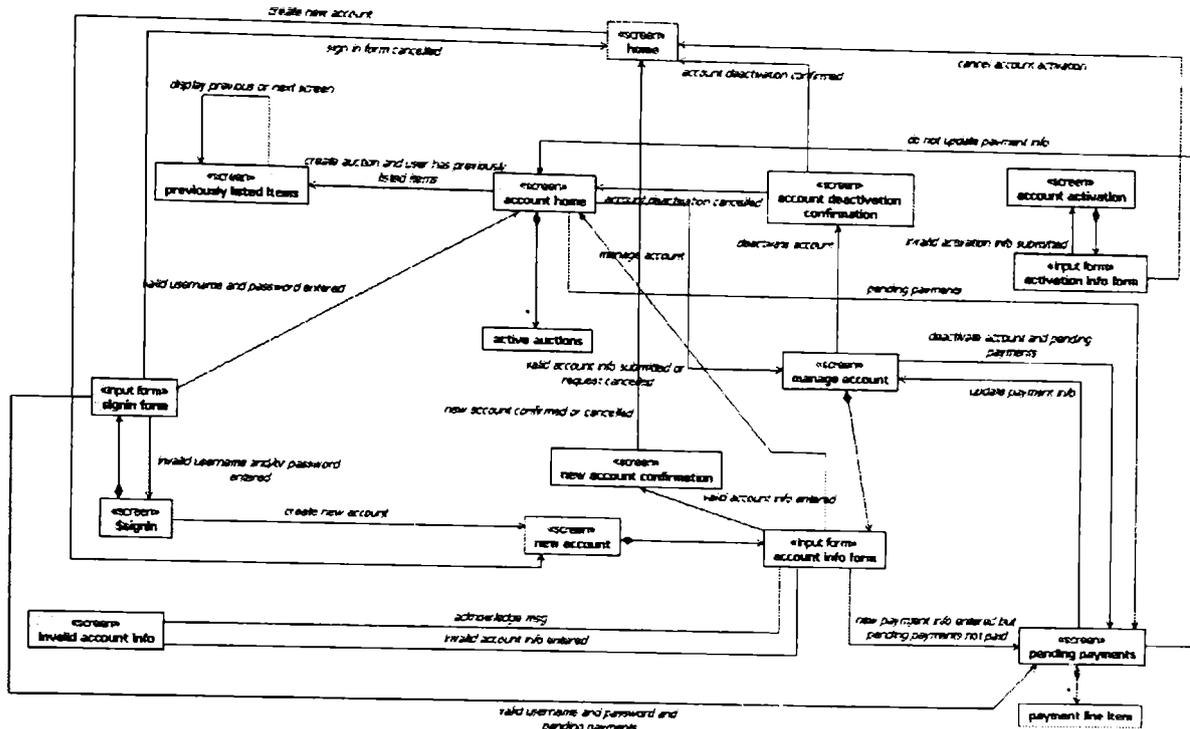


Figura 7.21: Mapa de navegación

Estos productos pueden ser generados a partir de otros productos de trabajo. Por ejemplo, es común realizar los Manuales de usuario o Ayudas en línea a partir de las especificaciones como Casos de uso. En términos de la posibilidad de satisfacer al MMU-ISO, nos interesa dar soporte al Proceso DCU 7 "Presentar y operar el sistema" dando cumplimiento a indicadores sobre el manejo de requerimientos de soporte a usuarios, encargados de mantenimiento y otros interesados.

Este producto puede empezar a construirse en etapas tempranas del proceso, dependiendo del tipo de sistema.

**Slot que completa** [Especificación técnica]

**Rol responsable** Especialista de UX

**Roles que pueden modificarlo** Analista

**Es input para las tareas** Proveer entrenamiento y soporte al usuario

**Es output de las tareas** Diseñar y producir material de entrenamiento y soporte al usuario

### 7.2.3. Tareas

En este apartado incorporamos las Definiciones de Tareas que utilizarán, modificarán y producirán los Productos de Trabajo definidos en el anterior.

Al igual que para los Roles y los Productos de Trabajo, aprovecharemos como antecedentes no sólo las especificaciones del MMU-ISO y la bibliografía de Diseño Centrado en Usuario, sino también las Definiciones de Tareas presentes en diversas configuraciones de la familia del Proceso Unificado (RUP[Krutchen 2000], Enterprise Unified Process[Ambler et al. 2005], Agile Unified Process[Ambler 2006]), por ejemplo las provenientes de la disciplina Modelización del Negocio que promueven la adaptación del proceso, la comprensión del contexto organizacional donde se dará la inserción del sistema, etc.

Hemos incorporado en la Práctica DesCU las siguientes definiciones de Tareas:

- Adaptar el proceso de Desarrollo Centrado en Usuario (tailoring del proceso)
- Especificar requerimientos y objetivos de Experiencia de usuario
- Entender y especificar el contexto de uso
- Diseñar la experiencia de usuario
- Diseñar componentes de interacción
- Revisar el diseño de la experiencia de usuario
- Preparar pruebas de usabilidad
- Ejecutar pruebas de usabilidad
- Diseñar y producir material de entrenamiento y soporte al usuario
- Proveer entrenamiento y soporte al usuario

Podemos plantear un mapa de tareas de la Práctica DesCU donde se vean las relaciones con los procesos MMU-ISO, tal como refleja la figura 7.22.

A continuación detallamos la definición de las tareas. Vamos a utilizar como base el template incluido en la Orientación: Detallando una Tarea de la versión 1.5 del MAM [Eclipse 2008b].

- Propósito: ¿porqué hay que incluir la tarea? ¿Cuál es su meta?
- Descripción y relaciones: explica la ejecución de la tarea, menciona los productos de trabajo que sirven de input y outputs que se generan y los roles involucrados en su realización.
- Pasos: instrucciones para el ejecutante de la tarea. Un Paso se describe utilizando:
  - Nombre descriptivo del paso
  - Descripción para brindar más detalles del paso específico
- Orientación: guías para brindar detalles sobre la realización de la tarea o de algún paso específico. Este tópico lo incluimos en los casos que resulta conveniente asociar la tarea con una guía u orientación que explique la forma de llevar adelante un paso concreto o la tarea en general.

### **Tarea: Adaptar el proceso de Desarrollo Centrado en Usuario**

En esta Tarea se realiza la adaptación y personalización de la configuración OpenUP para adaptarla a las necesidades específicas del proyecto.

#### ***Propósito***

Por una parte esta tarea persigue el propósito de asegurar la cobertura de los procesos *DCU1. Asegurar el enfoque DCU en la estrategia de sistemas* y *DCU2. Planificar y gestionar el proceso DCU* del Nivel 1 del MMU-ISO.

Pero, por otra parte, la inclusión de una Tarea que se enfoque en adaptación del proceso de desarrollo en función del proyecto concreto es un requisito esencial para permitir la obtención del Nivel de Capacidad 3 (Proceso establecido) en lo referido a la consideración de los usuarios e involucrados en el proceso. Por ejemplo en los atributos de definición de proceso (PA3.1) tales como identificar la definición adecuada del proceso, adaptarlo al proyecto e implementarlo.

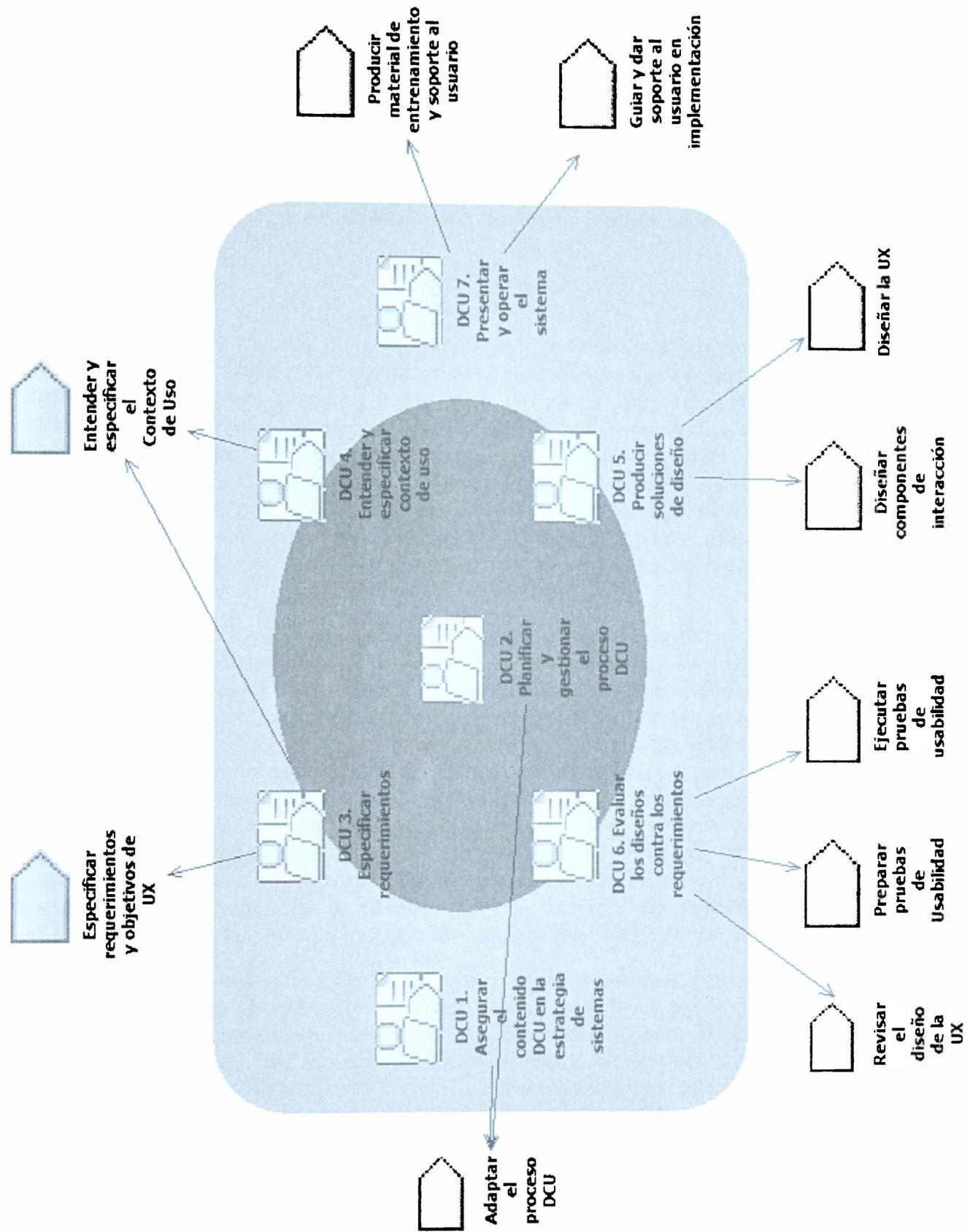


Figura 7.22: Mapa de relaciones entre Práctica DesCU y procesos MMU-ISO

### *Relaciones*

**Roles** Entre los roles definidos para esta Práctica, tanto el Patrocinador como el Líder técnico del dominio deberán colaborar en forma secundaria. Pero la responsabilidad de llevar adelante la tarea, recae en el Gerente del Proyecto.

**Input** Esta tarea se alimenta en forma mandatoria del proceso de desarrollo definido en la configuración OpenUP/MMU-ISO N[x], donde [x] indica el nivel de capacidad que se pretende alcanzar. En forma adicional se utilizará como input la Orientación: Definiciones de Procesos DCU y Orientación: Modelo ISO de Madurez en usabilidad

**Output** El resultado de la tarea es la instancia adaptada y publicada de la configuración elegida

### *Pasos*

**Analizar el proyecto** Es de vital importancia analizar la naturaleza, alcance y características específicas del proyecto para proveer un proceso que tenga el nivel de ceremonial y burocracia adecuado y relevante. En particular, en lo referido a la centralidad en el usuario, es de fundamental importancia conocer por ejemplo la disponibilidad real de todos los roles involucrados, el grado de innovación que se espera como resultado, el nivel de evangelización previa que puede requerirse para implementar todos los elementos de la configuración elegida, etc. Un proceso muy burocrático puede entorpecer el flujo normal del trabajo, un proceso demasiado "liviano" puede dar lugar a un entorno caótico donde no es posible asegurar ninguna calidad de resultados.

**Determinar el esfuerzo de adaptación adecuado** A partir del conocimiento previo de la naturaleza y especificidad del proyecto entre manos, en este paso se trata de acotar el esfuerzo de adaptación a lo estrictamente necesario. Puede ser que un proyecto no requiera de todos elementos de contenidos y proceso incluidos en la configuración de base. Este es el momento de definir por ejemplo las áreas o temas en las que los miembros del equipo disponen de suficiente experiencia previa y modos de trabajo que permitan evitar la introducción de nuevas herramientas y procesos. Por el contrario, pueden existir áreas donde el equipo no posee ninguna experiencia y que por lo tanto obliga a incluir procesos y guías que mantengan el proceso bajo control. En cualquiera de los casos, las mejoras al proceso se pueden realizar en forma incremental durante las sucesivas iteraciones, de modo que la determinación del esfuerzo debe incluir no sólo qué cambios realizar, sino también el cuándo.

**Desarrollar contenido específico DCU y configurar el proceso** OpenUP, en cualquiera de las configuraciones ofrecidas, es un framework de procesos que puede, y debe, ser configurado para las necesidades específicas. El Gerente del Proyecto (en cooperación estrecha con otros roles, fundamentalmente conocedores de buenas prácticas de ingeniería de software como el Especialista en UX) deberá determinar los contenidos del método que se incluirán, cuáles hay que desarrollar como nuevos, cuáles se podrán reutilizar de los activos de proceso del equipo, etc. Con todos esos elementos (roles, actores, tareas, artefactos) se procederá a instanciar una configuración específica para el proyecto.

**Definir el ciclo de vida DCU dentro del proceso** Es muy importante, dentro del proceso de adaptación, definir cual será el ciclo de vida general del proyecto (que hitos definiran los cambios de fases, cuantas iteraciones se desarrollaran para cada fase, etc). Esto es valido para cualquiera de las disciplinas involucradas en OpenUP, pero en el caso de los aspectos vinculados con usuarios, resulta crucial ya que entre

otras cosas deberá definirse cuándo involucrar a todos los roles específicos, cómo y cuándo dar por cerrado un ciclo de diseño y pasar a construcción, como y cuándo realizar pruebas de usuario en lugar de inspecciones de experto, etc.

**Publicar el proceso para todos los integrantes del equipo** Este paso pone a disposición de todo el equipo la configuración instanciada. Una forma práctica de realizar este paso es publicar la configuración como un sitio web accesible a todos, para que cada integrante sepa donde se espera que participe, que tareas y artefactos constituyen su responsabilidad (principal o secundaria) y cuáles son las orientaciones o guías que pueden ayudarlo en ese trabajo.

**Mantener el proceso** Para poder alcanzar realmente un nivel 3 de Capacidad en MMU-ISO no es suficiente con los pasos anteriores. Es necesario monitorear continuamente el estado del proceso y realizar todos los ajustes que haga falta durante el proyecto. La naturaleza iterativa de OpenUp facilita este monitoreo, ya que en cada comienzo de iteración se podrá evaluar los resultados de la anterior e incluir en la nueva planificación los cambios al proceso que resulten apropiados.

### *Orientación*

Para asegurar la conformidad en el nivel macro el MMU-ISO, puede usarse como guía el ciclo de vida básico del desarrollo centrado en usuarios según se indica en el diagrama de actividades de la Figura 7.23 que incluye los siete procesos DCU de MMU-ISO.

**Tarea: Especificar requerimientos y objetivos de experiencia de usuario**

#### *Propósito*

Esta tarea tiene dos propósitos principales. En primer lugar, ayudar a enfocar los esfuerzos relacionados con la UX durante todo el desarrollo brindando a los desarrolladores elementos concretos de guía y evaluación de las ideas de diseño. También sirve para establecer criterios de aceptación para las pruebas de usabilidad.

#### *Relaciones*

**Roles** El rol responsable de esta Tarea es el Especialista de UX, sin embargo es necesaria la participación también del Representante de usuarios, el Líder técnico de dominio y otros Analistas.

**Input** La Tarea se alimenta del Modelo de Usuario, el Modelo de Tareas y las metas de proyecto (de negocio y técnicas) disponibles.

**Output** El resultado de la tarea es el conjunto de Metas de usabilidad del proyecto

#### *Pasos*

La tarea consta de cuatro pasos:

**Obtener información de Modelos disponibles** Tanto el Modelo de Usuario como el de Tareas contienen información que contribuye a generar metas de usabilidad. Por ejemplo permiten identificar la importancia relativa de metas sobre facilidad de aprendizaje versus facilidad de uso o la preeminencia de la performance sobre la satisfacción. El Modelo de Tareas revela las tareas claves del usuario sobre las cuales será conveniente establecer metas cuantitativas. Las metas de negocio, contenidas en documento de visión y requerimientos funcionales también aportan información para identificar y definir metas de usabilidad.

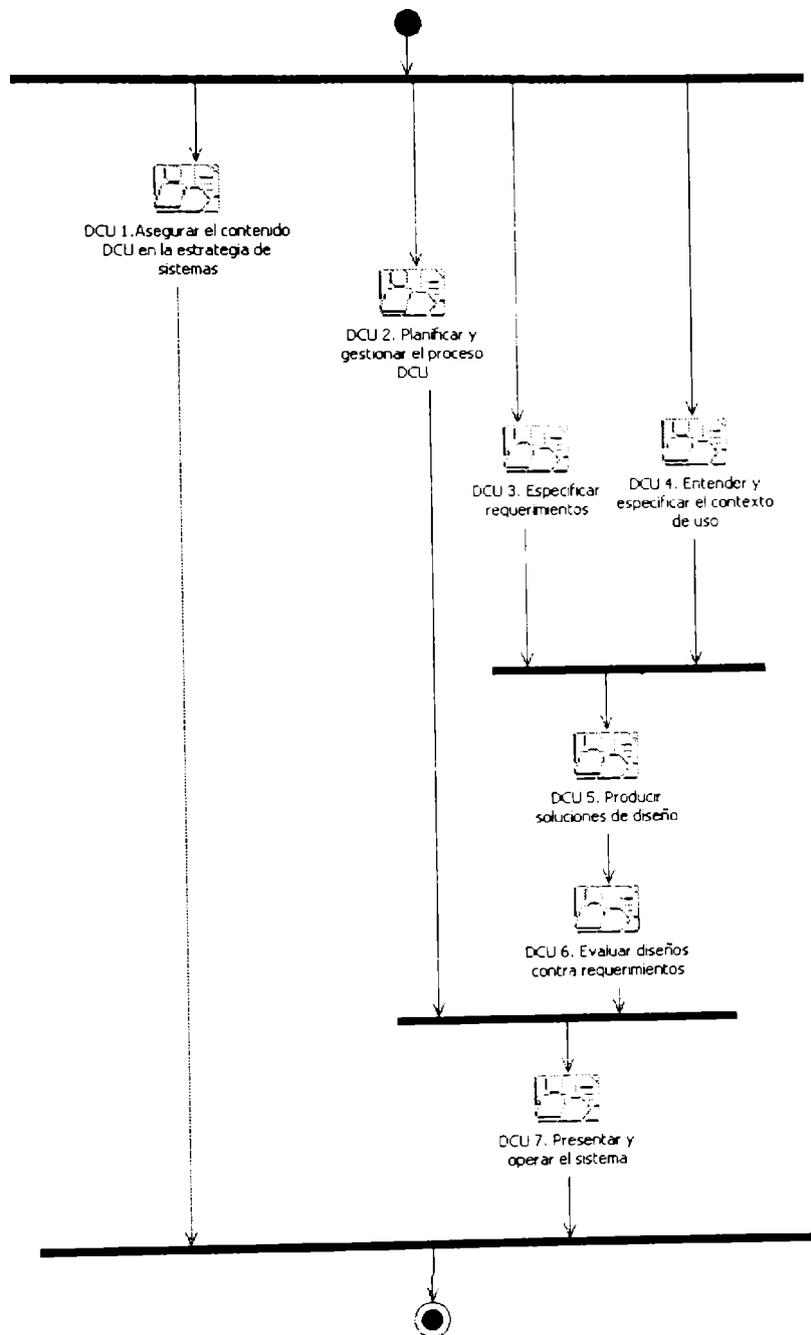


Figura 7.23: Ciclo de Vida de MMU-ISO

**Identificar y bocetar metas de usabilidad** La información del paso anterior revela metas de usabilidad candidatas, tanto de tipo cualitativo como cuantitativo. Estas deben ser identificadas y enunciadas en borrador a medida que son descubiertas. En un paso posterior se realizará una descripción precisa.

**Priorizar metas de usabilidad** Es fácil construir una larga lista de metas de usabilidad, tanto genéricas como específicas para un proyecto, aunque no todas sean alcanzables simultáneamente. Se hace necesario por tanto, establecer algún orden de prioridades para el equipo de desarrollo. Las metas identificadas deben clasificarse al menos en: a) requeridas, b) importantes pero no críticas, c) deseables.

**Documentar las metas de usabilidad** Todas las metas priorizadas deben ser documentadas con el producto de trabajo respectivo y conocidas por todo el equipo de trabajo.

**Revisar y consensuar metas de usabilidad** Aunque la participación adicional de stakeholders y analistas supone un acuerdo sobre las metas identificadas y priorizadas, es necesario realizar una revisión final que asegure el compromiso de la gestión del proyecto con los requerimientos de usabilidad especificados.

### **Tarea: Entender y especificar el contexto de uso**

La comprensión adecuada del contexto en el que utilizará el sistema y la especificación de los requerimientos que se derivan del mismo es imprescindible en un proceso centrado en usuarios.

#### ***Propósito***

Todo sistema en desarrollo se utilizará en un contexto determinado. Será utilizado por una población de usuarios con ciertas características, tendrán ciertas metas y desearán realizar algunas tareas específicas. El sistema podrá emplearse dentro de un rango de condiciones técnicas, físicas, sociales y organizacionales que a su vez podrán afectar su utilización. La calidad de uso del sistema, incluyendo la usabilidad, salud y seguridad del usuario dependerá entre otras cosas de haber comprendido bien este contexto de uso.

La extensión de esta Tarea podrá depender del conocimiento que se tenga del tipo de sistema en desarrollo. Para sistemas bien comprendidos, será suficiente identificar los involucrados y acordar reuniones para revisar el contexto de uso. En otros casos, será necesario complementar estas reuniones con un análisis de las tareas y un estudio más riguroso de los usuarios.

#### ***Relaciones***

**Roles** El rol responsable de la tarea es el Especialista en UX, aunque tanto el Representante de Usuarios como el Líder técnico del dominio pueden tener una participación secundaria.

**Input** El input para esta tarea lo constituyen todos los artefactos que proveen alguna información sobre los diferentes contextos de uso del sistema (las tareas del usuario, el entorno organizacional y físico, los atributos del usuario) como el documento Visión y Reporte de investigación de usuarios.

**Output** El resultado principal de esta tarea es el Modelo de tareas, un producto de trabajo que contiene la descripción de los diferentes contextos de uso. Opcionalmente, la tarea de comprender cabalmente el contexto de uso puede derivar en modificaciones de artefactos de otras prácticas tales como la Visión y el Glosario.

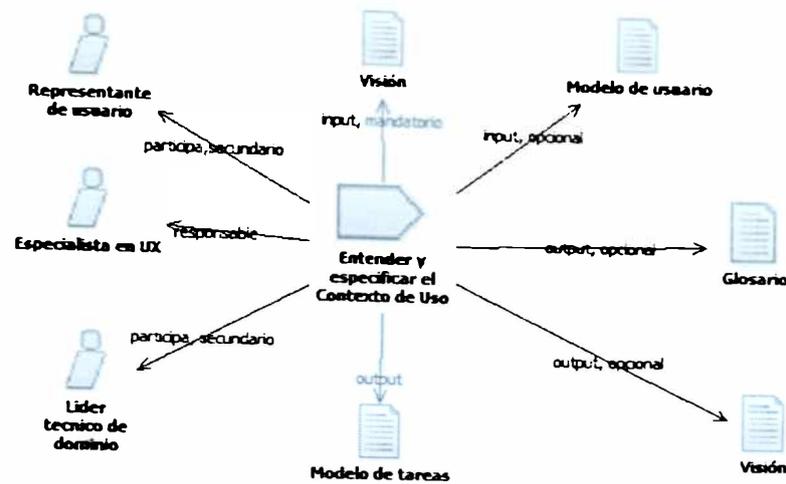


Figura 7.24: Definición de la Tarea Identificar y Especificar el contexto de uso

### Pasos

La tarea consta de cinco pasos básicos:

**Identificar y documentar las tareas del usuario** Describir las actividades que los usuarios realizan para alcanzar sus metas. Las descripciones de tareas no incluyen solamente las funciones o características del equipamiento. Las tareas pueden cambiar o evolucionar durante el ciclo de vida del sistema.

**Identificar y documentar los atributos de todos los involucrados** Identificar y analizar los roles de cada grupo de usuarios e involucrados que serán afectados por el sistema. Valorar la significación y relevancia del sistema para cada grupo. Describir las características relevantes de los usuarios finales del sistema, incluyendo conocimientos, lenguaje, capacidades físicas, nivel de experiencia, etc.

**Identificar y documentar el entorno organizacional** Describir el marco social y organizacional, la estructura y prácticas de gestión, etc.

**Identificar y documentar el entorno técnico** Describir las características relevantes de cualquier equipo que sea utilizado. Para sistemas nuevos las características del equipo dependen de las soluciones a proponer, por lo que es posible que no sean conocidas hasta bastante avanzado el proyecto.

**Identificar y documentar el entorno físico** Describir la ubicación, equipamiento del puesto de trabajo y condiciones ambientales.

### Orientaciones

Algunas de las técnicas y métodos que pueden emplearse para realizar esta Tarea se describen brevemente a continuación.

**Identificar y documentar los atributos de todos los involucrados** Es importante identificar todos los usuarios y otros involucrados que se verán impactados por el sistema. Estos grupos pueden incluir desde usuarios finales, supervisores, instaladores, mantenedores hasta quienes sólo reciban los outputs del sistema, la gente de marketing, compradores, gente de soporte, etc. Usualmente, este paso puede resolverse mediante reuniones entre el Gerente del Proyecto, el Especialista en Experiencia de Usuario, el Líder técnico y el Representante de usuarios.

**Análisis de contexto de uso** Método estructurado para elicitación de información detallada del contexto de uso como base para actividades de usabilidad posteriores. Los roles de Stakeholders participan en una reunión de Contexto para contribuir a completar un cuestionario detallado. Es una técnica simple para utilizar cuando la información relevante del contexto es conocida por los Stakeholders. Mayor información se puede encontrar en [Holtzblatt et al. 2005].

**Encuesta de usuarios** Las encuestas pueden ayudar a determinar las necesidades de usuarios, sus prácticas y actitudes hacia las nuevas ideas de un sistema. Permiten obtener tanto datos cualitativos como cuantitativos de una cantidad importante de usuarios. Consultar por ejemplo [Kuniavsky 2003].

**Estudios y observaciones de campo** Los métodos observacionales involucran un investigador mirando los usuarios mientras trabajan y tomando notas de la actividad mientras ésta tiene lugar. Algunos ejemplos de estos métodos se encuentran en [Shepherd 2001] y [Stanton 2004].

**Análisis de tareas** Puede definirse como el estudio de lo que se requiere que un usuario haga en términos de acciones y/o procesos cognitivos para realizar una tarea. El análisis detallado de las tareas, sobre todo en los casos de actividades muy complejas o no soportadas hasta el momento por sistemas de software es importante para determinar entre otras cosas una correcta distribución de responsabilidades entre las personas y la tecnología. Existen muchas variaciones de métodos para realizar análisis de tarea como el Análisis Jerárquico [Stanton 2006]. Un buen compendio de técnicas y métodos se encuentra en [Stanton 2004].

### **Tarea: Diseñar la Experiencia del Usuario**

Esta tarea consiste en realizar las acciones necesarias para proponer y crear soluciones a las situaciones problemáticas planteadas por los requerimientos del sistema para la futura experiencia de usuario empleando prácticas de diseño del estado del arte. Se trata de crear y desarrollar el Concepto de Experiencia de Usuario que se propone para el sistema o producto. Debe adoptar una visión general del producto y proponer las principales ideas que estructuran la interacción del usuario con el sistema, los conceptos del dominio involucrados, las restricciones que se plantean sobre la arquitectura del software para facilitar la usabilidad final, la arquitectura de la información, etc.

#### *Relaciones*

**Roles** El encargado responsable de la tarea es el Diseñador de la Experiencia de usuario, con aportes adicionales del Especialista en UX y todos los roles de usuarios.

**Input** Para desarrollar esta tarea es necesario contar con todas las [Especificaciones técnicas] y conocer las propuestas de arquitectura de software planteadas en el Cuaderno de Arquitectura.

**Output** El producto principal de la tarea es el Concepto de la UX, pero en sucesivas iteraciones también produce o modifica cualquier otro [Diseño de Interacción].

#### *Pasos*

Una tarea de diseño no puede prescribirse en una serie fija, lineal y ordenada de pasos. Lo que indicamos a continuación es el conjunto de acciones que el proceso DCU de Diseño de la experiencia de usuario debe contener. El orden, extensión e intensidad de cada paso estará relacionado con la fase e iteración del proceso de desarrollo y la experiencia de los diseñadores con el dominio del problema.

**Alocar funciones** Consiste en distribuir de manera conciente y explícita las responsabilidades de acciones entre los usuarios y la tecnología de soporte (el software y las condiciones organizacionales). Esta asignación debe tener en cuenta las capacidades y limitaciones de cada uno de ellos para obtener la performance apropiada. Este paso busca optimizar la performance del sistema en su conjunto para el logro de las metas.

**Producir el modelo de tareas compuesto** Desarrollar un modelo factible para nuevas tareas del usuario a partir del conocimiento de las mejores prácticas, los requerimientos, el contexto de uso, la asignación de funciones y las restricciones al diseño del sistema.

**Explorar el diseño y generar alternativas** Generar y estudiar varias opciones de diseño para cada aspecto del sistema relacionado con su utilización y su efecto en todos los Stakeholders.

### **Tarea: Diseñar componentes de interacción**

Esta tarea es consiste realizar las acciones necesarias para proponer y crear soluciones a las situaciones problemáticas planteadas por los requerimientos del sistema para la futura experiencia de usuario empleando prácticas de diseño del estado del arte.

#### **Relaciones**

**Roles** Como es de suponer, el Diseñador de la Experiencia de Usuario es el responsable de la tarea. Sin embargo, respondiendo a la naturaleza del proceso que estamos definiendo, también los roles de Stakeholders podrán tener alguna participación en el diseño. Finalmente, el Diseñador de UX podrá dar participación al Desarrollador para asegurar que sus propuestas sean implementables.

**Input** Todos los elementos que han sido construidos para especificar requerimientos constituirán el input de esta tarea: Visión, Glosario, Modelo de tareas, Modelo de Casos de uso y el Concepto de UX.

**Output** El resultado por antonomasia de esta tarea será el Prototipo de UX, extendido por el Storyboard y el Mapa de navegación. No debe descartarse que esta tarea haga propuestas de modificaciones a alguno de los artefactos que sirvieron de input, especialmente al Modelo de Tareas y Modelo de Casos de uso. Estas modificaciones podrán quedar sujetas a la ejecución de la Tarea Revisar el diseño de la UX.

La figura 7.25 presenta las relaciones de la tarea con otros elementos de la Práctica.

#### **Pasos**

**Explorar el diseño y generar alternativas** generar y estudiar varias opciones de diseño para cada parte del sistema que estará en contacto con los usuarios .

**Desarrollar la solución de diseño** Este paso es el núcleo de la Tarea. Se trata aquí de aplicar todo el conocimiento relevante sobre la gente y la tecnología y los diferentes requerimientos especificados para el proyecto y generar soluciones específicas para la interacción de usuario con cada uno de los componentes del sistema. Cada solución se concretará en simulaciones, modelos, maquetas o cualquier otra forma de Prototipo que resulte apropiada al proyecto.

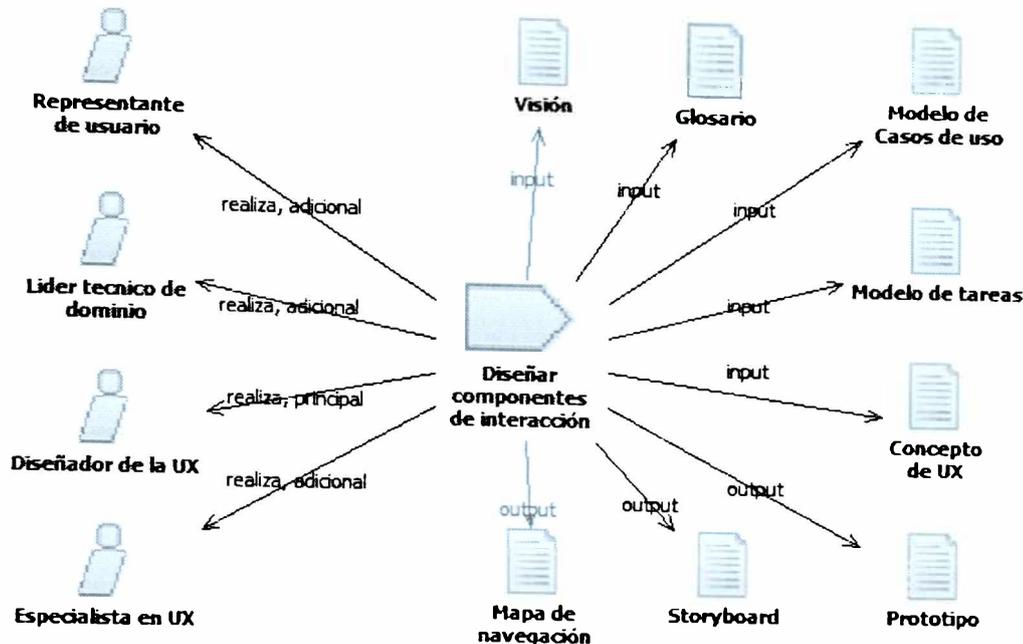


Figura 7.25: Definición de la Tarea Diseñar la UX

**Especificar el sistema** Las soluciones que se modelen y resulten adecuadas para someterlas a evaluación deberán ser especificadas de modo que las ideas y propuestas incluidas puedan ser implementadas en el resto del proyecto. La especificación será diferente dependiendo de la Fase e Iteración del proyecto. En las primeras iteraciones de Inicio o de Elaboración es probable que la especificación necesaria sea solamente aquella que permita al Tester de la UX realizar inspecciones heurísticas. Sin embargo, a medida que las soluciones se detallan, será necesario incluir especificaciones orientadas a los desarrolladores.

#### Tarea: Revisar el diseño de la Experiencia del usuario

En esta tarea se analiza las soluciones de diseño propuestas con para dar feedback temprano sobre aspectos de usabilidad, en particular, y experiencia de usuario, en general. Sobre la base del conocimiento de capacidades y limitaciones humanas, el Tester de UX analiza los prototipos, storyboards y cualquier otro artefacto realizado y devuelve feedback en las etapas iniciales para evitar que se avance en el desarrollo de soluciones erróneas.

#### Relaciones

**Roles** La tarea es responsabilidad directa del Tester de UX que se basará en sus conocimientos de HCI, usabilidad, accesibilidad, interacción, etc para analizar las soluciones y proveer feedback. Para aumentar la riqueza de la revisión se podrá requerir la colaboración del Especialista en UX, del Líder técnico de dominio y el Representante de usuarios.

**Input** Esta tarea se alimenta en forma obligatoria de las soluciones de diseño entregadas por el Diseñador de UX en cualesquiera de los formatos disponibles (Concepto de UX, Prototipo, Storyboard, Mapa de navegación). Adicionalmente es conveniente que se consulten las especificaciones de requerimientos tales como la Visión, el Glosario, el Modelo de Casos de Uso y el Modelo de Tareas.

**Output** El resultado esperable son las sugerencias de modificaciones a los productos de trabajo que se tomaron como input: Prototipo de UX, Storyboard, Mapa de navegación.

La figura 7.26 muestra que la Tarea se constituye en uno de los nodos centrales del Proceso DCU (junto con Diseñar la Experiencia del usuario), por cuanto permite la participación de los diferentes roles representantes de usuarios y aborda la revisión de la mayoría de los Productos de Trabajo incorporados en el proceso de diseño.

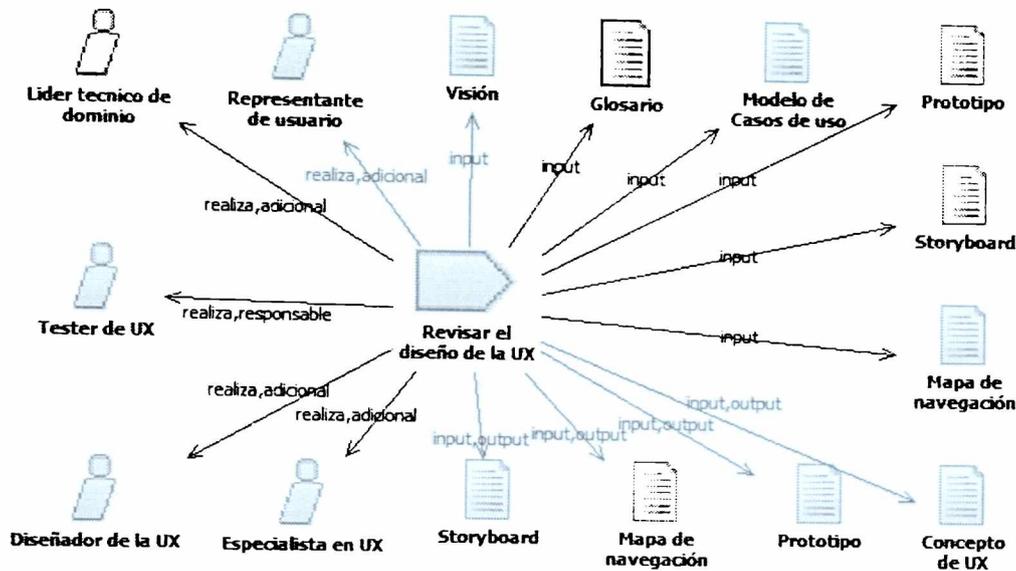


Figura 7.26: Definición de la Tarea Revisar el Diseño de la Experiencia de usuario

### *Pasos*

**Especificar y validar el contexto de evaluación** Es muy importante tener claras las diferencias entre las condiciones del contexto de evaluación y el contexto de uso especificado por el producto de trabajo correspondiente. Este paso asegura una buena gestión de la evaluación.

**Evaluar prototipos iniciales para mejorar el diseño** Utilizando la experiencia y conocimientos previos de tester, realizar pruebas de inspección sobre los Prototipos, Storyboard y Mapa de navegación. Se deberá revisar que en el diseño se han seguido las buenas prácticas de diseño de interacción, se han empleado adecuadamente los conocimientos de HCI disponibles, se utilizaron los activos del proyecto o la organización tales como guías de estilo, estándares de referencia, legislación vigente, etc.

**Evaluar el sistema para verificar que los requerimientos se cumplan** Testear las versiones en desarrollo o final del sistema para asegurar que cumple los requerimientos de los usuarios, las tareas y el entorno como fueron definidos.

**Presentar los resultados** Luego de realizadas las inspecciones y las pruebas contra requerimientos y prácticas, deberá presentarse al equipo de diseño los resultados de la manera más apropiada. Esta presentación incluirá además de los errores y falencias encontrados, las sugerencias de cambios para superarlos.

### *Orientaciones*

En la literatura de diseño de interacción existen tres abordajes principales para la evaluación de propuestas: a) pruebas de usabilidad, b) estudios de campo y c) evaluaciones analíticas. Cada uno de estos enfoques puede implementarse con diferentes métodos como observar usuarios, encuestar usuarios, encuestar expertos, probar con usuarios, inspeccionar el sistema, modelar la performance del usuario, etc.

Los dos primeros abordajes son utilizados dentro de la Práctica DCU en las tareas Preparar pruebas de usabilidad y Ejecutar pruebas de usabilidad. En esta Tarea utilizamos el enfoque de evaluaciones analíticas.

Las evaluaciones analíticas emplean básicamente dos categorías de métodos: las inspecciones y los modelos basados en teorías. Ambos tienen como característica central que no es necesario contar con la presencia de usuarios en las pruebas.

Las inspecciones pueden consistir en el chequeo de reglas heurísticas donde se aplica conocimiento previo de usuarios típicos para identificar problemas de usabilidad [Nielsen 1993] o recorridos, donde un experto “pasea” a través de escenarios con prototipos del sistema.

Las heurísticas se basan en conocimiento de sentido común y guías de usabilidad (tales como proveer un lenguaje familiar al usuario). Se desarrollaron originalmente para sistemas de pantallas, pero luego han sido adaptadas a diferentes sistemas basados en software.

Los recorridos cognitivos consisten en simular el proceso de resolución de problemas del usuario para cada paso del diálogo propuesto con el sistema y chequear el progreso paso a paso en las interacciones. Se suelen emplear para evaluar la facilidad de aprendizaje de un sistema. [Rieman et al. 1995]

Los modelos teóricos se han empleado principalmente para comparar la eficacia de diferentes interfaces de la misma aplicación y la disposición óptima de controles. Algunos de los principales modelos que se pueden mencionar son la Ley de Fitts, modelos de tecleo como GOMS, etc.

### **Tarea: Preparar pruebas de usabilidad**

Consiste en preparar los casos y datos necesarios para realizar pruebas que involucren activamente a usuarios para evaluar la satisfacción de requerimientos y objetivos de usabilidad planteados para el proyecto.

#### ***Propósito***

El principal propósito de las pruebas de usabilidad consiste en obtener feedback de usuarios finales (o una muestra representativa de ellos) sobre el logro de los objetivos planteados para la experiencia del usuario.

#### ***Relaciones***

**Roles** El responsable de la Tarea es el Tester de la experiencia de usuario. En la creación de las pruebas pueden colaborar como participantes secundarios el Especialista de UX (aportando su experiencia y conocimientos en interacción y usabilidad) y el Representante de usuarios (indicando tareas y datos que resulten significativos para los usuarios).

**Input** Como datos de entrada para la preparación de las pruebas se emplearán todas las especificaciones de requerimientos disponibles tanto como las propuestas de diseño que pueda haber.

**Output** El resultado de esta tarea será uno o más Casos de prueba orientados a los aspectos de usabilidad.

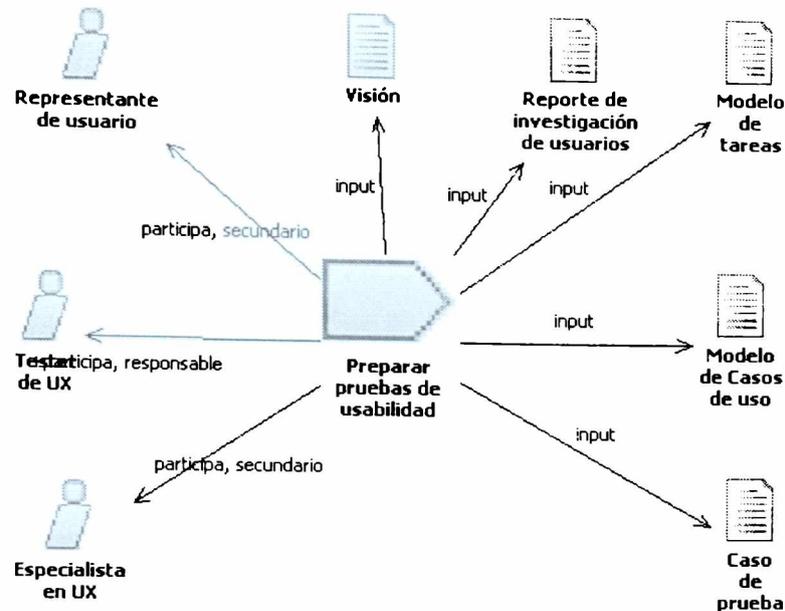


Figura 7.27: Definición de la Tarea Ejecutar pruebas de usabilidad

### *Pasos*

**Decidir los objetivos de UX a probar** El primer paso consiste en priorizar y elegir entre los requerimientos y objetivos de UX planteados para el proyecto, cuáles serán incluidos en las pruebas. Por ejemplo, si en un proyecto es más importante asegurar la facilidad de uso que la eficiencia o performance, las pruebas deberán focalizarse en aquellas tareas y características del sistema que permitan dar feedback a los desarrolladores sobre ese objetivo.

**Identificar tipo y rango de usuarios a incluir** Este paso es necesario cuando existen diferentes tipos de Stakeholder: Usuario final. En este caso habrá que decidir cuáles deberán participar en cada una de las pruebas que se prepare.

**Diseñar la tarea a probar** Las tareas que servirán de base a la prueba podrán ser extraídas (y adaptadas en la medida que resulte necesario) de las diferentes especificaciones de requerimientos: Modelo de Tareas, Modelo de Casos de Uso, Escenarios. Los roles Representante de usuarios y Usuario final son aliados claves en este paso para conseguir una adaptación significativa de las tareas.

**Diseñar la prueba y desarrollar los materiales necesarios** Es importante planificar cuidadosamente la secuencia de eventos para la prueba y la elaboración de todos los materiales necesarios. Dependiendo del alcance de la tarea y la rigurosidad de la prueba los elementos a considerar son:

- Brief del observador
- Guión de prueba incluyendo bienvenida e introducción
- Encuesta previa a la prueba
- Entrenamiento previo en el uso del material (por ejemplo, los prototipos)
- Formularios de consentimiento para grabaciones de audio y/o video
- Enunciados de tareas de prueba
- Planillas para la recopilación de datos

- Encuesta posterior a la prueba
- Plantillas para la presentación de resultados del análisis

**Diseñar y ensamblar el entorno de prueba** Si no se dispone de un laboratorio de usabilidad para hacer las pruebas (o la naturaleza de la tarea requiere una prueba en otro tipo de entorno), será necesario definir todos el equipamiento a utilizar así como el diseño del ambiente físico. En las orientaciones se sugieren fuentes de consulta sobre el diseño de este ambiente.

**Reclutar usuarios** Este paso consiste en el acuerdo de participación de los usuarios finales en las pruebas, así como el armado de un cronograma estimado durante el cual ellos estarán disponibles para realizarlas.

### *Orientaciones*

Las pruebas de usabilidad constituyen uno de los ejercicios clásicos en cualquier proceso de desarrollo centrado en usuario. Existe en la literatura un gran número de obras que describen sus fundamentos y el modo de realizarlas. Como ejemplo, a la hora de configurar un proceso DCU se podrá tomar como referencia las siguientes obras:

- The usability engineering lifecycle por Deborah Mayhew, págs 342-346 [Mayhew 1999]
- Interaction design por Sharp, Rogers y Preece, Capítulo 14 [Sharp et al. 2007]
- Observing the user experience por Mike Kuniavsky. Capítulo 10 [Kuniavsky 2003]

### **Tarea: Ejecutar pruebas de usabilidad**

Consiste en la realización completa de las pruebas preparadas, el análisis de los resultados y su comunicación al equipo de trabajo

#### *Propósito*

Proveer feedback al equipo sobre el nivel de logro de los requerimientos y objetivos planteados para el proyecto en aspectos vinculados con la experiencia del usuario, particularmente con la usabilidad del sistema.

#### *Relaciones*

**Roles** El responsable es el Tester de la experiencia de usuario, pero debe participar activamente tanto el Usuario final como el Representante de Usuarios. Adicionalmente podrán colaborar el Especialista de la UX para cubrir los roles de moderación previstos en las pruebas y para contribuir al análisis de los datos obtenidos.

**Input** Esta tarea se alimenta de los Casos de Prueba de Usabilidad generados en la Tarea: Preparar pruebas de usabilidad

**Output** El resultado es un registro de la prueba y los análisis efectuados sobre los datos obtenidos. Normalmente consistirá en versiones modificadas y comentadas de las propuestas de diseño empleadas al preparar la prueba: Prototipo, Storyboard, Mapa de Navegación

La figura 7.28 muestra las relaciones entre la Definición de la Tarea Ejecutar pruebas de usabilidad y los otros elementos de Método.

#### *Pasos*

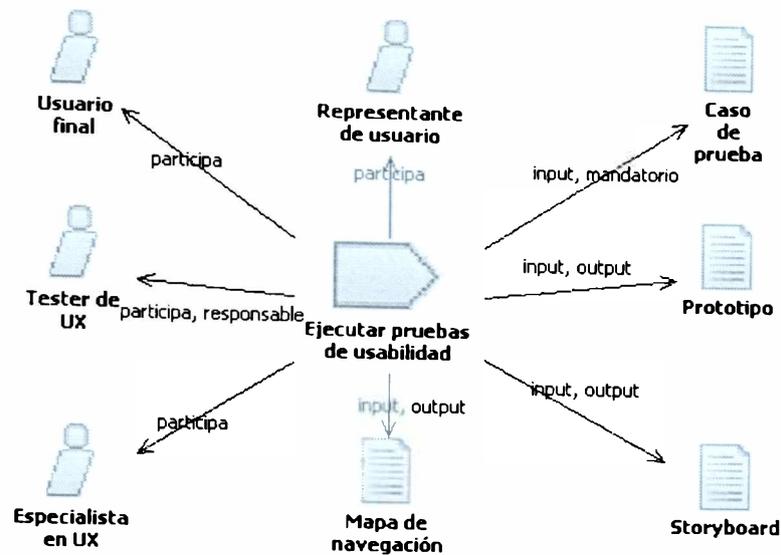


Figura 7.28: Definición de la Tarea Ejecutar pruebas de usabilidad

**Setear el entorno y condiciones de prueba** Establecer las condiciones para ejecución de la prueba tal como fue planificada en la Tarea: Preparar pruebas de usabilidad. Este paso no solo implica el entorno físico de pruebas, sino también asegurar que están disponibles tanto los usuarios como los colaboradores y el material requerido según el guión de la prueba.

**Ejecutar la prueba y recopilar los datos** Correr el procedimiento de prueba tal como fue planificado. El ciclo habitual será: dar la bienvenida al usuario, realizar las encuestas previas, conseguir las autorizaciones de registro pertinentes, presentar la tarea al usuario (con un pequeño entrenamiento en el manejo de los prototipos si la tarea lo requiere), dejar que el usuario complete la tarea sin influirlo ni guiarlo con los registros adecuados, reunir todos los datos obtenidos, recolectar los datos post-prueba que sean necesarios y volver a comenzar con otro usuario.

**Analizar e interpretar los datos obtenidos** Concentrarse en las áreas donde los datos mostraron que los objetivos o requerimientos de aceptación no se alcanzan. Tratar de interpretar estas fallas en términos de cambios de interacción, diseño de pantallas y también en el modelo de las tareas propuesto por los Casos de Uso.

**Sacar conclusiones y formular recomendaciones de cambio** Utilizar los datos para extraer conclusiones respecto de cada uno de los problemas identificados y proponer cambios para eliminarlos.

**Documentar y presentar resultados** Siempre que sea posible (sobre todo si los otros miembros del equipo no están presentes durante la prueba) es conveniente realizar una presentación de resultados y recomendaciones en forma oral apoyada por algún tipo de presentación de las grabaciones obtenidas. Adjuntar un reporte con los problemas identificados y los cambios que se recomiendan.

#### **Tarea: Diseñar y producir material de entrenamiento y soporte al usuario**

Un proceso de desarrollo que se enfoque en el usuario debe incluir una tarea que consista en el diseño y producción de materiales para el soporte y entrenamiento de los usuarios durante la implantación, puesta en producción o liberación del sistema.

### Propósito

Esta tarea tiene por objetivo asegurar que se brinda apoyo a los diferentes tipos de usuarios involucrados en la etapa de implementación (usuarios finales, encargados de mantenimiento, instaladores, etc).

### Relaciones

**Roles** El responsable de la tarea es el Especialista en UX aunque deberá contar con la colaboración directa del Líder técnico de dominio y del Representante de usuarios, especialmente en el material de entrenamiento.

**Input** Los materiales que esta tarea utiliza como insumos son todos los productos que describen los requerimientos del sistema y los planes de la iteración.

**Output** El resultado será uno o más Documento para soporte y entrenamiento de usuario.

La figura 7.29 muestra las relaciones entre esta tarea y otros elementos de la Práctica DCU.

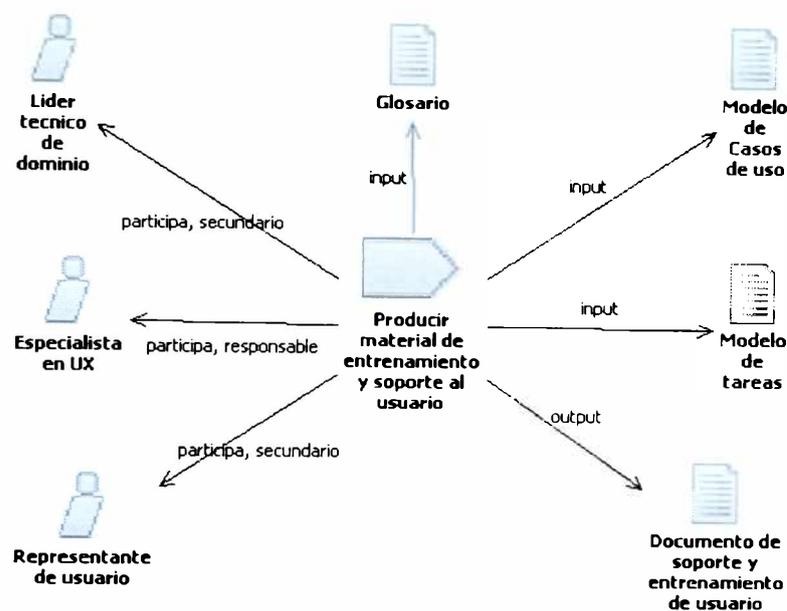


Figura 7.29: Definición de la Tarea Producir material de entrenamiento y soporte al usuario

### Pasos

El materia de soporte y entrenamiento para usuarios finales pueden llegar a constituir proyectos o aplicaciones en sí mismos. Sin llegar a ese punto, un proceso DCU debe contemplar clara y explícitamente tanto el entrenamiento y capacitación de usuarios a fin de facilitar la apropiación del nuevo sistema, como el soporte para la personalización del sistema allí donde las necesidades específicas de algún grupo de usuarios lo requiera.

Esta tarea deberá contemplar al menos:

**Definir los objetivos del entrenamiento y soporte** Considerar la audiencia a capacitar y decidir las mejores estrategias (manuales y tutoriales en línea, cursos de capacitación, prácticas y ejemplos guiados, etc.) y los objetivos para cada grupo de usuarios involucrado

**Diseñar material de entrenamiento y soporte** Realizar el diseño de todo el material necesario para que cada grupo identificado pueda alcanzar los objetivos planteados de entrenamiento y disponga del soporte adecuado para el uso y configuración del sistema.

**Realizar material de entrenamiento** Producir el material diseñado y liberarlo de acuerdo con el plan de despliegue del proyecto.

### Tarea: Guiar y dar soporte al usuario en implementación

Brindar entrenamiento y soporte a los usuarios, transferir el conocimiento del funcionamiento del sistema y nuevo diseño de tareas donde sea necesario.

#### Relaciones

**Roles** El encargado de esta tarea es el Especialista en UX, con la colaboración de todos los Stakeholders incorporados al equipo. En las configuraciones posteriores veremos que otros miembros del equipo (Gerente de Proyecto y Analista pueden ser asignados a esta tarea)

**Input** La tarea se nutre de los materiales de capacitación y configuración (Documento para soporte y entrenamiento de usuario) generados en la tarea anterior.

**Output** La tarea no produce resultados entregables, sino la capacitación y capacidad de configuraciones y adaptaciones por parte de los usuarios.

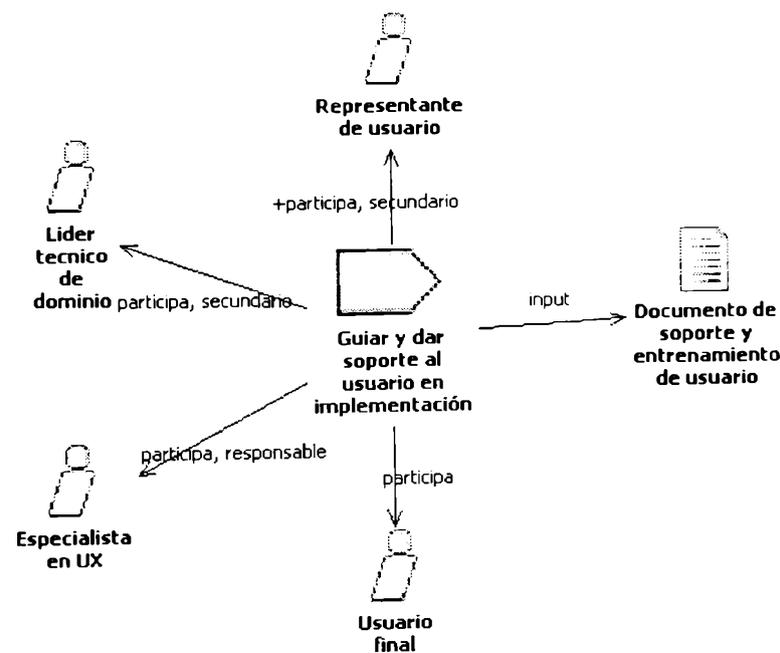


Figura 7.30: Definición de la Tarea Guiar y dar soporte al usuario en implementación

#### Pasos

La secuencia de pasos a ejecutar dependerá del esquema de capacitación y entrenamiento que se haya elaborado.

### 7.2.4. Orientaciones

La Práctica DesCU debe estar integrada por Orientaciones que contribuyan al ingeniero de procesos y los miembros del equipo a comprender y mantener el foco en las posibilidades y limitaciones de la práctica.

En particular proponemos la integración de dos orientaciones básicas. La primera contiene las definiciones de los Procesos DCU y la segunda, los niveles del MMU-ISO. A modo de hoja de ruta, ambas dan guía al ingeniero de proceso para la adopción total o parcial de la Práctica y su inclusión en los procesos de desarrollo.

El contenido de la primera Orientación:Definiciones de Procesos DCU corresponde al Apéndice A, mientras que el Apéndice B incluye la descripción completa de los atributos de Niveles de Capacidad para todo el MMU-ISO.

## 7.3. Configuraciones de Método: OpenUP/MMU-ISO

La implementación de la Práctica Desarrollo Centrado en Usuario en un proceso basado en OpenUP asegura la inclusión básica de todos los procesos DCU de MMU-ISO. En consecuencia, una extensión de la configuración OpenUP/Basic que incluya esta nueva Práctica alcanzaría para dar cobertura al Nivel 1 del MMU-ISO.

Sin embargo, esto no asegura que se logren los otros dos niveles que aspiramos a soportar. Para saldar esta necesidad proponemos completar un esquema de tres configuraciones de prácticas que puedan ser instanciadas para alcanzar cada uno de los niveles de capacidad en Nivel 1, 2 o 3 del MMU-ISO.

En líneas generales, entonces, podemos señalar que el Nivel 1 se alcanzará con una configuración que incluya la Práctica Desarrollo Centrado en Usuario tal como fue presentada en la sección anterior. Para los Niveles 2 (gestionado) y 3 (establecido) utilizaremos la variabilidad “extiende” sobre el Nivel 1 y 2, respectivamente con el objetivo de incorporar las actividades de gestión de proceso y producto y de adaptación de los métodos requeridas, tal como hemos visto en el capítulo anterior.

En los tres casos, se trata de adaptaciones a configuraciones existentes. Partimos de la configuración publicada como OpenUP/Basic en su release 1.5.0.1. Sobre ésta elaboramos OpenUP/MMU-ISO N1. OpenUP/MMU-ISO N2 se construye como extensión de N1 y lo mismo ocurre con N3 respecto de N2.

A continuación presentamos el contenido de estas adaptaciones, tomando como vista de navegación los Procesos de Despliegue de OpenUP/Basic que responden al ciclo de vida basado en la Práctica de Ciclo Riesgo-Valor con las cuatro fases iterativas del Proceso Unificado: Inicio, Elaboración, Construcción y Transición.

### 7.3.1. OpenUP/MMU-ISO N1

En esta configuración incluimos los cambios y extensiones necesarias para dar cobertura amplia al Nivel 1 del Modelo de Madurez de Usabilidad ISO.

En cada una de las Fases (Inicio, Elaboración, Construcción, Transición) describimos el flujo de actividades base, destacando aquellas que sufren alguna alteración y detallando las extensiones o cambios que proponemos.

#### FASE DE INICIO

El objetivo de esta fase es entender el alcance del problema y la factibilidad de una solución. El proceso de despliegue para una iteración típica de esta Fase en nuestra configuración contiene las mismas cuatro actividades base: Iniciar el proyecto, Identificar y

Refinar requerimientos, Acordar sobre el enfoque técnico y Planificar y Gestionar la Iteración, tal como se muestra en la figura 7.31, aunque, como veremos a continuación, con varios cambios al interior de cada actividad.

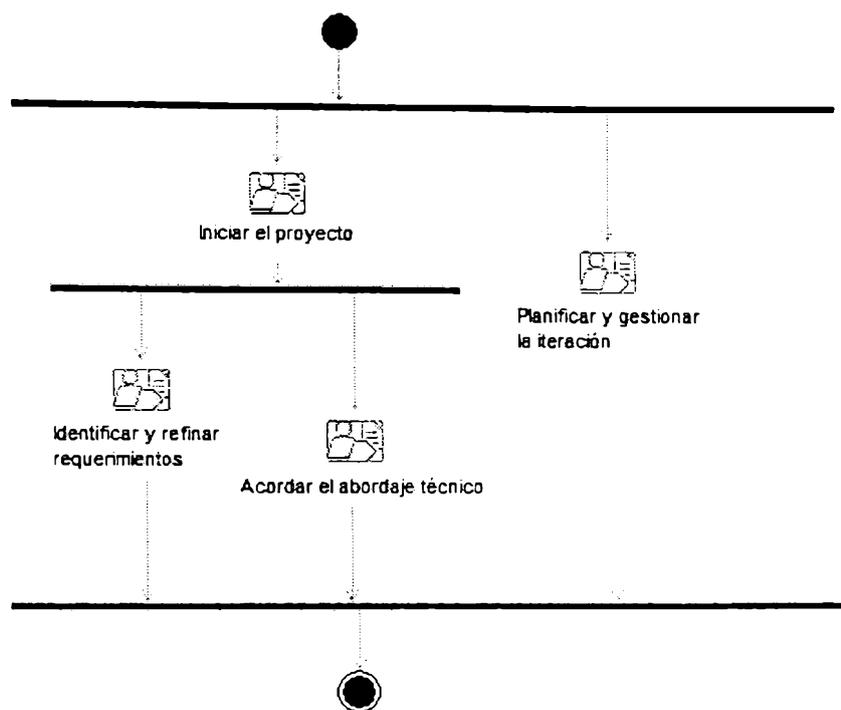


Figura 7.31: Iteración típica de Fase de Inicio con DesCU

**Actividad: Iniciar el proyecto** Esta actividad tiene lugar al comienzo de la primera iteración, cuando comienza el proyecto y su meta es establecer una visión de alto nivel tanto del proyecto como del plan. Incluye dos tareas como se muestra en la Figura 7.32: **Desarrollar la visión técnica** (donde el Analista y el Stakeholder trabajan juntos para obtener una visión del proyecto) y **Planificar el proyecto** (en la que el Gerente de Proyecto propone un plan de alto nivel con los hitos de cada una de las cuatro fases y un primer cronograma de iteraciones).

Para este Nivel de Capacidad, los cambios en la Actividad Iniciar el Proyecto buscan tres objetivos: a) dar lugar a los Roles específicos de DCU, b) poner énfasis en la consideración de usuarios en cada uno de los pasos para garantizar su centralidad desde el comienzo en concepción del sistema y esbozo de plan y c) sembrar adecuadamente el terreno para el trabajo posterior (por ejemplo, en la Tarea Desarrollar Visión Técnica agregaremos el paso Identificar y bocetar el contexto de uso para permitir que haya desde el comienzo material adecuado para la Tarea posterior Entender y especificar contexto de uso, que incluiremos en las actividades de requerimientos).

A continuación detallamos los cambios incluidos en esta Actividad.

**Tarea: Desarrollar Visión Técnica** Se incluyen dos roles DCU, se agrega un paso relacionado al contexto de uso y se refina el Template del documento Visión:

- debe contar con la participación activa del Líder técnico del dominio y del Representante de usuarios, como especificación del Stakeholder que involucra la versión básica.

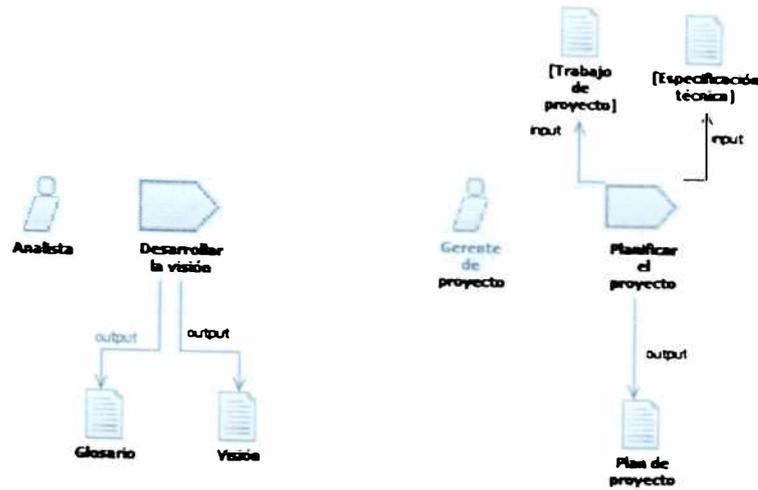


Figura 7.32: Actividad Iniciar el Proyecto en OpenUP/MMU-ISO N1

- se agrega el paso Identificar y Bocetar Contexto de uso con el objetivo de asegurar que se obtiene información pertinente para el Proceso ISO DCU4. Consiste en identificar, clarificar y registrar las características de todos los involucrados (usuarios finales e intermedios), sus tareas y el entorno físico y de organización en que se operará el sistema.
- se modifica el Template del Documento Visión, con el objetivo de ampliar la información contenida en la descripción de Stakeholders y en el Contexto de usuario. Se busca que estas secciones sirvan de input adecuado a las tareas posteriores de Identificar y especificar stakeholders y Entender y especificar Contexto de Usuario.

**Tarea: Planificar el Proyecto** Debe incluir la participación activa del Patrocinador y del Líder técnico del dominio para validar las ideas del Gerente del Proyecto.

Opcionalmente, en el segundo nivel de Planificación (ver Práctica: Planificación en dos niveles) se puede solicitar la participación del Representante de usuarios.

Por otra parte, la Práctica de Gestión: Planificar el Proyecto en dos niveles incluirá en el nivel 1 la participación de los usuarios mediante su inclusión con roles concretos dentro del equipo y en el nivel 2 definir los criterios de performance y productos de trabajo a incluir en cada una de las tareas de DCU.

**Actividad: Identificar y refinar requerimientos** Esta actividad describe las tareas para obtener, especificar, analizar y validar un subconjunto de requerimientos.

En virtud del carácter iterativo e incremental de OpenUP, es lógico que esta actividad esté incorporada en los procesos de despliegue de las fases Inicio, Elaboración y Construcción, aunque con enfoques diferentes. En la Fase de Inicio, el foco se coloca sobre la comprensión del problema a resolver, la identificación de las necesidades de stakeholders y el planteo de las características de alto nivel del sistema. Más adelante veremos que en las fases de Elaboración y Construcción el foco vira significativamente de la comprensión del problema a la definición y refinamiento de la solución.

La definición básica de la Actividad incluye cuatro tareas. Tres están a cargo del Analista: **Encontrar y bocetar requerimientos**, **Detallar escenarios de Casos de Uso** y **Detallar requerimientos globales**; el Tester será responsable de la cuarta: **Crear casos de prueba**.

En nuestra configuración proponemos un cambio importante en la composición de la actividad, específicamente en lo relacionado a determinar y especificar claramente todas las partes involucradas, el contexto de uso del sistema y la formulación de requerimientos y metas deseadas para la experiencia de usuario.

Desagregamos estos contenidos de las tareas **Encontrar y bocetar requerimientos** y **Detallar requerimientos globales** para dar lugar al Especialista en UX a identificar, especificar y detallar tanto las definiciones de cada una de las partes involucradas (stakeholders) y del contexto de uso (con la extensión y formalidad que el proyecto requiera) como los objetivos que se pretenden alcanzar para la experiencia del usuario con el sistema. El Analista concentrará en la Tarea: Encontrar y bocetar requerimientos el trabajo con los requisitos funcionales del sistema, mientras el Especialista en UX (también Analista) se enfocará en los temas que impactan directamente la interacción de usuario. El trabajo de ambos será incorporado en los modelos que contengan las [Especificaciones técnicas], como el Modelo de Casos de Uso y la Especificación de Requerimientos Globales.

Entonces, la configuración OpenUP/MMU-ISO N1 propone la Actividad: Identificar y refinar requerimientos con las siguientes tareas que se representan en la Figura 7.33:

**Tarea: Encontrar y bocetar requerimientos** Definida en OpenUP/Basic, la modificación que incluimos consiste en desagregar todos los requerimientos vinculados con la experiencia de usuario (usabilidad, navegación, interacción, etc.), sean o no funcionales. Esto afecta fundamentalmente a los pasos 3 (Identificar los tipos de requerimientos relevantes al sistema), 6 (Identificar y capturar requerimientos globales) y 7 (Lograr concurrencia) de la Tarea.

**Tarea: Entender y especificar el contexto de uso** El contenido de la Tarea ha sido descrito en párrafos anteriores.

**Tarea: Especificar metas de la experiencia de usuario** La Tarea fue descrita en párrafos anteriores y se incluye en esta configuración en su versión completa.

**Tarea: Detallar escenarios de casos de uso** Sin modificaciones respecto de OpenUP/Basic.

**Tarea: Detallar requerimientos globales** Los requerimientos relacionados con usabilidad dejan de ser incumbencia de esta tarea y pasan al ámbito de Especificar metas de UX. El resto de la tarea se realiza de la forma definida en OpenUP/Basic.

**Tarea: Crear casos de prueba** Sin modificaciones respecto de OpenUP/Basic.

**Actividad: Acordar el abordaje técnico** En esta Actividad se produce la primera aproximación a una solución de diseño en OpenUP. La meta es definir una aproximación técnica al sistema consistente con los requerimientos del proyecto y las restricciones conocidas. No se busca producir una especificación técnica completa y comprehensiva, sino algo de alto nivel que brinde las suficientes garantías de factibilidad al equipo (de hecho, habitualmente esta aproximación consiste en soluciones que ya han sido probadas en otros proyectos o dominios) y que pueda ser corroborada de alguna forma con todos los involucrados.

Es importante que desde estas primeras decisiones que determinarán las posibilidades y restricciones del sistema a la experiencia del usuario, se de participación a los especialistas en tales temas. Diversos autores han mostrado claramente la relación que existe entre las decisiones tempranas sobre la arquitectura del software y las posibilidades de alcanzar



buenos niveles de usabilidad, por ejemplo [Bass and John 2001a] y [Folmer and Bosch 2005].

Se propone la realización en simultánea de dos tareas: **Bocetar la arquitectura** y **Diseñar la Experiencia de usuario**. Entre ambas, proveerán al equipo con las primeras decisiones gruesas sobre la arquitectura del sistema y el concepto de experiencia de usuario que se busca implementar.

**Tarea: Bocetar la arquitectura** En esta tarea los aspectos arquitectónicamente relevantes de la Visión comienzan a tomar forma con la identificación restricciones, toma de decisiones y formulación de objetivos. No sufre cambios respecto de la especificación de OpenUP/Basic.

**Tarea: Diseñar la experiencia de usuario** Debería incluir una síntesis entre la arquitectura técnica y la UX propuesta. Esta tarea contiene el diseño en el nivel macro de la UX, con las propuestas de estilo de interacción, modelo conceptual del sistema, arquitectura de la información, etc.

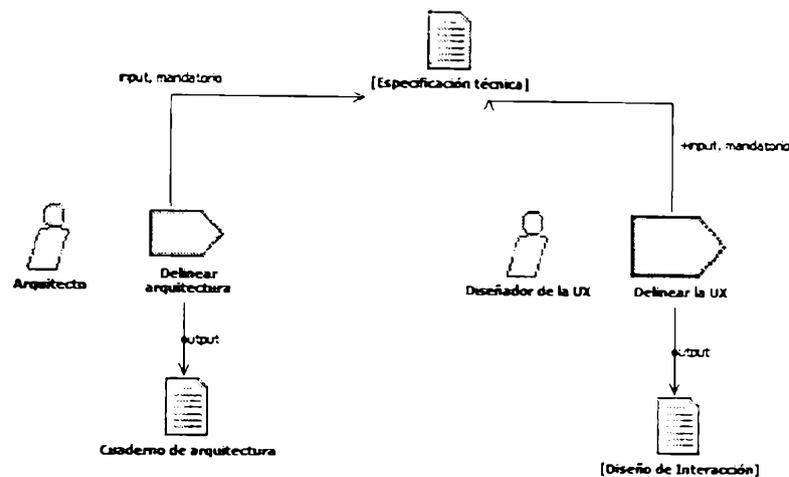


Figura 7.34: Actividad: Acordar el abordaje técnico con la Práctica DCU

**Actividad: Planificar y gestionar la iteración** En la Actividad: Planificar y gestionar la iteración no se incluyen nuevas tareas o modificación sustantivas de los flujos de información. Se enfatiza en la necesidad de la participación activa y constante de los representantes de usuarios y se agregan algunos pasos que permitan alcanzar la satisfacción de los niveles 3 y 4 del MMU-ISO. En particular, toma gran relevancia la participación de usuarios en la Tarea:Evaluar resultados.

**Tarea: Planificar la iteración** En esta tarea se planifica el alcance de la iteración con el aumento incremental de las capacidades del sistema y se construye un plan de bajo nivel que permita lograr esa capacidad.

El responsable de la tarea es el Gerente de Proyecto, sin embargo es importante incluir entre los participantes adicionales al Líder técnico del dominio, para asegurar que la priorización de items se mantiene en sintonía con las expectativas de los usuarios.

**Tarea: Gestionar la iteración** La tarea se concentra en monitorear el estado del proyecto e identificar tanto los conflictos como oportunidades, manejar excepciones y riesgos.

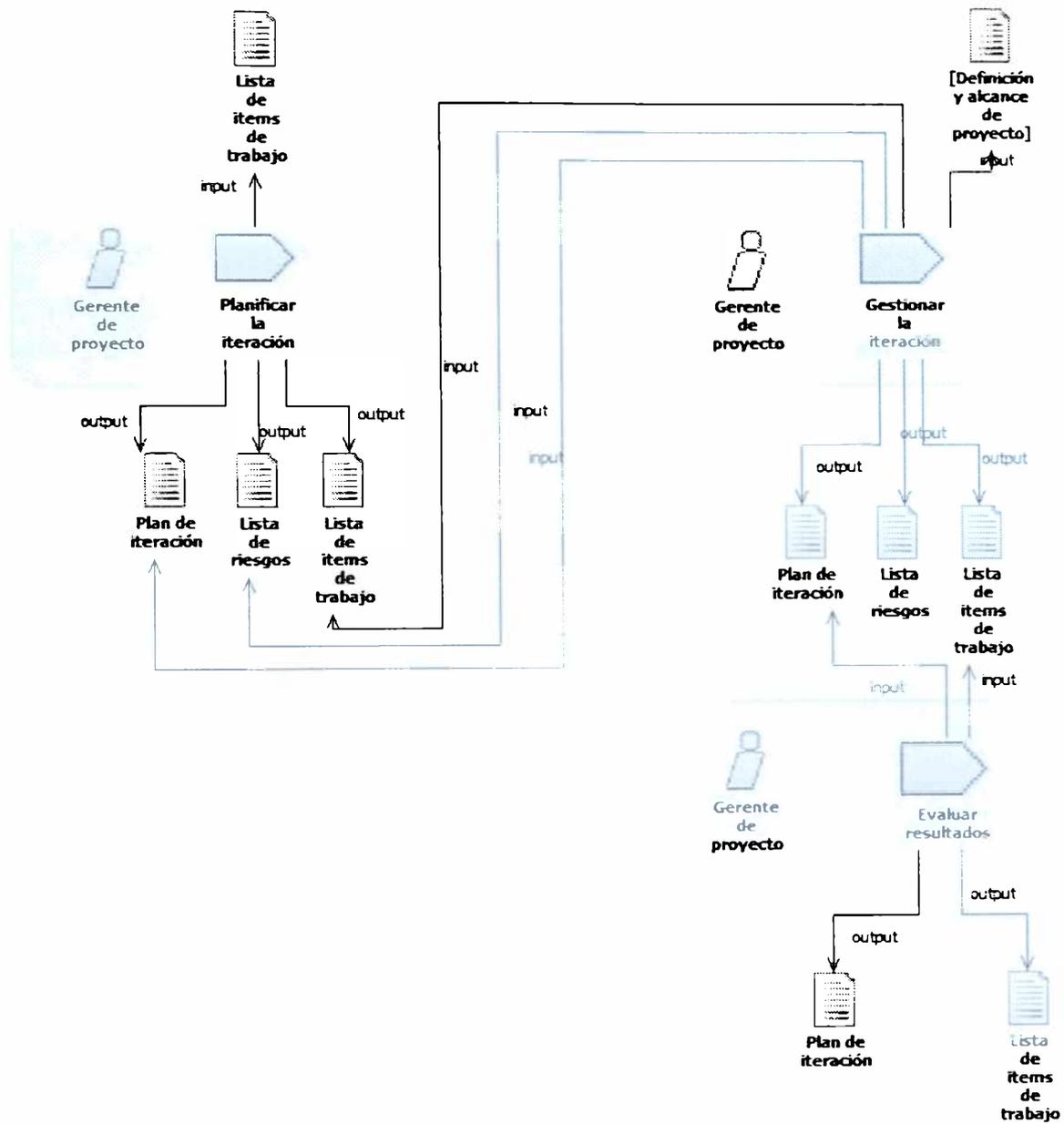


Figura 7.35: Actividad: Planificar y gestionar la iteración en OpenUP/MMU-ISO N1

Un aspecto importante de la tarea consiste en mantener comunicado el estado del proyecto a todos los involucrados. Este punto abre la puerta a la participación del Patrocinador y del Líder técnico del dominio. El primero, fundamentalmente porque el Hito de esta Fase determinará la conveniencia y factibilidad de seguir adelante con el proyecto y en consecuencia todos los aspectos que resulten en algún tipo de bloqueo o interferencia deberían ser resueltos rápidamente en esta etapa. El Líder técnico del dominio debe participar de la gestión de cada iteración para asegurar que la priorización de problemas a resolver se mantiene acorde con las expectativas de usuarios sobre todo cuando es necesario modificar las prioridades iniciales.

**Tarea: Evaluar resultados** Esta tarea consiste en la revisión de la iteración para determinar el éxito o fracaso de la misma, demostrando el valor agregado por el incremento de solución y registrando las lecciones aprendidas para poder mejorar el proceso en las próximas iteraciones.

Como tiene lugar al final de cada iteración, esta tarea puede coincidir también con el cierre de la Fase. En este caso deberá analizarse el cumplimiento o no del Hito correspondiente.

En todos los casos, es importante que participen el Patrocinador, el Líder técnico de dominio y el Representante de Usuarios de forma de garantizar que una conclusión de éxito o fracaso cuenta con el acuerdo explícito de la parte usuaria, garantizando que se avanza sobre terreno firme hacia la resolución de necesidades y expectativas identificadas.

## FASE DE ELABORACION

En esta segunda Fase se busca comprender mejor los requerimientos del sistema, crear y establecer una línea base de la arquitectura del sistema y mitigar los riesgos de mayor prioridad. Cada una de estas metas se vincula con alguna de las tareas incluidas en la Fase:

- Aumentar la comprensión de los requerimientos: Identificar y refinar requerimientos
- Diseñar, implementar y validar una línea base de arquitectura: Desarrollar la arquitectura, Desarrollar un incremento de solución y Testear la solución.
- Mitigar riesgos importantes: Planificar y gestionar la iteración

En OpenUP/MMU-ISO N1 el Proceso de Despliegue de la Fase Elaboración conserva la configuración básica con varias actividades fluyendo en paralelo, como muestra la iteración tipo representada en la Figura 7.36.

**Actividad: Identificar y refinar requerimientos** Es similar a la misma actividad en Fase de Inicio. Se modifica el foco en esta fase, pero no se agregan ni cambian tareas u otros elementos de método.

Dijimos antes que en esta Fase la Actividad cambia su foco desde el problema hacia la solución. Se trata ahora de encontrar los requerimientos con mayor valor para los involucrados, inclusive los que presentan mayor riesgo o compromiso hacia la arquitectura. Estos son los que deben ser priorizados e incluidos en las listas de Items de Trabajo para su implementación en la fase. Para cada uno de estos requerimientos se deben definir los Casos de Prueba asociados que permitirán validar las implementaciones.

La participación del Líder técnico del dominio y del Representante de usuarios contribuye a aumentar el nivel de comprensión de los requerimientos y priorizar su inclusión en la Lista de Items de Trabajo en sintonía con las expectativas y reconocimiento de valor de los interesados.

El diagrama de la Actividad se corresponde con el mostrado en la Figura 7.33.

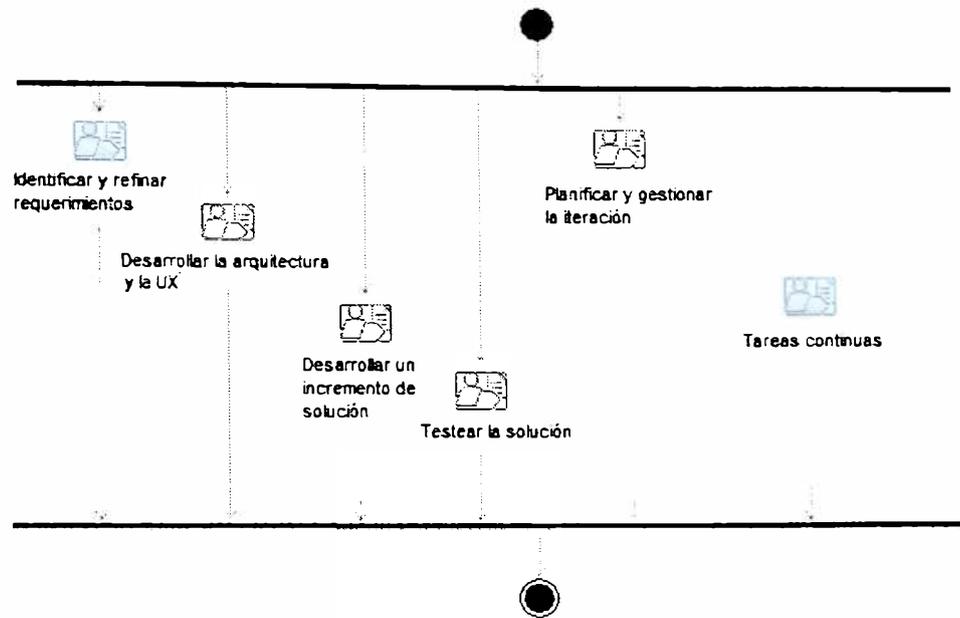


Figura 7.36: Iteración típica de Fase de Elaboración con DCU

**Actividad: Desarrollar la arquitectura y experiencia de usuario** Esta actividad intenta desarrollar y convertir en software funcional los requerimientos arquitectónicos que se priorizaron para la iteración y refinar el modelo conceptual de la experiencia del usuario. Se construye como una extensión de la Actividad: Desarrollar la arquitectura incluida en OpenUP/Basic.

Como muestra la Figura 7.37 la Actividad queda conformada por dos tareas: **Refinar la arquitectura** y **Diseñar la UX**.

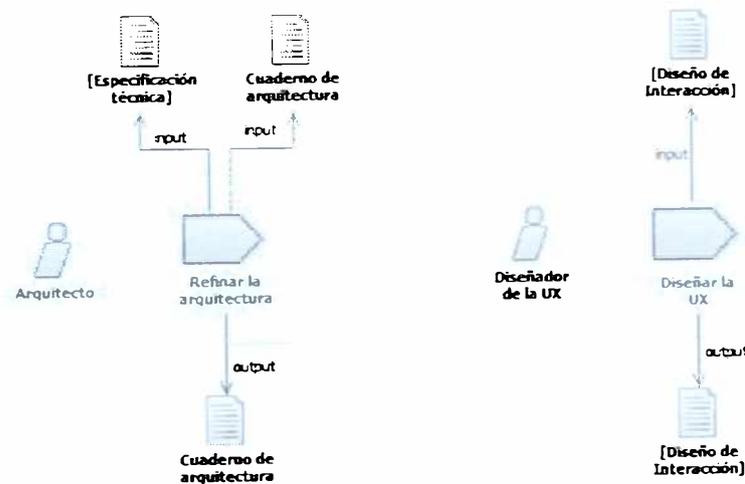


Figura 7.37: Actividad: Desarrollar la arquitectura y la UX en OpenUP/MMU-ISO N1

**Tarea: Refinar la arquitectura** La tarea que tiene por objetivo llevar el refinamiento de la arquitectura al nivel suficiente para dar soporte al desarrollo no tiene modificaciones en esta configuración. En esta configuración no hay cambios sobre la versión original.

**Tarea: Diseñar la UX** en el contexto de esta actividad, sirve como complemento de Refinar la arquitectura para poner foco en la Experiencia de Usuario de manera global y mantener la visión de conjunto de alto nivel, extendida a todo el sistema. Mientras se desarrolla a fondo un conjunto de pantallas y componentes, en el nivel general del sistema se continua refinando los prototipos de diseño de Interacción (prototipo, storyboard, mapa de navegación).

**Actividad: Desarrollar un incremento de solución** Esta actividad comprende las tareas para diseñar, implementar, probar e integrar la solución para el conjunto de requerimientos seleccionados para la iteración.

La versión básica de esta Actividad comprende cinco tareas realizadas en forma secuencial por el Desarrollador: diseñar la solución, Implementar test de desarrollador, Implementar la solución, Ejecutar test de desarrollador, Integrar y crear un build.

Todo incremento de solución debe contener aspectos vinculados con la experiencia del usuario, por lo tanto esta configuración extiende esta Actividad con la inclusión de dos tareas de la Práctica Desarrollo Centrado en Usuario:

**Tarea: diseñar componentes de interacción** Como vimos antes, esta tarea consiste en realizar las acciones necesarias para proponer y crear soluciones a los desafíos planteados por las diferentes especificaciones técnicas vinculadas con la interacción de usuario. Los productos de esta tarea (prototipos, storyboards o mapas de navegación) sirven a su vez como input para las tareas Diseñar la solución e Implementar la solución. Por ejemplo, en puede ser que en una iteración se diseñen las soluciones a un conjunto de requerimientos de interacción, que en la próxima iteración serán sujetos de las tareas vinculadas con el código de software (diseñar/implementar la solución)

**Tarea: revisar el diseño de la experiencia del usuario** Esta tarea busca dar feedback rápido sobre los diseños de interacción utilizando métodos de inspección. Puede combinarse con la tarea anterior, generando un conjunto sólido de diseños para que empleen como input los desarrolladores en la iteración siguiente.

Con estas dos tareas, el detalle de la Actividad se muestra en la Figura 7.38

**Actividad: Testear la solución** Esta actividad tiene el objetivo de evaluar la implementación de los requerimientos seleccionados para la iteración, con dos tareas básicas: Implementar pruebas de software y Ejecutar pruebas de software. La extendemos para incluir las tareas específicas de la Práctica Desarrollo Centrado en Usuario: **Preparar pruebas de usabilidad** y **Ejecutar pruebas de usabilidad**, con la meta de obtener en forma inmediata a la implementación de un conjunto de requerimientos el feedback de usuarios finales sobre la experiencia de uso que puede obtenerse.

**Tarea: preparar pruebas de usabilidad** A cargo del Tester de la Experiencia de Usuario, consiste en preparar los casos y datos necesarios para realizar las pruebas que involucren activamente a usuarios finales con el fin de evaluar la satisfacción de requerimientos y objetivos de usabilidad planteados para el proyecto.

Esta tarea debe desarrollarse en sintonía con la Tarea: diseñar la interacción del usuario. Es decir que una iteración debería incluir para un mismo conjunto de requerimientos, tanto el diseño de la interacción que luego deberá ser implementado por los desarrolladores en una iteración posterior, como el conjunto de pruebas de usabilidad a ejecutar sobre tal implementación.

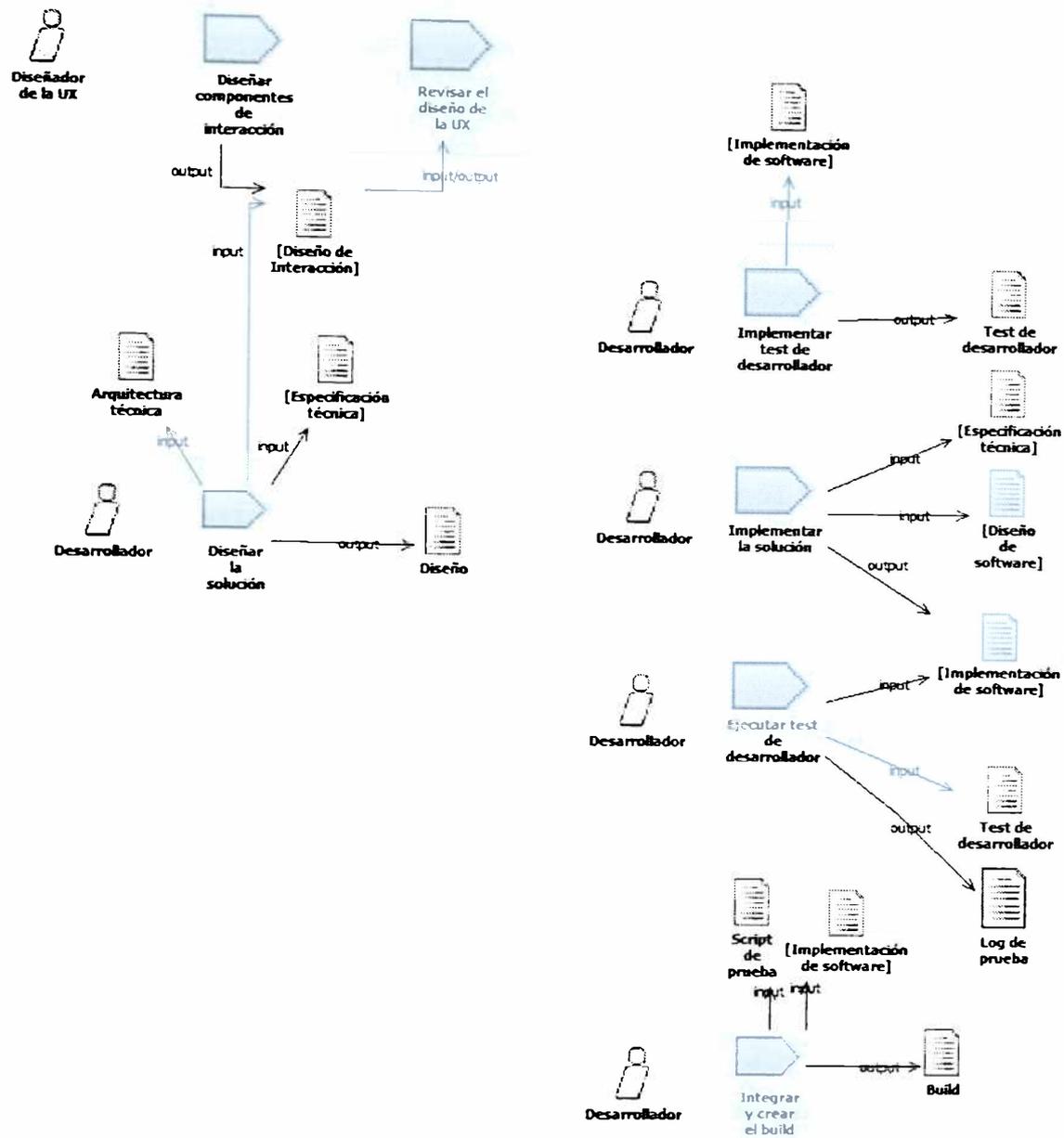


Figura 7.38: Actividad: Desarrollar un incremento de solución

**Tarea: ejecutar pruebas de usabilidad** Con el mismo responsable, el Tester de la UX, la tarea busca realizar en forma completa las pruebas preparadas en una iteración previa, analizar los datos obtenidos y comunicarlos al equipo de trabajo.

De esta forma, el nuevo detalle de la Actividad se muestra en la Figura 7.39

**Actividad: Tareas continuas** Esta actividad que habilita la realización de tareas que no estén incluidas necesariamente en el cronograma o la planificación del proyecto, pero que son necesarias para la buena marcha del mismo, no sufre modificaciones en cuanto a su configuración básica.

Sólo señalamos que si bien tarea que la compone (Tarea: Pedir cambios) puede ser llevada adelante por cualquiera de los roles intervinientes, es frecuente que lo haga alguno de los Stakeholders. Es habitual que desde este grupo de roles se generen pedidos de cambios y nuevos requerimientos (o modificaciones en la priorización de los mismos), sobre todo a partir de los resultados de las pruebas de usabilidad y evaluaciones de tareas.

Por esta razón es sumamente importante que todos los roles de Stakeholders (Patrocinador, Líder técnico de dominio, Representante de usuarios, Usuario final) tengan alguna forma concreta de plantear estos pedidos para que sean formalmente considerados en el proceso.

## FASE DE CONSTRUCCION

Esta fase se inicia luego de lograr el Hito de Elaboración que indica que la arquitectura ha alcanzado un nivel de estabilidad suficiente para mantenerse sin cambios significativos en el resto del proyecto, dando soporte sólido a la implementación de los requerimientos restantes.

Este enfoque facilita sobre todo que el Gerente del Proyecto pueda enfocarse en optimizar la eficiencia del equipo y reducir los costos del desarrollo, gracias a la limitación de los riesgos que implica una arquitectura estable.

A lo largo de las iteraciones de esta fase la funcionalidad continuará siendo implementada, testeada e integrada. La fase puede incluir hacia el final una o más versiones beta. El despliegue de versiones finales es el foco de la Fase siguiente, Transición.

Una iteración tipo en la Fase de Construcción incluye tres actividades específicamente orientadas a satisfacer el objetivo de completar la funcionalidad (Identificar y refinar requerimientos. Desarrollar un incremento de solución y Testear la solución), mientras que las de gestión (Planificar y Gestionar la iteración y Tareas continuas) se enfocan en mejorar la ecuación de costos y aumentar el paralelismo en la medida de lo posible .

La Figura 7.40 muestra el ciclo de vida de una iteración tipo de esta fase.

En la Configuración de Nivel 1 para el MMU-ISO, las actividades que conforman esta fase no requieren modificaciones respecto de las versiones que se han descrito para la Fase Elaboración. Por esta razón no incluimos aquí mayores comentarios. En el Capítulo 8 mostraremos la implementación publicable de esta configuración con el detalle de estas actividades en el plugin, de modo que pueda ser instanciado en un proyecto específico haciendo uso de la variabilidad de extensión si eso fuera necesario.

## FASE DE TRANSICION

De acuerdo con la Práctica del Ciclo de vida riesgo-valor, la fase de Transición se enfoca en desplegar el software a los usuarios y asegurar que sus expectativas son satisfechas.

Sin embargo, en la versión básica de OpenUP, el objetivo principal de la fase se reduce al ajuste fino de la funcionalidad, performance y calidad de la versión beta del sistema generada al finalizar la fase de Construcción, sin incluir actividades específicas que son propias de la preparación del despliegue tales como el test final de aceptación o el soporte del usuario (ver Capítulo 4). La iteración tipo propuesta por la configuración básica posee

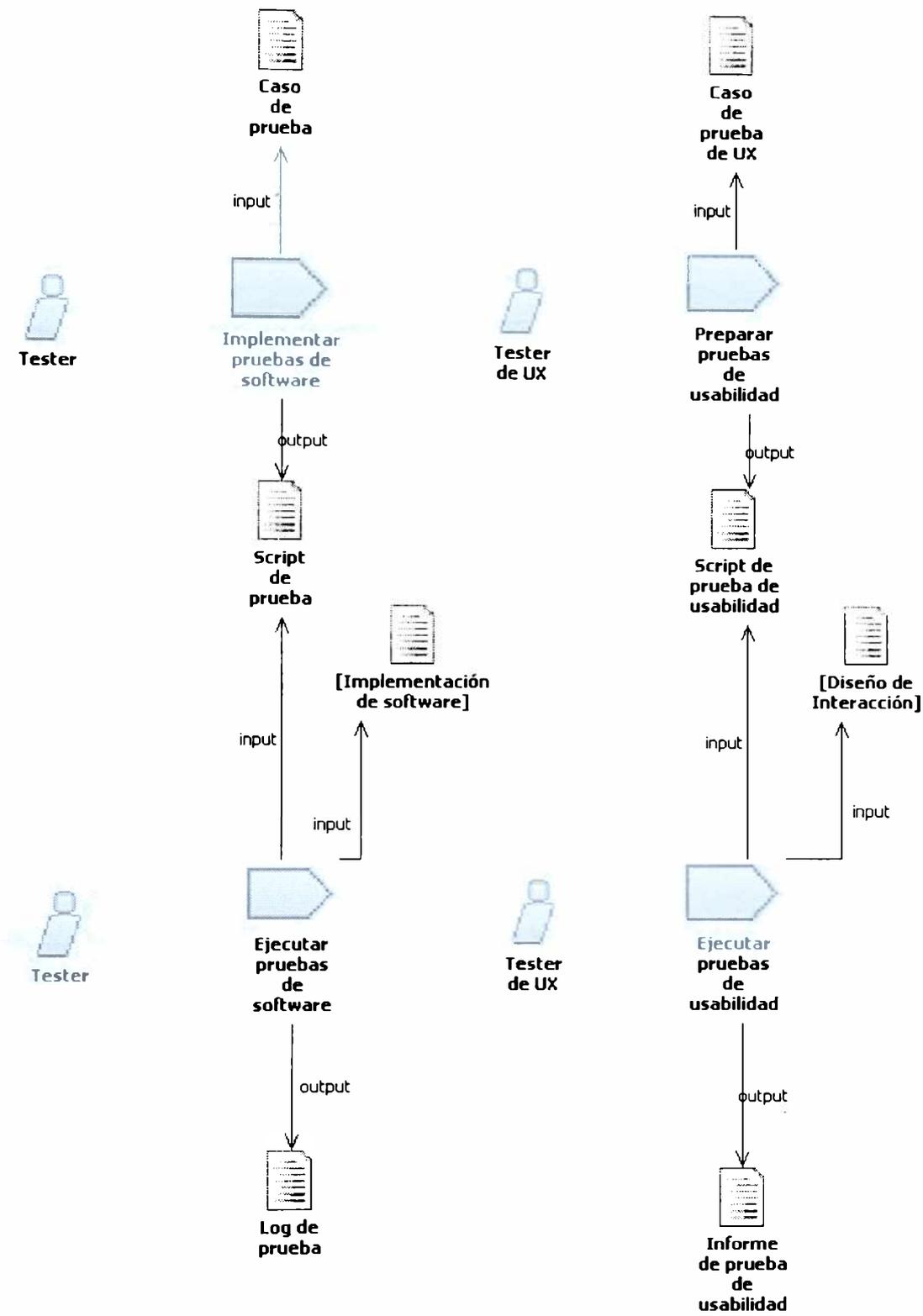


Figura 7.39: Actividad: Testear la Solución

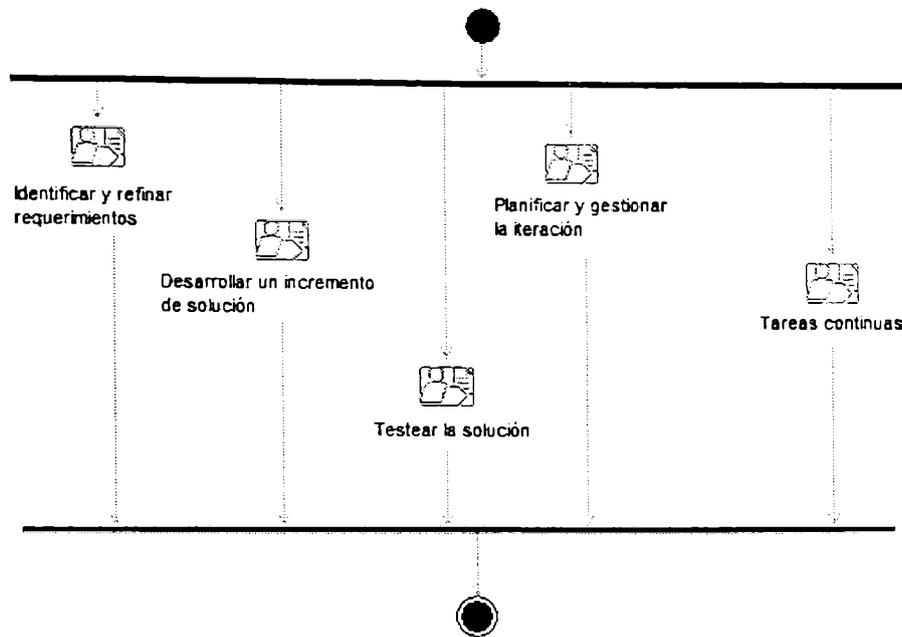


Figura 7.40: Iteración tipo de la Fase Construcción

cuatro actividades: Desarrollar el incremento de solución, Testear la solución, Planificar y gestionar la iteración y Tareas continuas.

Para superar esta carencia y dar cumplimiento al nivel inicial de capacidad del proceso ISO DCU 7, en la configuración que estamos presentando, incluimos la tarea de la Práctica Desarrollo Centrado en Usuario **Guiar y dar soporte al usuario en implementación** dentro de la Actividad: Guiar implementación y dar soporte al usuario. También refinamos la extensión de la Actividad: Testear solución (realizada para las fases elaboración y construcción) incorporando el formato de Prueba de Aceptación de Usuario como variante de las pruebas de usabilidad.

El ciclo de vida de una iteración tipo de la Fase Transición para esta configuración se muestra en la Figura 7.41

Las actividades Desarrollar un incremento de solución, Planificar y gestionar la iteración y Tareas continuas no sufren cambios respecto de lo comentado en las fases anteriores.

**Actividad: Testear la solución** Esta actividad debe incluir en la Fase Transición las pruebas de Aceptación de usuario, indispensables para garantizar el acuerdo de los usuarios con el producto a liberar.

Las pruebas de aceptación de usuario, cuya formalidad puede variar, implican coordinar el trabajo de los testers de sistema y de la experiencia del usuario.

**Actividad: Guiar implementación y dar soporte al usuario** Esta actividad persigue la meta de desarrollar todo el material de soporte necesario (tutoriales, documentación, ejemplos, etc.) sobre los requerimientos que representan el alcance la iteración, para facilitar la apropiación por parte de los usuarios finales del producto a entregar.

La Figura 7.42 muestra la composición de la actividad.

**Tarea: Producir material de entrenamiento y soporte al usuario** El producto de esta tarea puede consistir en las versiones finales de los Documentos de Usuario,

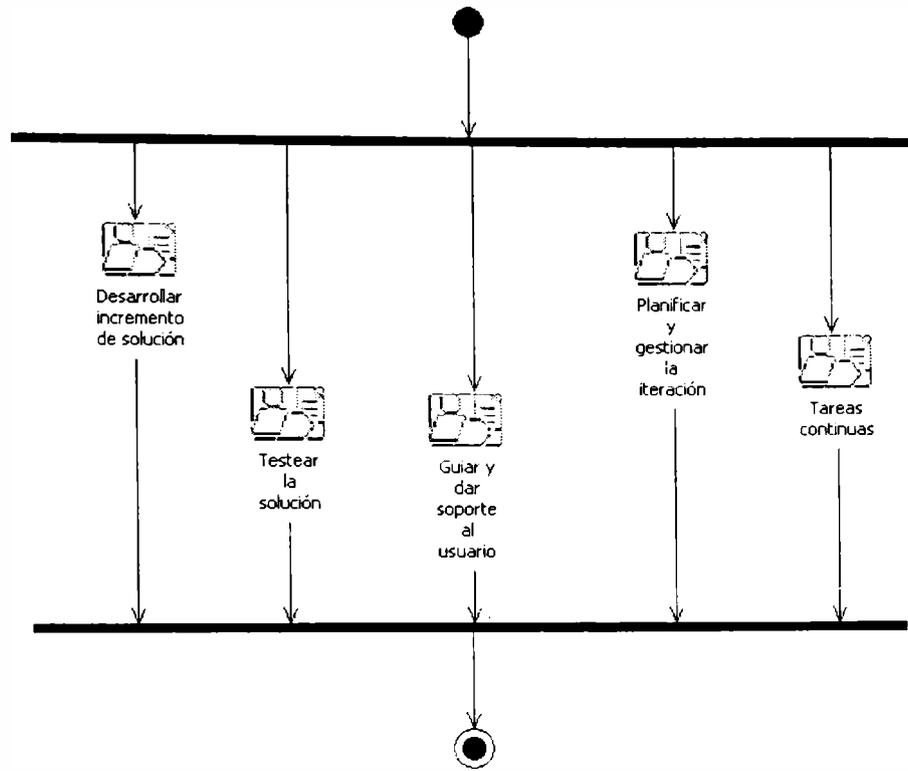


Figura 7.41: Iteración tipo de la Fase Transición

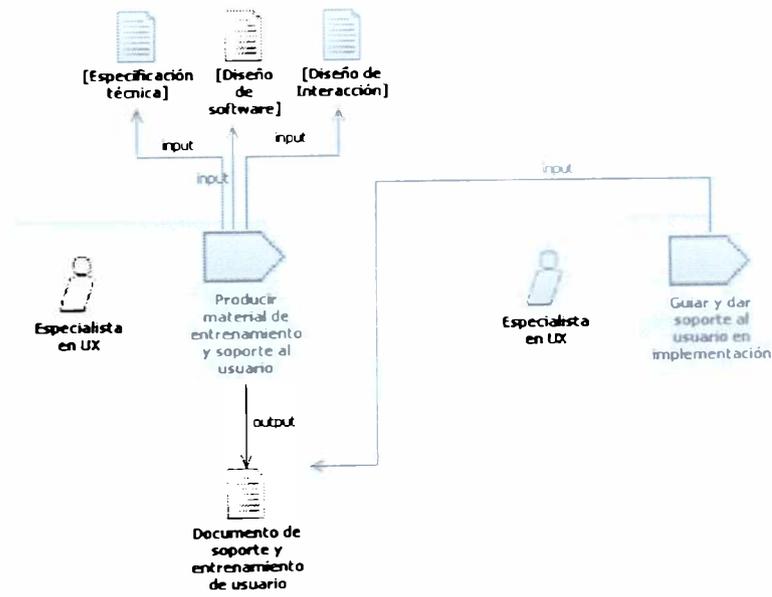


Figura 7.42: Actividad: Guiar y dar soporte al usuario en implementación

en organizaciones pequeñas o puede ser la base para que otros sectores de la organización realicen las tareas de edición y producción necesarias para entregar a usuarios.

**Tarea: Guiar y dar soporte al usuario en implementación** Esta Tarea se llevará adelante en desarrollos de sistemas a medida, en organizaciones donde el equipo de proyecto sea el encargado de la puesta en producción e implementación del producto.

### 7.3.2. OpenUP/MMU-ISO N2

Para este Nivel 2 (Proceso gestionado) es necesario identificar atributos tanto de gestión de proceso como de gestión de producto respecto del DCU. En el primer caso, es importante generar los productos de trabajo dentro del tiempo establecido y los requerimientos de recursos. Respecto de los productos, interesa encontrar prácticas de documentación y control sobre los productos de trabajo para satisfacer los requerimientos funcionales y no funcionales. Para el detalle completo de los atributos a identificar, ver el Apéndice B

Estos atributos de gestión de proceso y de producto implican asegurar en el método la presencia de contenido de Gestión de Cambios y Configuraciones y de Control de Calidad. Como vimos antes, OpenUP/Basic ayuda a encontrar calidad mediante la Práctica Desarrollo Iterativo (provee un feedback frecuente sobre la calidad), la incorporación de Templates y Checklists a modo de criterios de calidad para casi todos los productos de trabajo, la especificación de técnicas que contribuyen a la calidad en las Orientaciones y algunos elementos de QA en la Práctica Testing Concurrente. Sin embargo es necesario extender estas ayudas específicamente con métricas y evaluaciones sobre la calidad de los procesos y productos DCU.

Por otra parte, la Práctica Gestión Grupal de Cambios en la EPL describe el abordaje más simple posible, que convierte a la gestión de cambios en parte de la Gestión de Proyecto. Cualquier miembro del equipo puede requerir un cambio, que se registra en el backlog de pendientes. En la planificación correspondiente el equipo revisa ese backlog, prioriza el trabajo y decide qué cambios implementar. Sin distorsionar el carácter de bajo ceremonial y burocracia que exhibe OpenUP y a los efectos de conseguir una conformidad mínima con el MMU-ISO es necesario incluir algunas prácticas de Gestión de la Configuración sobre los productos y procesos DCU. Ya ha sido señalado en la Segunda ley de la evolución del software que la entropía de un sistema de software crece continuamente a menos que existan pasos para controlarla [Humphrey 1989]. Si el diseño de las interacciones no se mantiene bajo algún proceso de control, todos los cambios que se van generando durante el ciclo de vida, aumentando la entropía del sistema van desviando progresivamente el producto de las intenciones de los diseñadores originales, provocando la progresiva degradación de la experiencia de usuario.

La configuración que mostraremos aquí corresponde a una extensión de OpenUP/MMU-ISO N1 que incorpora los elementos de método y contenido necesarios para satisfacer estos requerimiento, superando las carencias que identificamos en el Capítulo 6 para que OpenUP/Basic pueda dar satisfacción al Nivel 2 del MMU-ISO.

#### FASE DE INICIO

Se modifican dos Actividades sobre la configuración anterior: **Iniciar el proyecto** y **Planificar y gestionar la iteración**. El resto del proceso se mantiene de acuerdo con lo indicado para OpenUP/MMU-ISO N1 en las cuatro fases.

**Actividad: Iniciar el proyecto** En esta actividad se ve afectada fundamentalmente la Tarea: Planificar el proyecto para establecer criterios de gestión sobre los elementos de UX que serán utilizados en el proyecto.

**Tarea: Planificar el proyecto** Deben definirse los criterios de performance y los productos de trabajo específicos de DCU que se utilizarán en el proyecto, así como la gestión de su calidad e integridad. Para mantener los lineamientos indicados en párrafos anteriores extendemos esta Tarea agregando dos pasos que aseguren la planificación de la Gestión de Cambios, de la Configuración y el Control de Calidad de todos los productos y tareas de DCU.

Los pasos de la tarea en su versión base son:

- Establecer un equipo cohesionado
- Estimar el tamaño del proyecto
- Evaluar los riesgos
- Estimar velocidad y duración de proyecto
- Diagramar el ciclo de vida del proyecto
- Establecer costo y articular valor
- Planificar el despliegue

En esta configuración agregamos dos pasos que nos aseguren la planificación de la gestión DCU como lo pide MMU-ISO:

- **Planificar la calidad de la UX** Consiste en definir un mínimo plan que haga previsible el aseguramiento de un mínimo de calidad en tareas y productos vinculados con la UX. Será necesario definir los objetivos de calidad, los roles y las responsabilidades, las tareas y cronograma. Las técnicas específicas de evaluación de productos de trabajo, tales como Revisiones o Auditorías, Inspecciones y Recorridas constituyen una forma poderosa de mejorar la calidad y productividad del proceso de desarrollo. Un Plan de Aseguramiento de Calidad es un documento que contiene toda la información necesaria para realizar las tareas de aseguramiento de calidad en el proyecto. Este plan debería contener el tipo de esfuerzo de evaluación de calidad de productos que se utilizará en el proyecto: Revisiones, Inspecciones o Recorridas. También deben incluirse las métricas de calidad que se emplearán, utilizando los parámetros habituales como cantidad de defectos, cobertura de pruebas, performance, conformidad con estándares, aceptabilidad y satisfacción.
- **Planificar la Gestión de Configuración y Cambios de la UX** El propósito de este paso es establecer las políticas de gestión de la configuración a emplear en el proyecto para monitorear y salvaguardar los activos y para reforzar las buenas prácticas de desarrollo de software. Estas políticas deberían mejorar la comunicación entre los miembros del equipo y minimizar los problemas encontrados cuando se integra su trabajo. Este Plan de la Gestión de Configuración de UX debería contener al menos la definición de los elementos a mantener, las prácticas para establecer líneas base, las prácticas de archivado de activos y el ritmo de los reportes de configuración.

**Actividad: Planificar y gestionar la iteración** Hemos visto que la gestión de proyecto en OpenUP se apoya entre otras en la Práctica de Planificación en dos niveles. En el nivel macro, la extensión de la Tarea: Planificar el proyecto incluye los contenidos necesarios para este nivel de Capacidad del MMU-ISO. Debemos asegurar la consistencia con el nivel micro (la iteración), por lo que vamos a proponer una extensión en el mismo sentido para la Actividad: Planificar y gestionar la iteración.

**Tarea: Planificar la iteración** Necesitamos asegurar que las planificaciones en este nivel micro acompaña las definiciones planteadas en el Plan de Proyecto, en particular las relacionadas con la calidad de la UX y la configuración y los cambios de los productos que la proveen.

Los pasos de la Tarea son:

- Priorizar la Lista de items de trabajo
- Definir los objetivos de la iteración
- Comprometer el trabajo para la iteración
- Identificar y revisar riesgos
- Definir criterios de evaluación
- Refinar la definición y alcance de proyecto

La extensión que se propone no agrega nuevos pasos, sino que redefine algunos para asegurar que el nivel micro de planificación se mantiene en línea con lo establecido para el nivel macro.

Todos los pasos que toman como insumo la Lista de items de trabajo y la reordenan o comprometen trabajo para la iteración deben tener en cuenta los criterios de gestión del cambio de la UX que se establecieron en el Plan de Proyecto. Esto nos obliga a modificar las definiciones de los pasos: Priorizar la Lista de Items de Trabajo, Definir objetivos de iteración, Comprometer trabajo para la iteración y Refinar la definición y alcance de proyecto.

En la definición de los criterios de evaluación (paso 5), deben fijarse las pautas de calidad esperables de los productos relacionados con la experiencia de usuario. Este paso debe asegurar la referencia de cualquier criterio y actividad vinculada con la calidad que se establezca para la iteración presente con el Plan de Calidad definido en el Plan de Proyecto.

**Tarea: Gestionar iteración** Es el ámbito para monitorear las métricas de usabilidad, el nivel de calidad de prototipos y los otros productos DCU que se haya decidido incluir en el proyecto, tanto en tiempos de realización como en calidad y cantidad de revisiones con usuarios

Los pasos de la tarea son:

- Rastrear el progreso de la iteración
- Capturar y comunicar el estado del proyecto
- Manejar excepciones y problemas
- Identificar y administrar riesgos
- Administrar objetivos

No se proponen cambios en los pasos de la Tarea, pero se modifica la Orientación Asignar Cambios a una Iteración para incluir explícitamente la consideración del Plan de Gestión de Cambios y Configuración de UX en el proceso de revisión, priorización de cambios en la iteración actual y su asignación en una futura.

Se modifican las Consideraciones Claves de la Tarea para incluir en la recolección de métricas a monitorear las que se hayan definido en el Plan de Proyecto para la Gestión de Calidad de la UX.

**Tarea: Evaluar resultados** Esta tarea es clave del ciclo DCU y para asegurar la gestión tanto de sus procesos como de sus productos. En el Nivel 1 ya incorporamos la participación del rol Representante de usuarios, en este segundo nivel no modificamos.

La tarea se compone de los siguientes pasos:

- Preparar la evaluación de la iteración
- Demostrar valor y obtener feedback
- Realizar una retrospectiva de fin de iteración, fin de fase o fin de proyecto

No es necesario modificar esta definición. Sólo deben respetarse en cada paso las prácticas sobre gestión de procesos y productos de DCU definidas en los Planes de Proyecto y de Iteración.

## FASES DE ELABORACION, CONSTRUCCION, TRANSICION

En las Fases posteriores al Inicio, las prácticas de gestión de procesos y productos DCU se completa respetando en la Tarea Pedir un Cambio el proceso de gestión de cambios definido en los planes.

Esta tarea forma parte de la Actividad: Tareas continuas y cualquiera de los roles puede disparar un pedido de cambio en OpenUP, como vimos en el Capítulo 4. La definición base de la Tarea consta de dos pasos: Obtener la información del pedido y Actualizar la lista de items de trabajo.

El resto de la gestión del cambio queda incorporado en las tareas de Gestión de Iteración con las modificaciones que incorporamos en la Fase de Inicio.

### 7.3.3. OpenUP/MMU-ISO N3

Para el Nivel 3 del MMU-ISO (Proceso establecido) es necesario asegurar el despliegue de un proceso que esté definido no sólo en las prácticas básicas de DCU y una gestión ordenada de su proceso y sus productos, sino en los buenos principios de ingeniería de software en general. Entonces debemos poder reconocer atributos del proceso que estén relacionados con su misma definición (adaptar los procesos estándar a las situaciones específicas del proyecto) y con la determinación de los recursos que serán necesarios para llevarlo adelante. Es decir, no se trata de gestionar bien un proceso y unos recursos dados. Se trata de definir en forma adecuada y basada en buenos principios, la mejor estructura para el proceso y los recursos que serán necesarios para ejecutarlo.

Al igual que en la configuración anterior, OpenUP/MMU-ISO N3 se construye de manera incremental sobre N2, agregando todos los elementos del método que puedan dar lugar a instanciaciones que cumplan con lo expresado en el párrafo anterior.

Para el detalle de los atributos de definición del proceso y de los recursos ver el Apéndice B.

Las carencias a suplir, según el assessment realizado (ver Capítulo 6) son la falta de prácticas para definir el mejor proceso para el proyecto y establecer los recursos necesarios en todos los procesos DCU del MMU-ISO. Esto se verá reflejado fundamentalmente en la presencia de extensiones en actividades como la Preparación del Proyecto y la Planificación y Gestión de cada iteración.

## FASE DE INICIO

**Actividad: Iniciar el Proyecto** En esta Actividad agregamos la Tarea: Adaptar el proceso DCU, para realizar al inicio del proyecto la adaptación y configuración de todo el proceso de acuerdo con las necesidades específicas del proyecto. El resto de la Actividad

es igual a la Configuración N2, con las Tareas de Desarrollar la Visión y Planificar el Proyecto.

**Tarea: Adaptar el proceso DCU** Esta tarea se alimenta de la Configuración que aquí mostramos y, de existir, podrá tener en cuenta los reportes de la organización respecto del estado actual de sus procesos de desarrollo, herramientas, recursos, etc. Ya describimos en este mismo Capítulo el alcance de la tarea que consiste básicamente en analizar el proyecto, determinar el esfuerzo de adaptación que se justifica para el proyecto, configurar el proceso adecuado incluyendo el ciclo de vida DCU dentro del mismo y publicarlo para que todos los integrantes del equipo lo conozcan. Durante el proyecto, en las tareas de la Actividad: Planificación y Gestión de la Iteración se llevará a cabo el mantenimiento del proceso definido, con las correcciones que sean necesarias en función del devenir del proyecto.

En consecuencia, el detalle de esta Actividad queda como se muestra en la Figura 7.43

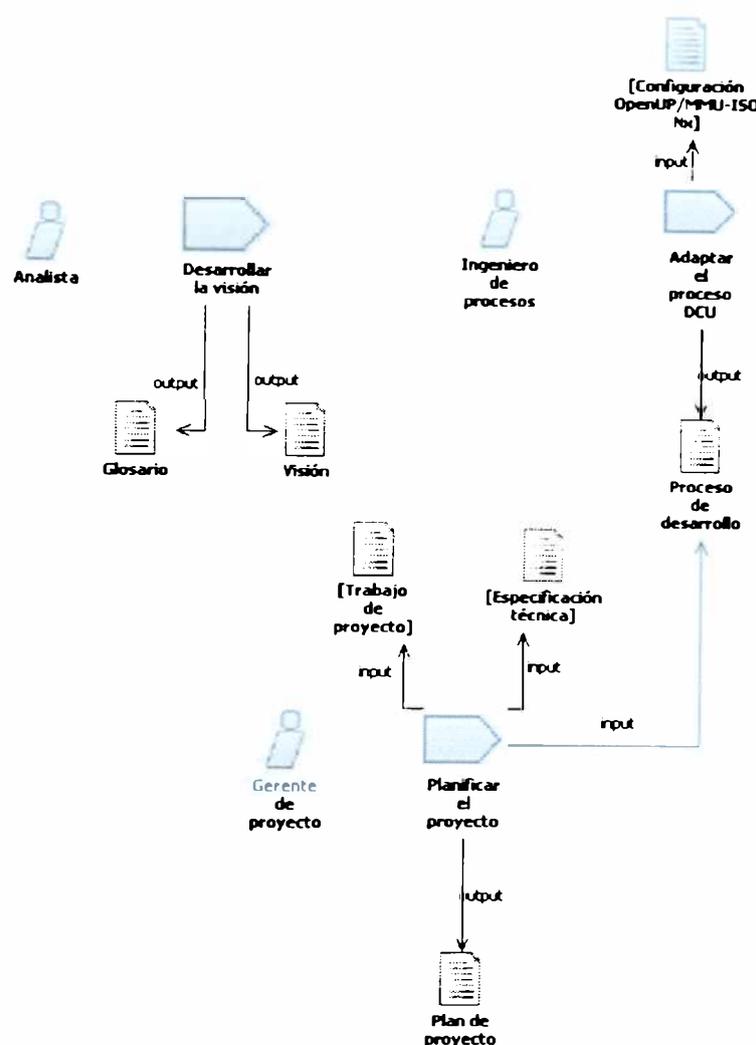


Figura 7.43: Detalle de la Actividad: Iniciar el Proyecto en OpenUP/MMU-ISO N3

**Actividad: Planificar y gestionar la iteración** Además del nivel macro de Proyecto, en el nivel micro de cada Iteración también debe haber una gestión de la adaptación realizada al proceso de desarrollo.

**Tarea: Planificar la iteración** Se extiende con un paso para incorporar las modificaciones de proceso que puedan ser sugeridas por la tarea de Evaluar resultados.

**Paso: Actualizar el proceso** Incorporar las modificaciones de proceso sugeridas por la última retrospectiva.

**Tarea: Gestionar la iteración** Sin cambios respecto de OpenUP/MMU-ISO N2.

**Tarea: Evaluar resultados** En esta tarea se incorpora un cuarto paso denominado “Monitorear el proceso” cuando se realiza esta tarea coincide con la retrospectiva:

**Paso: Monitorear el proceso** Analizar el impacto (positivo o negativo) que las adaptaciones incorporadas en el proceso han tenido en el desarrollo de la iteración y proponer cambios en caso de ser necesario.

### **FASES DE ELABORACION, CONSTRUCCION, TRANSICION**

Los procesos y métodos de estas Fase no sufren otros cambios en el Nivel 3. Las actividades de planificación y gestión que se detallaron para la Fase Inicio se realizan con la misma definición.

## Capítulo 8

# OpenUP/MMU-ISO como plugins del EPF Composer

### 8.1. El EPF Composer

El Eclipse Process Framework Composer (“EPF Composer” o simplemente “Composer”, en el resto del Capítulo) es una plataforma de herramientas para ingenieros de procesos, líderes y gerentes de proyecto responsables de mantener e implementar procesos para organizaciones que desarrollan software.

Durante la implantación de un proceso de desarrollo es necesario resolver dos cuestiones. Por un lado, que todos los desarrolladores conozcan y entiendan las prácticas claves del proceso de desarrollo y fundamentalmente de qué manera un equipo de trabajo las adapta a sus particularidades (aunque los métodos ágiles se basen en grupos auto-organizados, esto no habilita a asumir que todos conocen cómo encara una organización concreta las tareas de elicitación de requerimientos, diseño, construcción, testing, etc.). El otro tema se relaciona con la inclusión de esas prácticas en el ciclo de vida completo del proyecto.

El Composer intenta satisfacer ambas necesidades proveyendo el contenido completo de la Librería de Prácticas del EPF (la EPL que ya describimos en capítulos anteriores) y una herramienta para seleccionar, customizar, ensamblar y publicar partes de esa Librería como un proceso específico adecuado a las particularidades de un equipo. El Composer provee catálogos de procesos predefinidos para situaciones típicas de proyectos y los procesos de despliegue (bloques de construcción de procesos que representan las mejores prácticas de desarrollo para una disciplina, tecnología o estilo de desarrollo específicos). Estos bloques forman una caja de herramientas para ensamblar rápidamente los procesos para un proyecto en particular. El Composer también permite establecer procesos de despliegue propios de una organización y publicar todo como sitios web, para dar acceso al equipo de trabajo.

El escenario de uso más simple del Composer consiste en utilizar las prácticas de la EPL y los procesos de despliegue del EPF tal como vienen dados. Sin embargo, habitualmente una organización necesita sólo un subconjunto de esos elementos. Probablemente, además, será necesario incluir elementos propios y vincularlos con los existentes. Esto da lugar a diferentes escenarios de autoría y configuración con el Composer, que describimos a continuación.

El Composer es una herramienta gratuita, desarrollada dentro del entorno ECLIPSE. No sólo permite editar fragmentos de método, procesos o metodologías, sino también generar automáticamente la documentación adecuada en formato para la web. La Figura 8.1 muestra la pantalla de edición de una Tarea dentro del Composer. Dichos fragmentos

se almacenan en formato XMI y, al estar basados en la UMA (cuyo metamodelo es el estándar SPEM 2), pueden ser reutilizados por otras herramientas CASE. El Composer resulta entonces un editor de procesos SPEM 2, que incluye opciones adicionales para publicar de forma automáticamente sitios web.

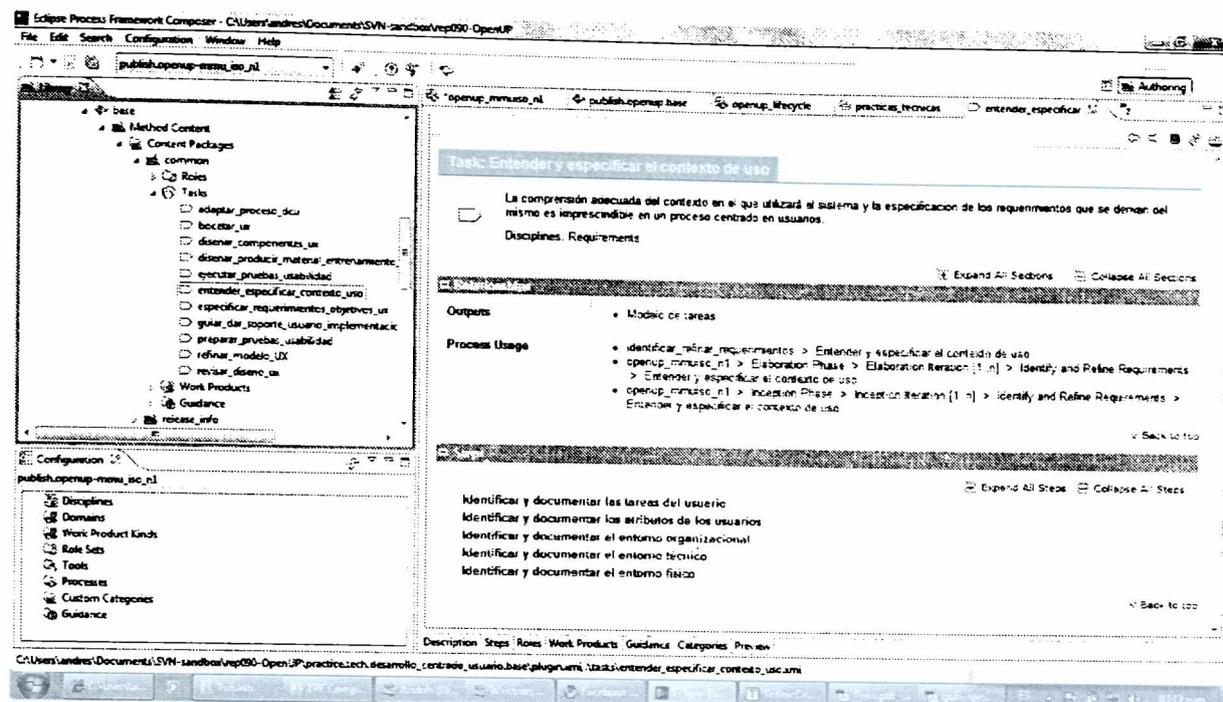


Figura 8.1: Pantalla de edición de una Tarea en el Composer

## Organización del repositorio de métodos y procesos en Composer

El repositorio o biblioteca de métodos y procesos es una colección de uno o más plugins y una o varias configuraciones. Cada plugin se almacena en un directorio de disco diferente e incluye contenido de método y de procesos. A su vez, el Contenido de Método está formado por paquetes de contenido, categorías estándar y categorías personalizadas. La rama de Procesos contiene patrones de proceso o entrega y procesos para despliegue. La Figura 8.2 muestra la estructura mencionada.

En el Capítulo 4 describimos en detalle los diferentes elementos que pueden poblar el Contenido de Método (Tareas, Roles, Productos de trabajo, Orientaciones) y los Procesos (Elementos de desglose, Actividades) y las relaciones de asociación entre ellos.

## Reuso y variabilidad

La organización en plugins facilita la reutilización de los elementos de contenido y proceso definidos en la biblioteca, mediante la referencia de un plugin nuevo a otros existentes.

Para permitir esta reutilización el Composer ofrece el mecanismo de variabilidad que permite modificar elementos de método o de proceso sin tocar el original, definiendo las diferencias (agregados, cambios, omisiones). Describimos a continuación las opciones de variabilidad que brinda el Composer y que utilizamos luego en la construcción de nuestros plugins.

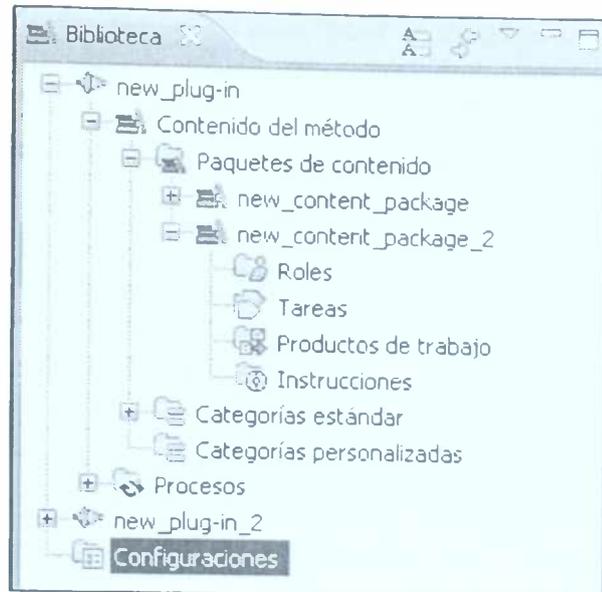


Figura 8.2: Estructura de organización del repositorio en el Composer

### Variabilidad de Elementos de Contenido

El Composer incluye los cinco tipos previstos para elementos de contenido en SPEM 2.0: No aplicable, Contribuye, Extiende, Reemplaza y Extiende y reemplaza:

**No aplicable** Es el valor por defecto. No hay relación de variabilidad entre el elemento en cuestión y otros.

**Contribuye** Un elemento E que contribuye a un elemento base EB añade sus valores de atributos e instancias de asociación a EB sin modificar directamente las propiedades que ya tiene EB (es un agregado). Al generar una vista el elemento base EB se muestra combinado con los atributos y relaciones de E, mientras que E queda oculto.

**Reemplaza** Un elemento E puede reemplazar los atributos y asociaciones de salida de un elemento base EB sin modificar directamente ninguna propiedad de EB. El elemento E que reemplaza se incluirá en las vistas generadas mientras que el reemplazado EB no aparecerá.

**Extiende** El elemento E hereda las propiedades del elemento base EB que extiende, dejando EB totalmente intacto. El elemento E incluye las propiedades heredadas de EB más las propias. Tanto EB como E aparecen en las vistas generadas.

**Extiende y reemplaza** esta variabilidad combina los efectos de Extiende y de Reemplaza. El resultado es que las propiedades que no se han definido en E que extiende y reemplaza se heredan del elemento base EB. En cambio, las propiedades definidas en E sustituyen a las de EB. E aparece en las vistas que se generan, pero EB no. Este tipo de variabilidad se utiliza al generar plugins que renombran elementos, reemplazan descripciones, etc. sin remodelar completamente todas las relaciones y atributos del plugin base.

### Variabilidad de Elementos de Proceso

Los elementos de proceso de tipo actividad (fase, iteración, actividad, patrón de proceso) se pueden reutilizar en otros elementos de proceso más grandes. En este caso reciben el nombre genérico de Actividad en uso. Una Actividad en uso define la habilidad para

reutilizar las estructuras definidas en los elementos de desglose de una actividad en una segunda actividad, sin necesidad de copiar físicamente dichas estructuras.

Los mecanismos de variabilidad previstos en Composer para los procesos son:

**No aplicable** Cuando una actividad no tiene asociaciones de Actividad en uso.

**Extensión** Una actividad extendida A hereda la estructura interna (subestructuras e instancias de asociación) de una actividad base AB. Se pueden anuevos elementos de desglose a A, pero no se pueden modificar los elementos de desglose heredados desde AB. Los cambios realizados en A no afectan a AB. Las modificaciones realizadas en AB con posterioridad se reflejan automáticamente en A y en las demás actividades en uso que extienden AB.

**Contribución local** Sirva para definir adiciones locales específicas (contribuciones) a los elementos de desglose heredados vía la extensión del tipo anterior.

**Reemplazo local** de forma similar al anterior sirve para definir sustituciones locales específicas (reemplazos) a los elementos de desglose heredados vía la extensión.

Además los elementos de proceso de tipo actividad (fase, iteración, patrón de proceso) permiten variabilidad de los tipos con las mismas características explicadas para elementos de contenido: Extiende, Contribuye, Reemplaza.

## 8.2. El uso del Composer

### Crear contenido de método

Tal como vimos en el Capítulo 7 el primer paso es crear los elementos de contenido que luego se utilizarán en la creación de procesos. En el Composer, los elementos de contenido se pueden organizar y jerarquizar mediante la creación de paquetes de contenido, dentro de los cuales pueden definirse las tareas, roles, productos de trabajo y orientaciones. La especificación de los atributos de estos elementos se realiza en la solapa Descripción del editor del elemento. Para cada tipo de elemento, Composer ofrece un conjunto de solapas que permite definir además las relaciones de ese elemento con otros y previsualizar el contenido final a exhibir (ver figura 8.3).

### Crear Procesos

Después de definir los elementos de contenido de método, éstos se pueden reutilizar para crear procesos. El Composer habilita la creación de dos tipos de procesos SPEM 2: los **Patrones de proceso o capacidad** (*capability patterns*) que describen una agrupación reutilizable de tareas o actividades y los **Procesos de despliegue** (*Delivery process*) que describen un enfoque completo e integrado para una metodología completa, un tipo de ciclo de vida, tipo específico de proyecto, etc.

Al igual que los elementos de contenido, los patrones de proceso y los procesos de despliegue se pueden organizar y jerarquizar mediante paquetes de proceso. La Figura 8.4 muestra a la izquierda, en la ventana de Librería la organización de Patrones de Capacidad y Procesos de Despliegue para el plugin OpenUP/MMU-ISO Nivel 1 (que presentamos en detalle más adelante) y a la derecha la WBS o Estructura de Desglose del Ciclo de Vida de dicho proceso.

Para representar estos procesos se emplean estructuras de descomposición o desglose, mediante las cuales se representa la jerarquía de actividades e hitos y se ordena el flujo de trabajo indicando relaciones de precedencia entre los elementos de la estructura.

Para ver y editar desde diferentes puntos de vista la estructura de desglose de un proceso/metodología completo o de un fragmento, el Composer incluye las siguientes solapas:

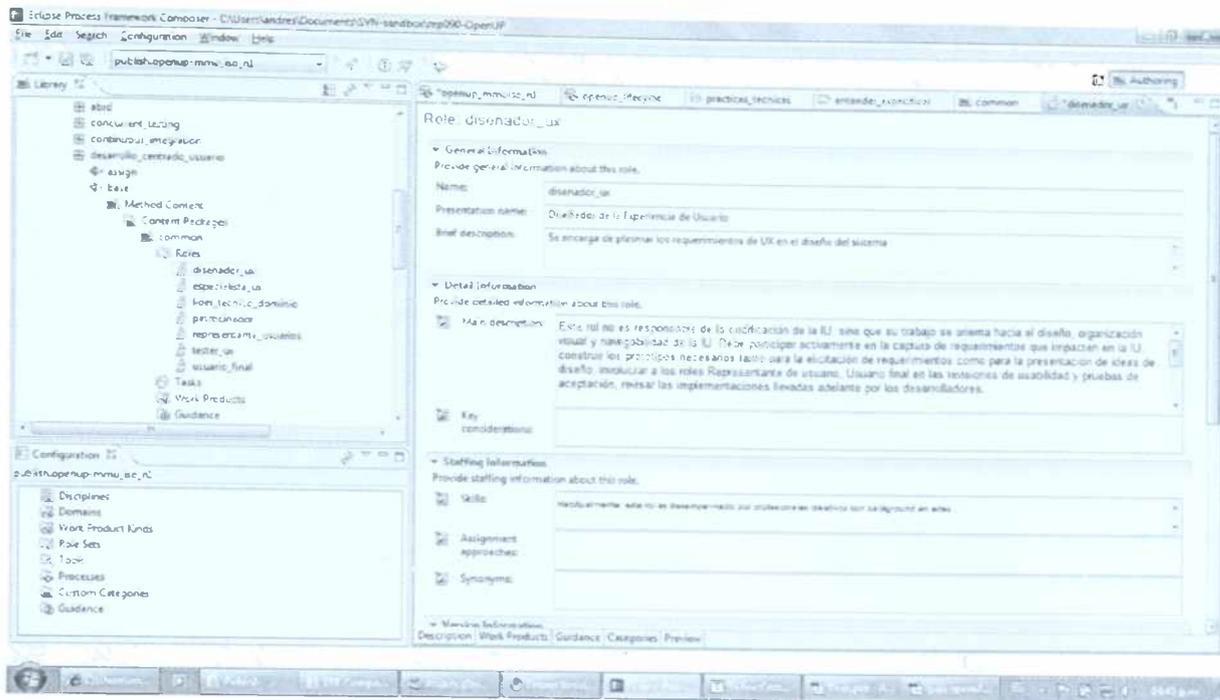


Figura 8.3: Creación de un Rol dentro del Paquete de Contenido de Método

The screenshot shows the WBS view in Eclipse Process Framework Composer. The table below represents the data shown in the screenshot:

Presentation Name	Index	Predecessors	Type	Planned	Repeat	Multi	Ongoing	Event
openup_mmui_so_n1	0		Delivery Process					
Inception Phase	1		Phase					
Inception Iteration [1..N]	2		Iteration					
Isolate Project	3		Activity					
Plan and Manage Iteration	6		Activity					
Identify and Refine Requirements	10	3	Activity					
Agree on Technical Approach	17	3	Activity					
Lifecycle Objectives Milestone	20	2	Milestone					
Elaboration Phase	21	20,1	Phase					
Elaboration Iteration [1..N]	22		Iteration					
Plan and Manage Iteration	23		Activity					
Identify and Refine Requirements	27		Activity					
Develop the Architecture	34		Activity					
Develop Solution Increment	45		Activity					
Test Solution	53		Activity					
Ongoing Tasks	58		Activity					
Lifecycle Architecture Milestone	60	22	Milestone					
Construction Phase	61	60,21	Phase					
Construction Iteration [1..N]	62		Iteration					
Plan and Manage Iteration	63		Activity					
Identify and Refine Requirements	67		Activity					
Develop Solution Increment	72		Activity					
Test Solution	80		Activity					
Ongoing Tasks	85		Activity					
Initial Operational Capability Milestone	87	63	Milestone					
Construction Iteration [1..N]	88		Iteration					
Transition Phase	113	87,63	Phase					
Transition Iteration [1..N]	114		Iteration					
Plan and Manage Iteration	115		Activity					
Develop Solution Increment	119		Activity					

Figura 8.4: WBS del Ciclo de Vida OpenUP/MMU-ISO Nivel 1

- Estructura de Desglose de Trabajo (*Work Breakdown Structure*): Corresponde al conocido WBS de gestión de proyectos. En el Composer es una tabla que contiene la descomposición jerárquica de los elementos que definen el trabajo a realizar: actividades (patrones de proceso, fases, iteraciones), hitos, tareas, etc. Un ejemplo se muestra en la parte derecha de la Figura 8.4
- Asignación de Equipos (*Team Allocation*): Muestra una jerarquía similar a la anterior, pero mostrando para cada actividad (cada nivel) la lista de roles que participan.
- Utilización de Productos de Trabajo (*Work Product Usage*): Esta perspectiva se centra en los productos de trabajo de forma similar a como la anterior se centra en los roles.
- Vista Consolidada (*Consolidated View*): Combina las informaciones que se muestran en las tres vistas anteriores. Solo existe con fines informativos, es decir, no permite edición. Es similar a la Previsualización para los elementos de contenido.

## Crear configuraciones

Una configuración de método es una selección de contenidos de los plugins de una librería, de forma que se limita la vista de la librería al subconjunto seleccionado.

En una configuración se seleccionan los paquetes y procesos que queremos incluir. La selección de elementos incluidos se puede refinar añadiendo o sustrayendo de la configuración todos los elementos categorizados en una misma categoría.

Las configuraciones de método permiten definir qué elementos aparecerán en la web publicada y cuáles no. También permiten seleccionar los elementos que son visibles para los procesos creados. Ofrecen una alternativa en la creación de procesos, ya que puede definirse un proceso general que incluya contenidos para varios tipos de procesos más específicos. Mediante el uso de configuraciones se seleccionan los contenidos de cada uno de esos tipos de procesos más específicos. Crear los diversos tipos de proyecto es sencillo seleccionando la configuración adecuada para el proceso.

La creación de configuraciones se hace desde la vista *Biblioteca*. Para especificar el contenido de una configuración y refinar dicha selección se siguen los siguientes pasos en la solapa *Selección de paquete y plug-in* del editor de la configuración:

1. Seleccionar los plug-ins que nos interesen para la configuración.
2. Seleccionar los paquetes de contenido y los procesos de los plug-ins seleccionados que nos interesen en la configuración.
3. Refinar la selección utilizando las categorías que se incluirán y las categorías que se sustraerán de la configuración. De esta forma se pueden incluir o no conjuntos de elementos relacionados de forma lógica en vez de por estar relacionados de forma física (estar incluidos en el mismo paquete).

En una configuración de método, además de seleccionar los contenidos de la configuración, podemos seleccionar las vistas de la configuración. Las vistas son, básicamente, categorías que agrupan los contenidos de la configuración. Estas vistas son útiles al publicar la configuración, ya que se generarán árboles de navegación con los elementos definidos en la vista mediante los que explorar el contenido de la configuración. Este concepto no aparece en el estándar SPEM, es un añadido del Composer.

## Publicar contenidos

La publicación de una configuración consiste en la generación de un sitio web que contiene los elementos de la configuración. El Composer ofrece opciones para personalizar el sitio a publicar, modificando los elementos a publicar (título, glosario, índice, diagramas, etc.) y para validar automáticamente los enlaces y páginas generados.

El sitio a generar puede ser estático, como un conjunto de páginas HTML organizadas en carpetas, o como una aplicación web EE de Java empaquetada en un archivo tipo .war, para desplegar en un servidor que soporte Servlets.

La Figura 8.5 muestra un ejemplo de un sitio web generado con el Composer.

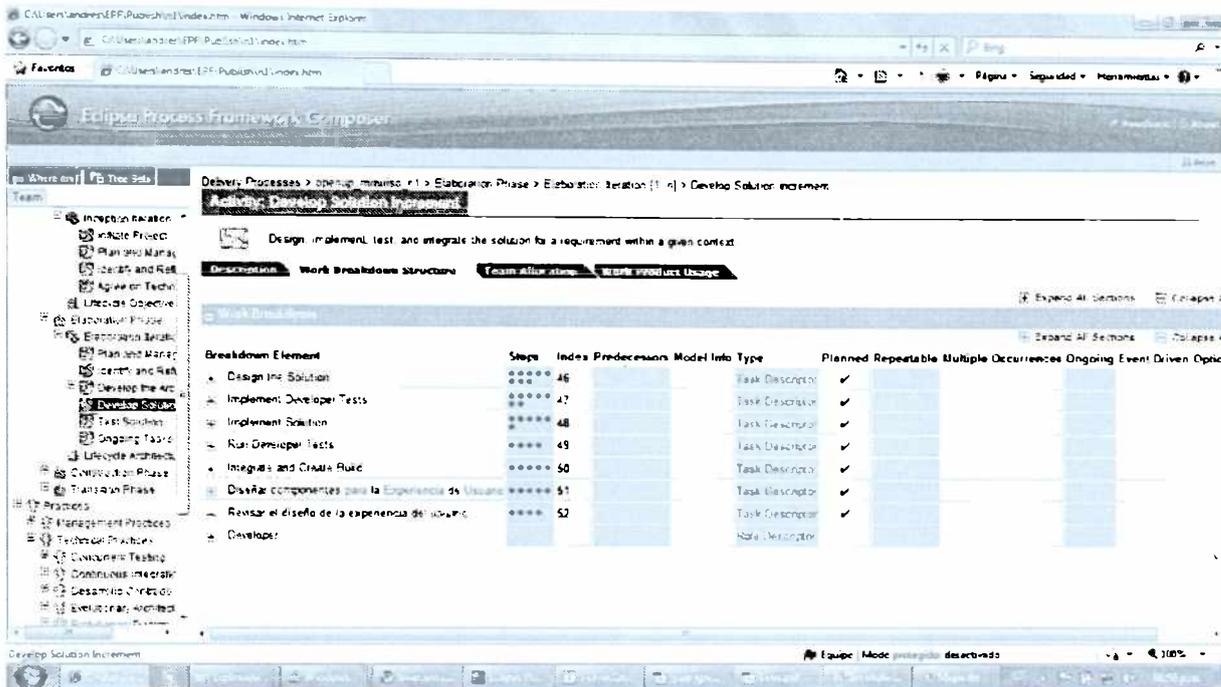


Figura 8.5: Ejemplo de sitio web generado con Composer

## 8.3. Plugins lógicos y físicos en la EPL

Para asegurar que los plugins dentro de la EPL se comportan adecuadamente juntos y maximizar el reuso de elementos, el EPF describe algunos tipos específicos de plugins. La intención es facilitar la distribución en la generación de contenidos de métodos y aumentar la capacidad de plug-and-play de los mismos. Para ello, los plugins separan áreas de concern siguiendo una estructura de empaquetado consistente.

La Figura 8.6 resume los tipos de plugins “lógicos” que se definen en la EPL y las dependencias permitidas entre ellos. Como veremos más adelante y muestra la Figura 8.7 cada uno de ellos puede ser representado por varias partes físicas de plugin.

### Tipos principales de plugin

Los plugins **Core**, contienen elementos dirigidos a ser compartidos entre otros plugins en el framework. Brindan la base del framework y permiten su reutilización mediante otros elementos como prácticas y procesos. Incluyen elementos que sirven de interfaz entre prácticas (Slots de Productos de Trabajo) y elementos reusables claves como productos de trabajo u orientaciones que deben ser compartidas por varias prácticas.

Los plugins de **Práctica** contienen los elementos de una práctica en particular. Existe un plugin por práctica. Sólo dependen de los plugins Core.

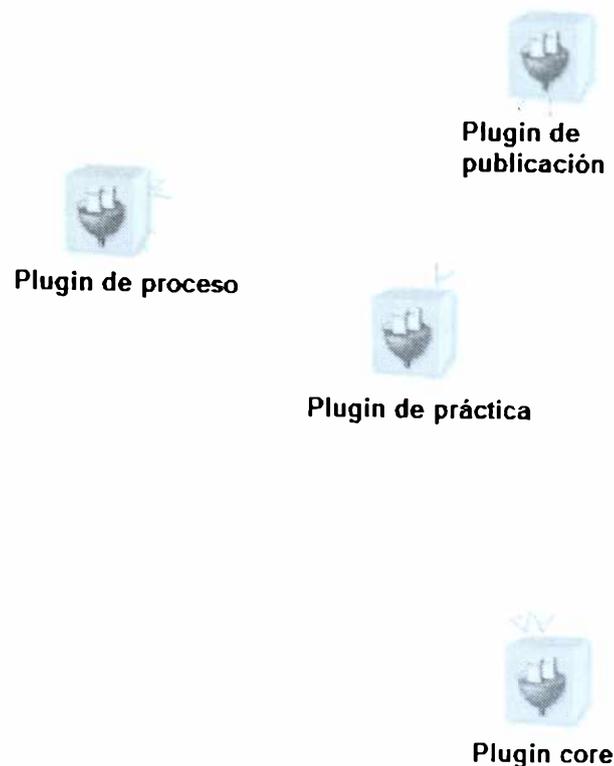


Figura 8.6: Tipos de plugins en EPL y relaciones entre ellos

Los plugins de **Proceso** contienen los elementos que definen procesos transversales a varias prácticas y que deben compartirse entre las configuraciones (los procesos que son específicos de una práctica son contenidos en los plugins de Práctica y los que son específicos de una configuración en plugins de Publicación). Los procesos cross-práctica se ensamblan con elementos contenidos en plugin de Práctica (tareas o procesos). Puede haber varios plugins de proceso. Pueden depender de plugins Core (para los elementos compartidos), de Práctica (para elementos específicos de una práctica) y de Proceso (para procesos compartidos).

Los plugins de **Publicación** contienen elementos de método que son únicos a una configuración específica que será publicada. Por ejemplo, vistas de navegación, hojas de ruta, etcétera.

### Subtipos de plugin Core

Existen varios elementos que son compartidos por todo el framework y por lo tanto se incluyen en los plugins Core. Sin embargo, algunos de ellos necesitan ser reutilizados en forma independiente. Por ejemplo, cuando se comparten las definiciones de herramientas, pero se necesita crear nuevas definiciones de categorías. Para ello se agregan los subtipos de plugin Core

- Los plugins **Slot** contienen definiciones de Slot de Productos de Trabajo y las orientaciones asociadas.
- Los plugins **Comunes** contienen los elementos reutilizables como productos de trabajo u orientaciones para compartir entre plugins.
- Los plugins de **Definiciones de Rol** contienen los roles o conjunto de ellos para compartir entre prácticas. No contiene asignaciones de esos roles a productos de trabajo o tareas ya que el framework utiliza el concepto de Asignación diferida.

- Los plugins de **Definición de categoría** contienen definiciones de disciplina, dominio, clase de producto de trabajo y conjunto de roles.
- Los plugins de **Definición de herramientas** definen las herramientas y mentores para su uso.
- Los plugins de **Definición de vistas y navegación** presentan las definiciones de vistas para publicar configuraciones.
- Los plugins de **Derecho de copia de release** presentan los elementos sobre derechos de copia para la release. Cada plugin debería tener un plugin de este tipo asociado.

## Partes de plugins

Para aumentar la configurabilidad y navegabilidad de la EPL, este concepto de partes permite que cada tipo de plugin se implemente en la práctica como uno o más plugins físicos.

La Figura 8.7 muestra las partes de plugin y sus relaciones. En este diagrama [Plugin X], [Plugin Y] y [Plugin Z] representan plugins lógicos.

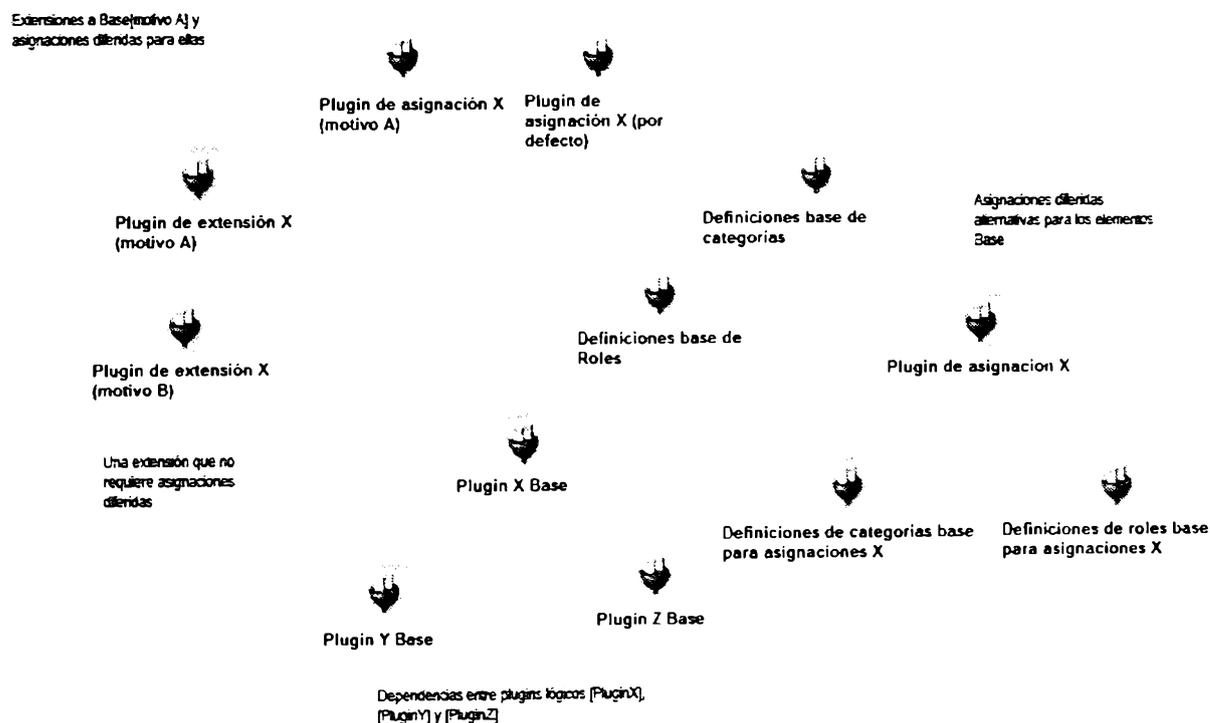


Figura 8.7: Partes de un plugin y sus relaciones

Los plugins **Base** contienen las definiciones básicas de los elementos de método del plugin lógico. Un plugin lógico tiene sólo uno Base. Estos pueden depender de otro Base (estas dependencias representan las dependencias entre los plugins lógicos).

Los plugins **De extensión** contienen elementos que extienden o amplían los elementos del Base. Un plugin lógico puede carecer de plugins Extiende o tener varios. Estos dependen de los Base que contienen los elementos a extender.

Los plugins **De asignación** contienen las asignaciones de los elementos de método en un plugin Base o Extiende cuando es soportada la asignación diferida. Un plugin lógico

puede carecer o tener varios de los plugins Asigna. Estos dependen del Base o Extiende que contiene las definiciones de los elementos asignados.

Al igual que con los plugins lógicos, los principales beneficios de esta organización física son la separación de concerns entre las definiciones de los elementos base del método, las extensiones de esas definiciones y su asignación diferida, la facilidad para definir asignaciones alternativas y la facilidad para generar configuraciones mediante selección de elementos.

## 8.4. La familia de plugins OpenUP/MMU-ISO

En el Capítulo anterior presentamos la propuesta de extensión de la EPL y de la configuración OpenUP/Basic para dar soporte a procesos de desarrollo que se basen en el Proceso Unificado y que resulten conformes al Modelo de Madurez de usabilidad ISO.

Organizamos estas extensiones en cuatro plugins lógicos que corresponden a cada una de esas extensiones:

- **Un plugin del tipo Práctica** que contiene las definiciones del contenido de método de la práctica (tareas, productos de trabajo y roles) DCU detallados en el capítulo 7.
- **Un plugin del tipo Publicación** para la Configuración OpenUP/MMU-ISO N1: extiende la configuración OpenUP/Basic para incorporar los elementos de la Práctica DCU de manera de poder instanciar un proceso que resulte conforme al MMU-ISO en el Nivel 1 de capacidad o madurez.
- **Un plugin del tipo Publicación** para la Configuración OpenUP/MMU-ISO N2: extiende la configuración OpenUP/MMU-ISO N1 para llevar el proceso al Nivel 2 de capacidad o madurez en el MMU-ISO.
- **Un plugin del tipo Publicación** para la Configuración OpenUP/MMU-ISO N3: de forma similar al anterior, extiende la configuración OpenUP/MMU-ISO N2 para permitir la implementación de un proceso de Nivel 3 en el MMU-ISO.

Cada uno de estos plugins lógicos es implementado en el Composer como un conjunto de partes o plugins físicos, siguiendo las guías señaladas en la sección anterior. A continuación describimos la composición de las partes físicas que componen cada uno de estos plugins lógicos.

### 8.4.1. El plugin de la Práctica DesCU

El plugin de la Práctica DesCU se agrega en el árbol de plugins del Composer como muestra la Figura 8.8 dentro de las prácticas técnicas, junto otras como testing concurrente, desarrollo centrado en casos de uso, integración continua, etc.

Este plugin lógico está implementado como dos plugins físicos uno de Base y otro de Asignación, como se indica en la Figura 8.9. El primero contiene las definiciones básicas de Tareas, Productos de Trabajo, Roles y Orientaciones. También se incluirán en él las Orientaciones relacionadas con derechos de copia y la categorización de los elementos definidos. Las asignaciones de los Roles a las Tareas y los Productos de trabajo se ubican en el segundo plugin físico.

La definición de la Práctica DesCU en el Capítulo 7 no incluye procesos propios (ni patrones de capacidad, ni procesos de entrega) de modo que sólo aparecen en estos plugins elementos de contenido de método.

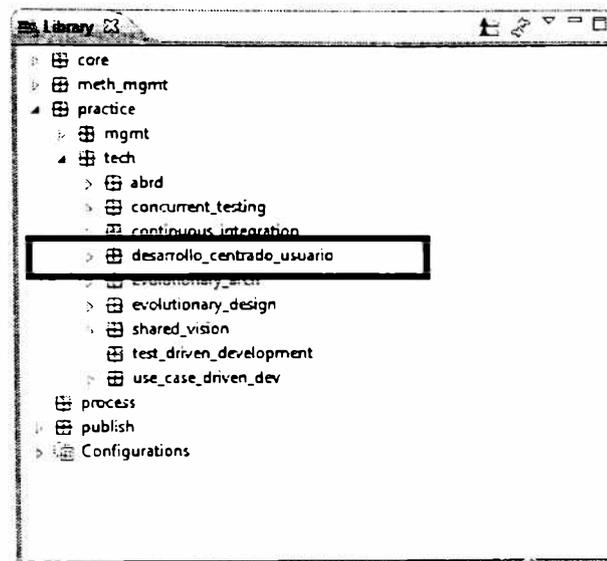


Figura 8.8: Ubicación del Plugin Práctica DCU en Composer

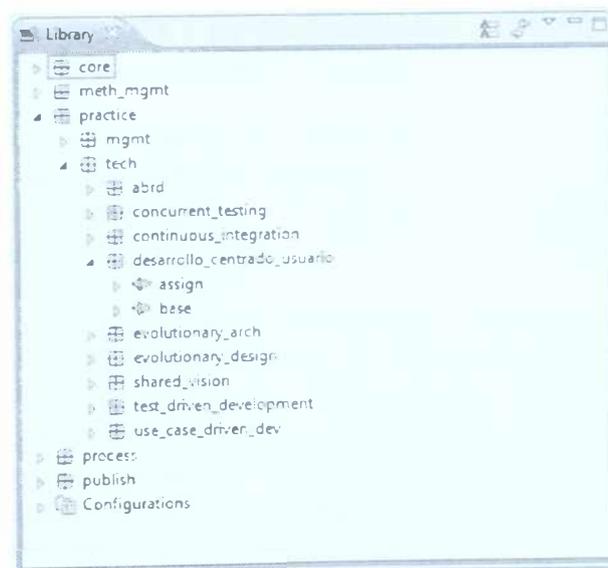


Figura 8.9: Plugins físicos de la Práctica DCU en Composer

## El plugin Base

En la Figura 8.10 se muestra la organización interna del plugin Base (`practice.tech.desarrollo_centrado_usuario.base`)

El plugin Base se apoya en las definiciones compartidas para todas las prácticas de la EPL, en particular en los siguientes plugins y sus elementos que se indican:

- `core.default.cat_def.base`: provee las definiciones de Dominios y Disciplinas (arquitectura, desarrollo, gestión de proyecto, testing, requerimientos).
- `core.default.role_def.base`: provee las definiciones de los Roles básicos de la EPL (analista, arquitecto, desarrollador, gerente de proyecto, stakeholder, tester, rol genérico).
- `core.default.tool_def.base`: provee el conjunto de herramientas básicas para las prácticas, que consiste en tres orientaciones sobre el uso del Composer y la personalización de métodos.
- `core.default.uma_concept.base`: contiene las definiciones básicas para comprender la estructura de un proceso y navegarlo de acuerdo con la Unified Method Architecture.
- `core.gen.common.base`: define productos de trabajo y orientaciones genéricos para ser utilizados por todas las prácticas (como el glosario).
- `core.mgmt.slot.base`: donde se ubican las definiciones de Slots de Productos de Trabajo (definición y alcance de proyecto, riesgo de proyecto, trabajo de proyecto, estado de proyecto) y Orientaciones de gestión (orientación para la planificación, orientación para la asignación y orientación para la colaboración).
- `core.tech.common.base`: contiene definiciones técnicas provistas por IBM que extienden el contenido open source de la EPL.
- `core.tech.slot.base`: define los Slots de Productos de Trabajo (arquitectura, diseño, implementación, especificación y resultados de prueba) y Orientaciones técnicas (orientación para diseño, orientación para pruebas).

La organización de los componentes en el Plugin Base, tal como se muestra en la Figura 8.10, consiste en dos paquetes de Contenido de Método, denominados Common y Release.info siguiendo las guías de estilo de la EPL. En el primer paquete se agrupan las definiciones de Tareas, Productos de Trabajo, Roles y Orientaciones. En el segundo, los contenidos sobre versiones de la Práctica.

En el paquete Common, organizados en el conjunto de Roles DCU, se incluye la definición de los siete roles de la Práctica: Diseñador de la UX, Especialista de UX, Tester de UX, Patrocinador, Líder técnico de dominio, Representante de Usuarios y Usuario final. En el Capítulo 7 todos se definen como subtipos de alguno de los roles básicos. Para implementarlo, utilizamos las opciones de variabilidad de contenido que ofrece la EPL.

Los seis primeros utilizan la variabilidad *Extiende*, en el caso del Usuario Final el plugin hace uso de la variabilidad *Extiende y reemplaza* tal como se muestra en el diagrama de la Figura 8.11.

Para las definiciones de cada rol utilizamos las descripciones, habilidades y detalles incluidos en el capítulo 7 tal como se muestra en las figuras 8.12 y 8.13 para el Diseñador de la UX y el Tester de la UX.

En la Práctica DesCU se definen estos productos de trabajo propios: Modelo de usuario, Meta de usabilidad, Modelo de tareas, Concepto de UX, Prototipo de Experiencia de Usuario, Storyboard de Experiencia de Usuario, Mapa de Navegación, Documento de



Figura 8.10: Estructura interna del plugin Base en la Práctica DCU

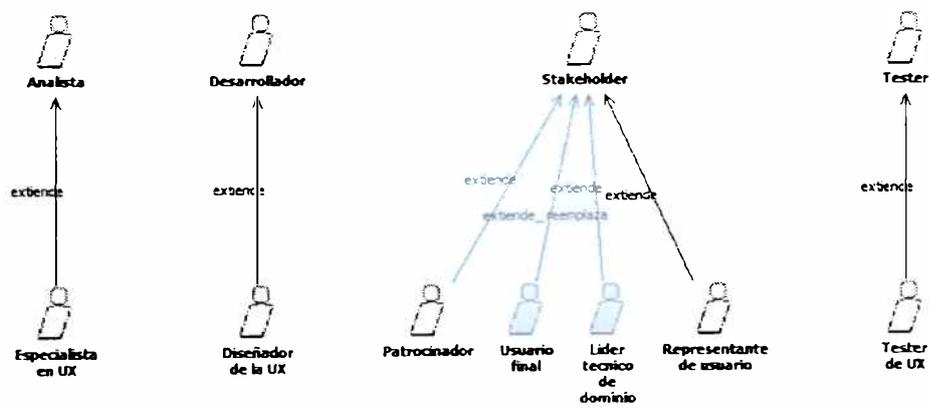


Figura 8.11: Relación entre roles básicos de EPL y roles de DCU

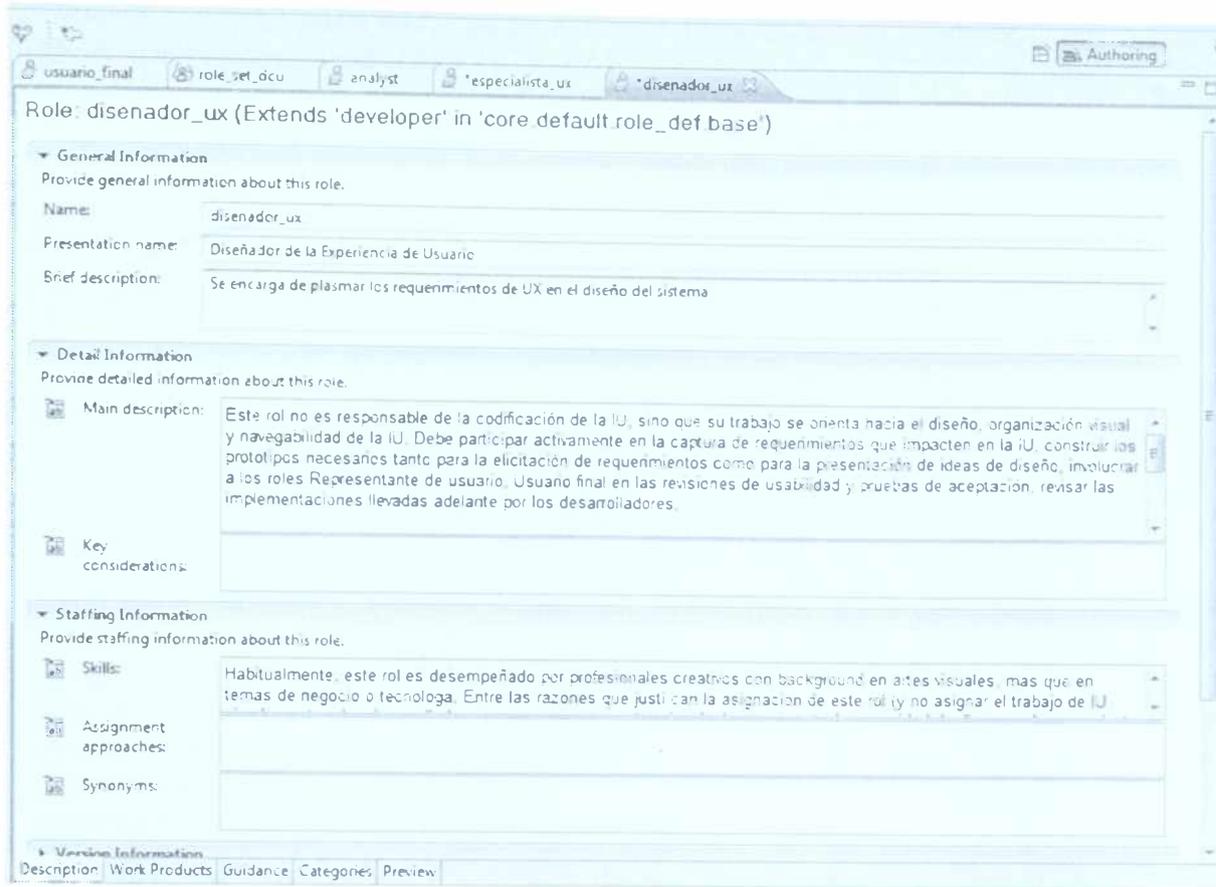


Figura 8.12: Descripción de Diseñador de UX

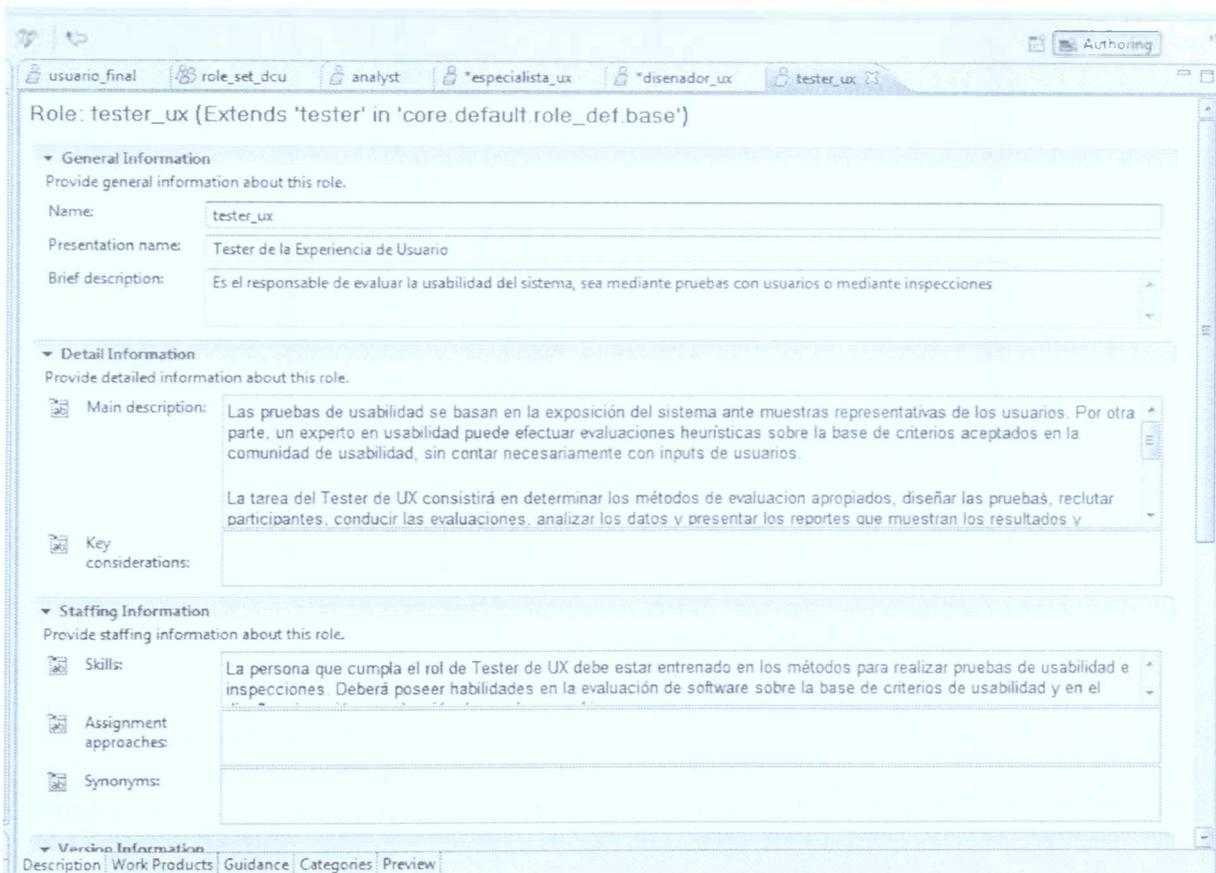


Figura 8.13: Descripción de Tester de UX

Usuario. Los ocho están incluidos en el Plugin Base, cada uno identificado con un Slot de Producto Técnico (ver Tabla 8.1)

Producto de trabajo	Slot
Meta de usabilidad	Especificación técnica
Modelo de usuario	Especificación técnica
Modelo de tareas	Especificación técnica
Concepto de UX	Diseño técnico
Prototipo de UX	Diseño técnico
Storyboard de UX	Diseño técnico
Mapa de navegación	Diseño técnico
Documento de usuario y entrenamiento	Documentación de usuario

Tabla 8.1: Relación entre Productos de Trabajo y Slots

A partir de estos Productos de Trabajo es necesario incluir las Tareas que los manipularán ya sea en condición de input o output. En la Figura 8.14 se muestra el listado completo en el árbol de Contenido de Método del plugin y la Tabla 8.2 su distribución por Disciplina. Como se observa en esta tabla, es necesario incorporar una disciplina más al conjunto básico provisto por la EPL. Se trata de la Disciplina Despliegue que contiene las actividades necesarias para hacer disponible al sistema para los usuarios (instalaciones, material de entrenamiento, capacitaciones, etc.).

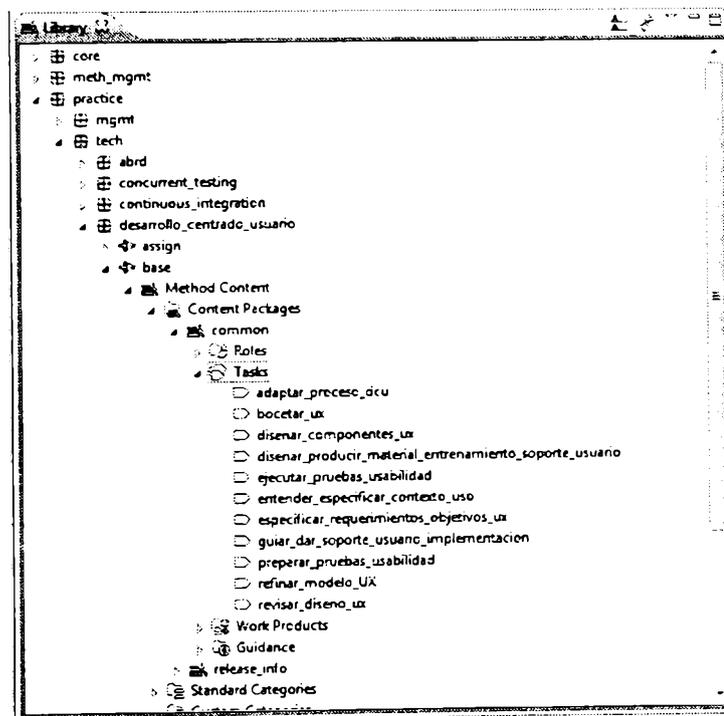


Figura 8.14: Definiciones de Tarea incluidas en el Plugin Base de la Práctica DCU

Todas las definiciones utilizan los contenidos incluidos en el Capítulo 7. Como ejemplo, en las Figuras 8.15, 8.16, 8.17, 8.18 y 8.19 se presentan las solapas de Descripción, Pasos, Productos de Trabajo, Categorías y Previsualización para la Tarea Entender y especificar el contexto de uso.

El Plugin Base se completa con la Orientación que presenta la Práctica, con su descripción completa (Propósito, Metas, Background, Niveles de Adopción) y los elementos articulados por la misma.



Figura 8.15: Definición de Tarea Entender y Especificar Contexto de Uso. Solapa Descripción

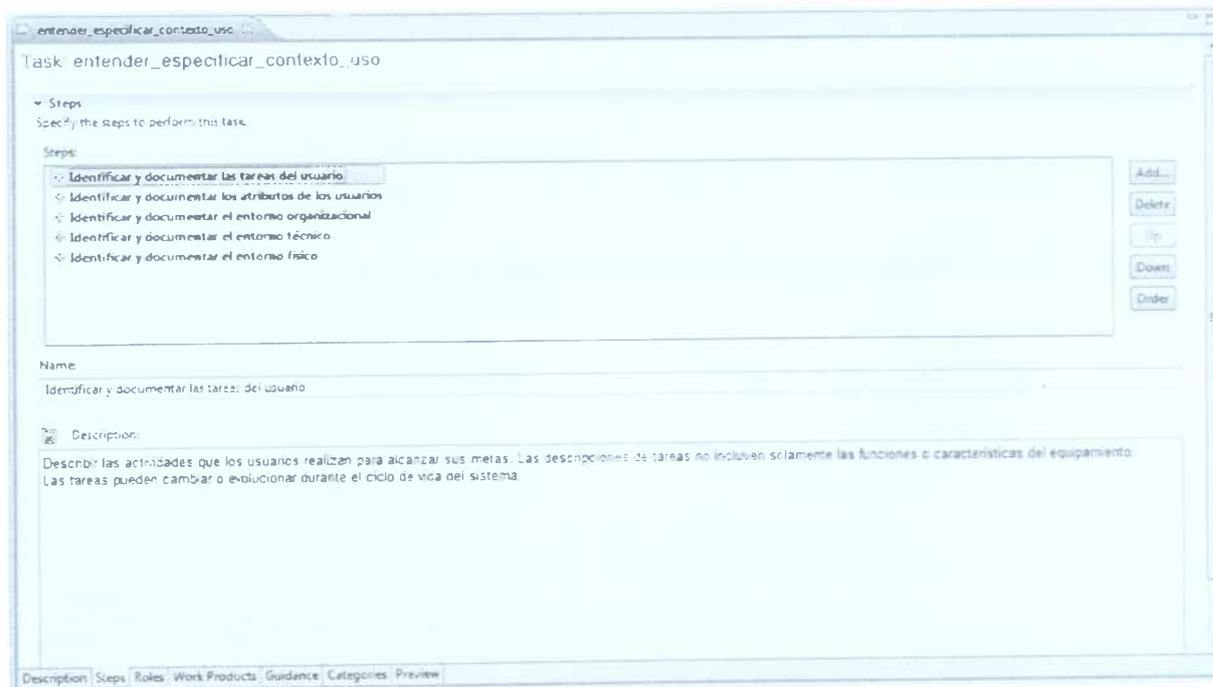


Figura 8.16: Definición de Tarea Entender y Especificar Contexto de Uso. Solapa Pasos

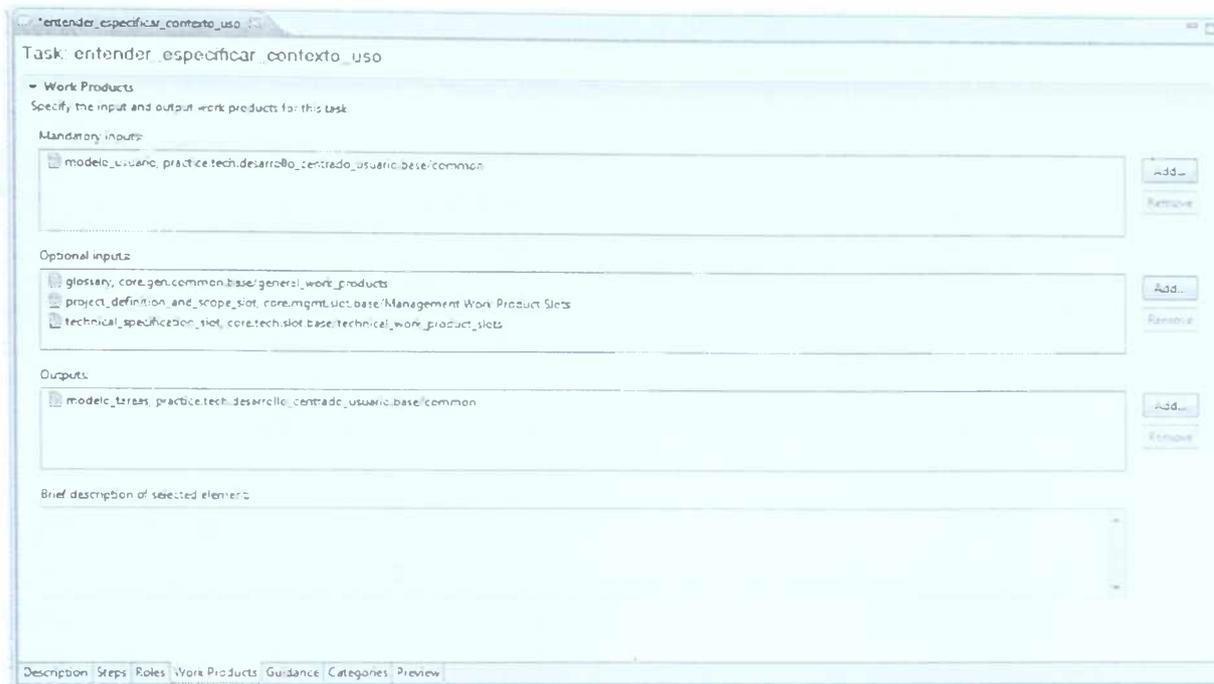


Figura 8.17: Definición de Tarea Entender y Especificar Contexto de Uso. Solapa Productos de Trabajo

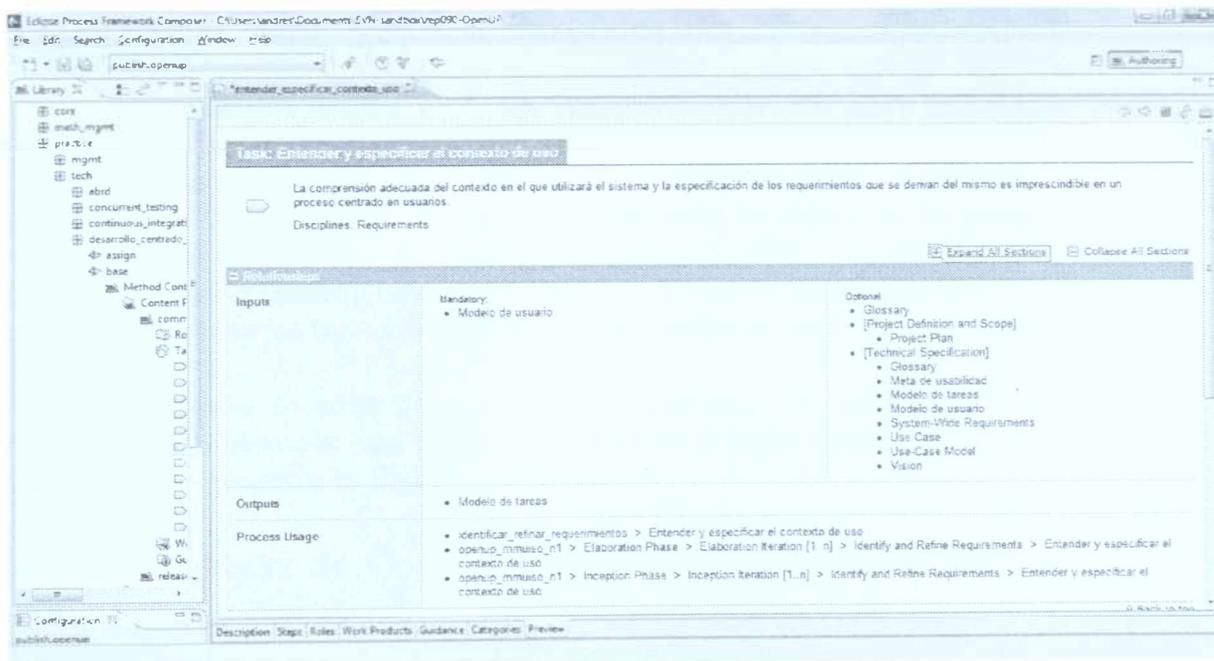


Figura 8.18: Definición de Tarea Entender y Especificar Contexto de Uso. Solapa Previsualización

Tarea	Disciplina
Adaptar el Proceso DCU	Gestión de proyecto
Diseñar la UX	Arquitectura
Diseñar componentes de interacción	Desarrollo
Diseñar y producir material de entrenamiento y soporte de usuario	Despliegue
Ejecutar pruebas de usabilidad	Testing
Entender y especificar el Contexto de Uso	Requerimientos
Especificar requerimientos y objetivos de UX	Requerimientos
Guiar y dar soporte al usuario en implementación	Despliegue
Preparar pruebas de usabilidad	Testing
Refinar el modelo de UX	Arquitectura
Revisar el diseño de UX	Desarrollo

Tabla 8.2: Relación de las Tareas en Práctica DCU y las Disciplinas de la EPL

### El plugin Assign

Como dijimos más arriba, el plugin de la Práctica se completa con la parte de Asignación (denominado `practice.tech.desarrollo.centrado.usuario.assign`) que obviamente contiene las relaciones entre los Roles y las Tareas definidos por la Práctica DCU. La organización del plugin es similar al Base. Un paquete de contenido (denominado `asignacion_rol`) incluye las definiciones de los tipos de elementos a vincular.

Para realizar la asociación se utiliza la variabilidad de contenido *Contribuye*, que ya describimos. Se incluyen dos contribuciones a los roles definidos, indicando en cada contribución la responsabilidad sobre productos de trabajo de la Práctica:

- `diseñador_ux.asigna.wp` contribuye a `diseñador_ux` y es responsable de Mapa de navegación, Prototipo de UX y Storyboard de UX.
- `especialista_ux.asigna.wp` contribuye a `especialista_ux` y es responsable de Meta de Usabilidad, Modelo de Tareas y Modelo de Usuario.

La previsualización permite confirmar las relaciones establecidas, tal como se presenta en la Figura 8.20 y 8.21

Del mismo modo que para establecer las relaciones entre Roles y Productos de Trabajo, a asignación de Tareas a los Roles se realiza mediante la definición de nuevas tareas que contribuyen a las básicas de la práctica y que aportan fundamentalmente la relación de una Tarea con quienes la realizan tanto en su carácter de figuras primarias como adicionales. La Figura 8.22 presenta las tareas agregadas, por el nombre es fácil describir a cuál contribuye cada una.

Como cada una de estas nuevas tareas sólo persigue el objetivo de contribuir una relación de asignación con uno o más roles, éste es el único dato que completamos en su definición, como muestra la Figura 8.23 y 8.24

#### 8.4.2. El plugin de Configuración OpenUP/MMU-ISO N1

Este plugin extiende la configuración OpenUP/Basic y le incorpora los elementos de la Práctica Desarrollo Centrado en Usuario para permitir la instanciación de procesos conformes al Modelo de Madurez en Usabilidad ISO con un nivel 1 de Capacidad.

Este Plugin lógico está organizado en tres elementos físicos:

- Un plugin de Proceso: `process.openup.mmu-iso_n1`
- Un plugin de Publicación: `publish.openup-mmuiso.n1.base`

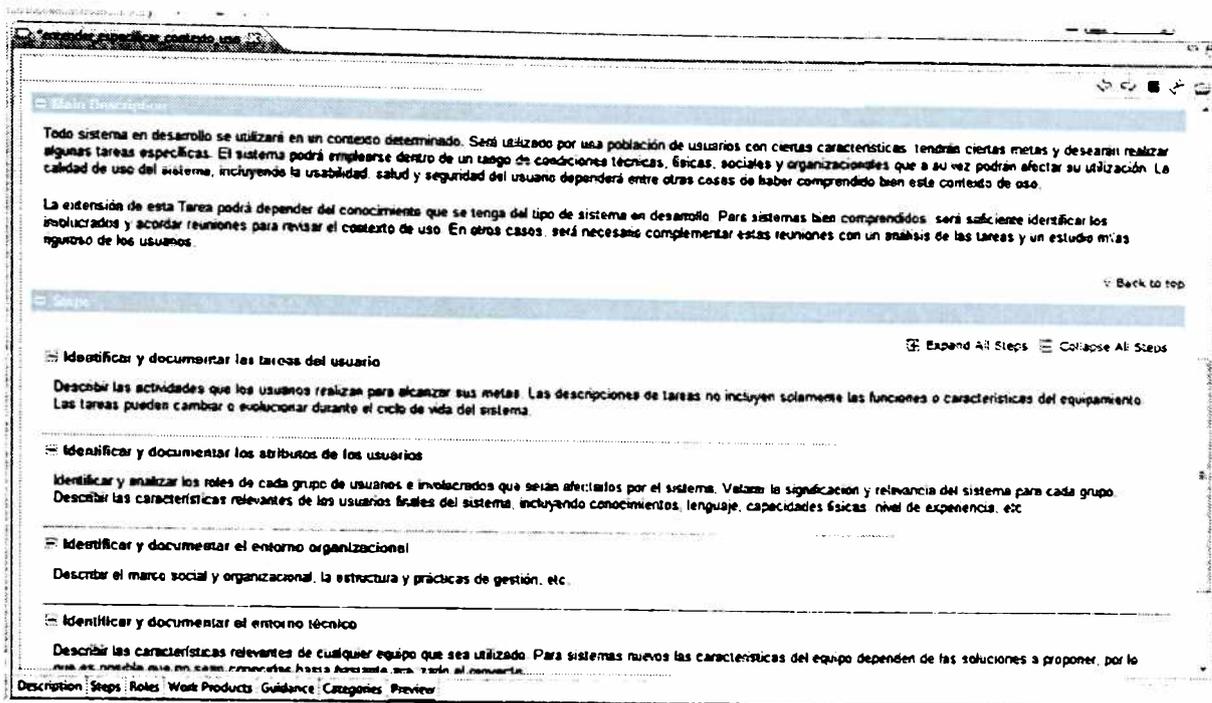


Figura 8.19: Definición de Tarea Entender y Especificar Contexto de Uso. Solapa Previsualización

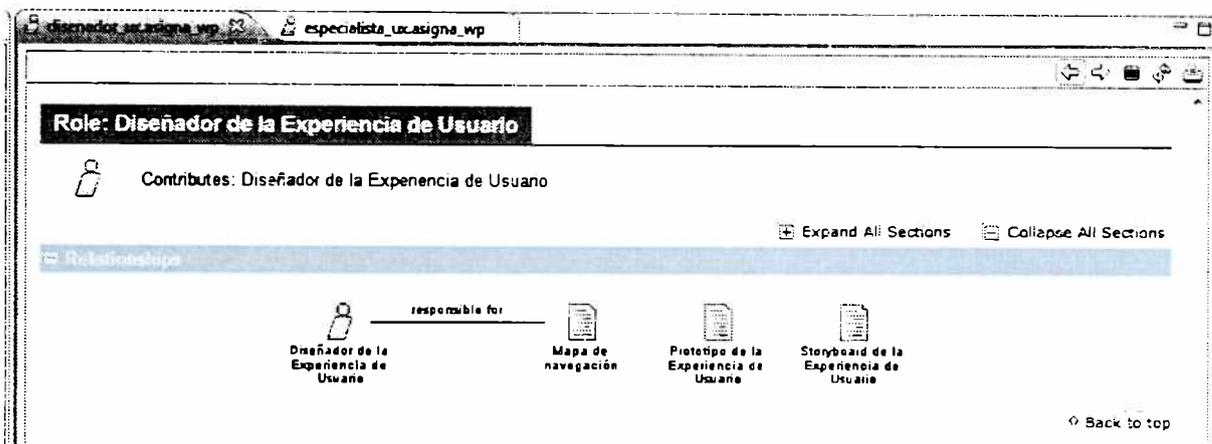


Figura 8.20: Responsabilidad de Productos de Trabajo del Diseñador de UX

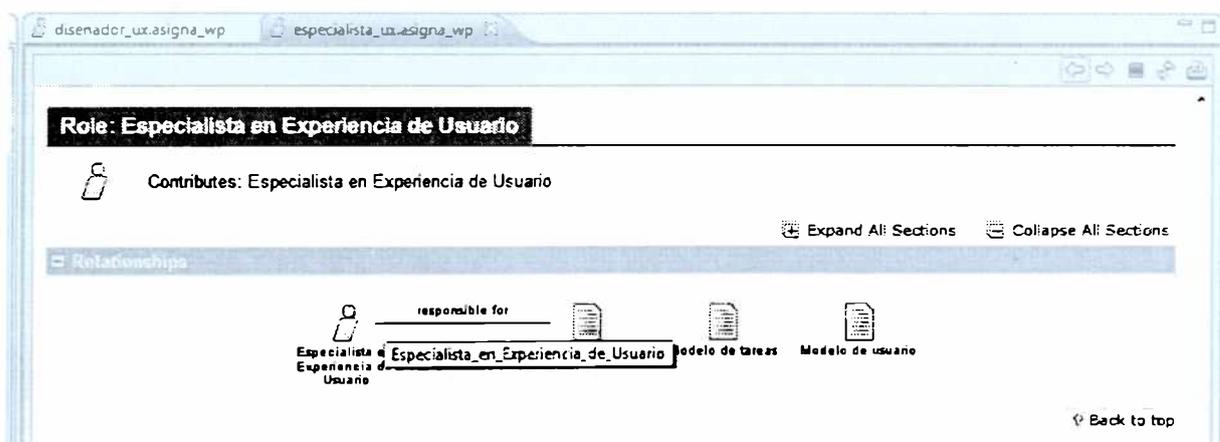


Figura 8.21: Responsabilidad de Productos de Trabajo del Especialista de UX

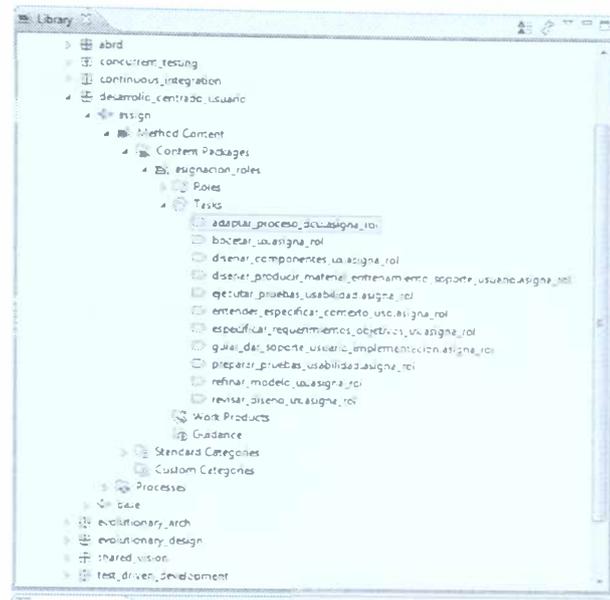


Figura 8.22: Tareas definidas para contribuir asignaciones a las básicas

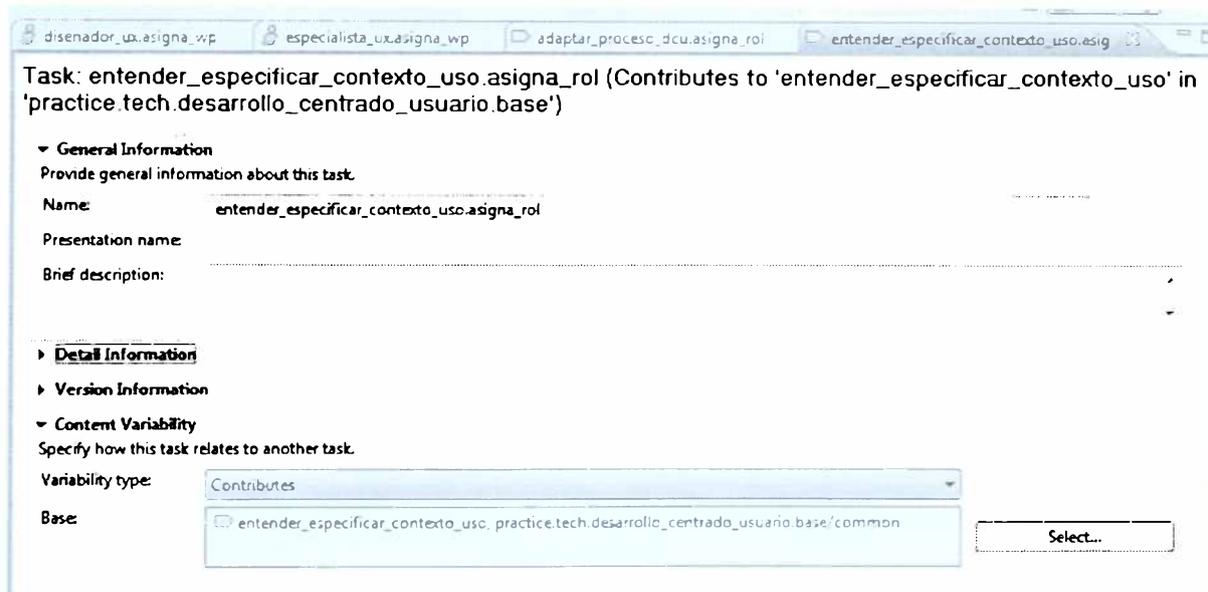


Figura 8.23: Asignación de roles a Entender y especificar Contexto de Uso mediante contribución

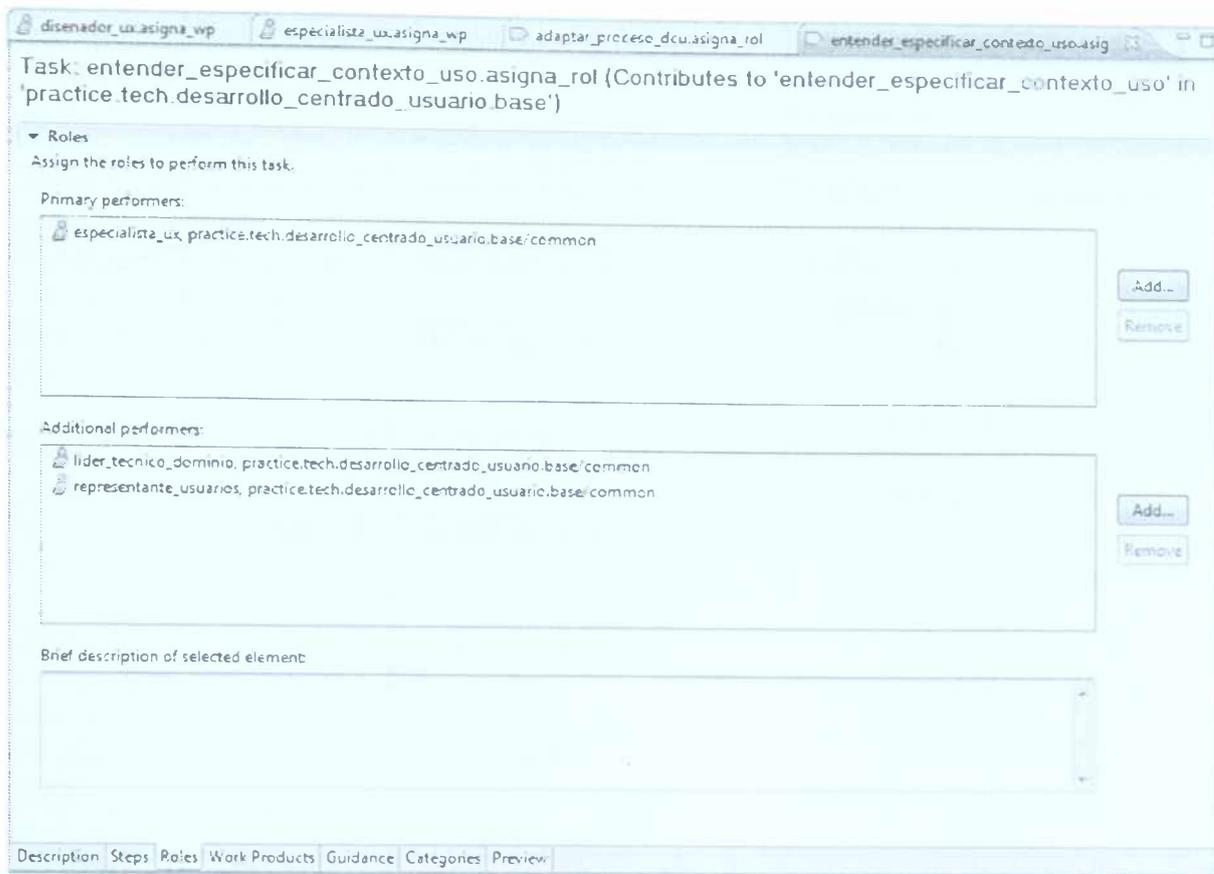


Figura 8.24: Asignación de roles a Entender y especificar Contexto de Uso mediante contribución

- Una Configuración de método: `publish.openup-mmu-iso_n1`

La Figura 8.25 muestra la ubicación de estos componentes en la estructura de plugins de la EPL.

El plugin Base de proceso, hace referencia a la base de OpenUP y a las dos partes físicas de la Práctica DCU.

En el paquete de contenidos propio del plugin, se agregan cinco definiciones de tareas necesarias para extender las de OpenUP base y permitir la conformación de un proceso conforme a MMU-ISO. Las modificaciones que se incluyen, en todos los casos utilizando la variabilidad *Contribuye*, son las siguientes:

- Desarrollar Visión Técnica: se agrega el paso Identificar y bocetar el contexto de uso.
- Evaluar resultados: agrega como realizadores adicionales al Líder técnico de dominio, Patrocinador y Representante de usuarios.
- Gestionar iteración: agrega como realizadores adicionales al Líder técnico de dominio y al Patrocinador.
- Planificar iteración: agrega como realizador adicional al Líder técnico de dominio
- Planificar el proyecto: agrega como realizadores adicionales al Líder técnico de dominio y al Patrocinador.

En el paquete de procesos del plugin se modifican varios de los patrones de capacidad para reflejar estas incorporaciones y otros cambios necesarios en el Proceso de entrega general, tal como se indicó en la descripción de la Configuración en el Capítulo 7.

Tarea	Realizador primario	Realizador adicional
Adaptar el proceso DCU	Gerente de proyecto	Lider técnico de dominio, Patrocinador
Diseñar la UX	Diseñador de UX	Especialista de UX
Diseñar componentes de la UX	Diseñador de UX	Desarrollador, Representante de Usuarios
Diseñar y producir material de entrenamiento	Especialista de UX	Lider técnico de dominio, Representante de usuarios
Ejecutar pruebas de usabilidad	Tester de UX	Usuario final, Representante de usuarios, Especialista de UX
Entender y especificar Contexto de Uso	Especialista de UX	Lider técnico de dominio, Representante de usuarios
Especificar requerimientos y objetivos de UX	Especialista de UX	Lider técnico de dominio, Representante de usuarios, Usuario final
Guiar y dar soporte a usuario en implementación	Especialista de UX	Lider técnico de dominio, Representante de usuarios, Usuario final
Preparar pruebas de usabilidad	Tester de UX	Especialista de UX, Representante de usuarios
Refinar modelo de UX	Diseñador de UX	
Revisar diseño de UX	Tester de UX	Diseñador de UX, Especialista de UX

Tabla 8.3: Asignaciones de Roles a Tareas de DCU

En la Actividad Identificar y Bocetar requerimientos se incluyen las Tareas específicas de los procesos DCU: Entender y especificar contexto de uso y Especificar requerimientos y metas de la UX.

La Tarea Diseñar la Experiencia de Usuario se incorpora en el patrón de capacidad Acordar el abordaje técnico (que contribuye a Agree on technical approach). De esta manera, esta actividad concentra las actividades de diseño de alto nivel tanto respecto de la funcionalidad del sistema como de la experiencia de usuario requerida.

De la misma forma, la Actividad Desarrollar la arquitectura, incorpora la tarea Diseñar el Modelo de la UX, para mantener durante las primeras fases el foco tanto en el nivel micro del diseño de componentes UX como en el macro del modelo general de la misma. Ya vimos en los capítulos previos la estrecha relación que existe entre las definiciones en el nivel de la arquitectura del software y los requerimientos y posibilidades de usabilidad del producto final.

La Actividad Desarrollar un Incremento de solución incluye las tareas del diseño de componentes de UX y su revisión.

En el patrón de capacidad Testear la solución se incorporan las Tareas Preparar pruebas de usabilidad y Ejecutar pruebas de usabilidad.

Finalmente, será necesario incluir un nuevo patrón de capacidad para dar soporte a la Actividad incluida en la descripción del Capítulo 7, Guiar y dar soporte al usuario, que se compone de las dos tareas previstas para las fases de implementación del sistema o producto: Producir material de entrenamiento y soporte al usuario y Guiar y dar soporte al usuario en implementación.

A partir de estos cambios, se utiliza el Patrón de Entrega de OpenUP/Basic como base para proponer el proceso OpenUP/MMU-ISO N1 que incluye las mismas cuatro fases del original (Inicio, Elaboración, Construcción y Transición). La estructura e hitos de cada

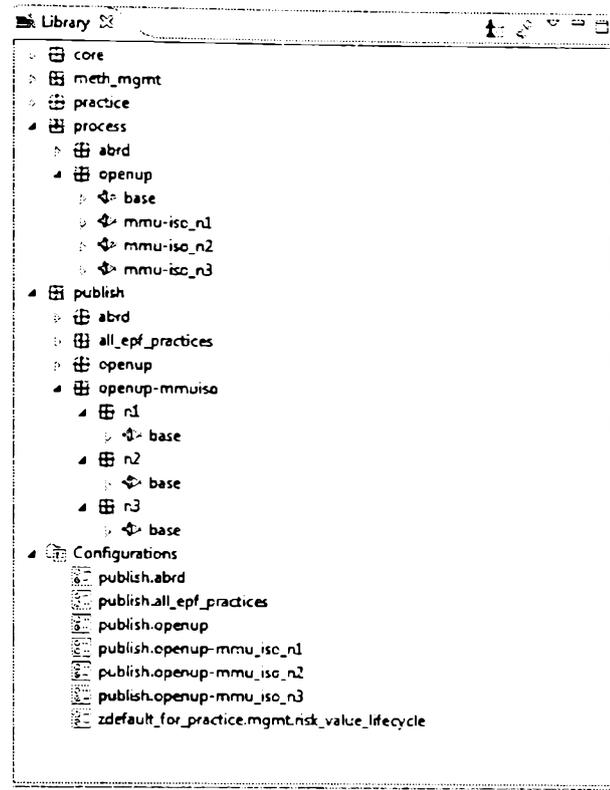


Figura 8.25: Ubicación de plugins físicos para configuraciones OpenUP/MMU-ISO Nx

fase siguen siendo los mismos, salvo para la fase de Transición donde se agrega el patrón de capacidad mencionado en el párrafo anterior, tal como muestra la Figura 8.26.

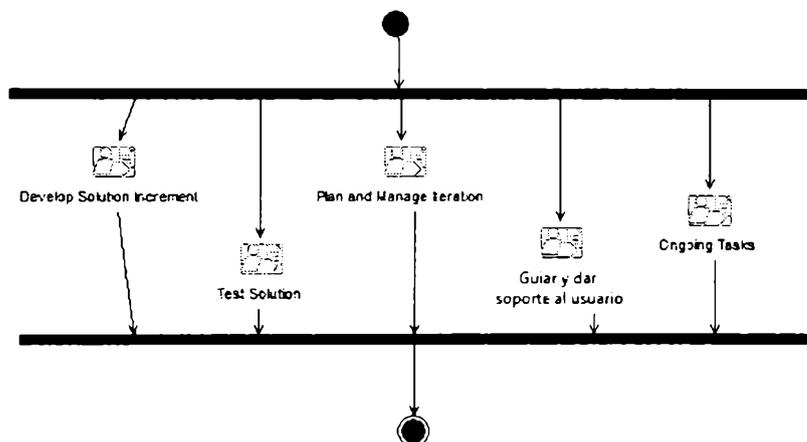


Figura 8.26: Iteración tipo en fase de Transición de OpenUP/MMU-ISO N1

La Configuración toma como referencia los siguientes elementos:

- Los plugins Core con los elementos básicos para constituir la EPL
- Todas las prácticas incluidas en la EPL, con la única excepción de Agile Business Rule Development (ABRD). Es particularmente importante la inclusión de Desarrollo Centrado en Usuario del grupo de prácticas técnicas.
- Los plugins de procesos OpenUP, particularmente el mmu-iso\_n1. Del plugin OpenUP se excluye el proceso de entrega con su ciclo de vida, ya que será redefinido como el

ciclo OpenUP/MMU-ISO N1.

- Los plugins de publicación que contienen los elementos y vistas a presentar en web: OpenUP y OpenUP-MMUIISO.N1.

Utilizamos las mismas vistas de navegación para el sitio web a publicar, incorporando en algunas de ellas información específica del Modelo de Madurez de Usabilidad ISO o de la adaptación OpenUP/MMU-ISO para ayudar al ingeniero de procesos en la adopción del proceso. En particular, se incluyen las orientaciones definidas en la Práctica DCU como Presentación y Conceptos de la Práctica DCU, Presentación del MMMU-ISO y la Hoja de Ruta para adopción.

### 8.4.3. El plugin de Configuración OpenUP/MMU-ISO N2

Este plugin extiende la configuración OpenUP/MMU-ISO N1 y le incorpora los elementos de la Práctica Desarrollo Centrado en Usuario para permitir la instanciación de procesos conformes al Modelo de Madurez en Usabilidad ISO con un nivel 2 de Capacidad.

Este Plugin lógico está organizado en tres elementos físicos:

- Un plugin de Proceso: `process.openup.mmu-iso_n2`
- Un plugin de Publicación: `publish.openup-mmuiso.n2.base`
- Una Configuración de método: `publish.openup-mmu_iso_n2`

Hemos visto que en este nivel de capacidad interesa demostrar que los procesos se gestionan para producir los productos en tiempo establecido y dentro de los requerimientos especificados y que los productos de trabajo son documentados y controlados para satisfacer requerimientos funcionales y no funcionales. En el Apéndice B se detallan las prácticas asociadas a estos dos objetivos.

En el plugin de la configuración incorporamos modificaciones en dos Actividades: Iniciar el Proyecto y Planificar y gestionar la iteración tal como se propuso en el Capítulo 7.

En el plugin de proceso se agregan cuatro tareas que contribuyen respectivamente a otras tantas del proceso OpenUP/MMU-ISO N1 con pasos adicionales que generen evidencia de la gestión DCU necesaria.

- `Planificar_proyecto_mmuiso_n2`, contribuye a Planificar el proyecto, con el paso Definir productos DCU a utilizar en proyecto.
- `Planificar_iteracion_mmuiso_n2`, contribuye a Planificar la iteración con dos pasos: Establecer los recursos necesarios para las actividades DCU y Establecer requerimientos de calidad de productos de trabajo DCU.
- `Gestionar_iteracion_mmuiso_n2`, contribuye a Gestionar la iteración con el paso Monitorear métricas de calidad y usabilidad de productos de trabajo DCU.
- `Evaluar_resultados_mmuiso_n2`, contribuye a la tarea Evaluar resultados con el paso Ponderar resultados de pruebas de usabilidad.

### 8.4.4. El plugin de Configuración OpenUP/MMU-ISO N3

De manera similar al anterior, este plugin extiende la configuración OpenUP/MMU-ISO N2 y le incorpora los elementos de la Práctica Desarrollo Centrado en Usuario para permitir la instanciación de procesos conformes al Modelo de Madurez en Usabilidad ISO con un nivel 3 de Capacidad.

Este Plugin lógico está organizado en tres elementos físicos:

- Un plugin de Proceso: `process.openup.mmu-iso_n3`
- Un plugin de Publicación: `publish.openup-mmuiso_n3.base`
- Una Configuración de método: `publish.openup-mmuiso_n3`

El plugin de proceso contiene la incorporación de la Tarea Adaptar el proceso DCU en la Actividad Iniciar el Proyecto, tal como se muestra en el WBS del Proceso de Entrega OpenUP/MMU-ISO N3 en la Figura 8.27.

Presentation Name	Index	Pred...	Type	Planned	Repeat...	Multipl...	Ongoing	Event...
OpenUP/MMU-ISO N3	0		Delivery Pro...	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Inception Phase	1		Phase	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Inception Iteration [1..n]	2		Iteration	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Initiate Project	3		Activity	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Adaptar el proceso de Desarrollo Centrado en Usuario	4		Task Descr...	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Develop Technical Vision	5		Task Descr...	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Plan Project	6		Task Descr...	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Plan and Manage Iteration	7		Activity	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Identify and Refine Requirements	11	3	Activity	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Agree on Technical Approach	18	3	Activity	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Lifecycle Objectives Milestone	21	2	Milestone	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Elaboration Phase	22		Phase	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Elaboration Iteration [1..n]	23		Iteration	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Lifecycle Architecture Milestone	61	23	Milestone	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Construction Phase	62		Phase	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Construction Iteration [1..n]	63		Iteration	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Initial Operational Capability Milestone	88	63	Milestone	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Construction Iteration [1..n]	89		Iteration	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Transition Phase	114		Phase	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Transition Iteration [1..n]	115		Iteration	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Product Release Milestone	138	115	Milestone	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Figura 8.27: WBS del Proceso de Entrega OpenUP/MMU-ISO N3

La inclusión de esta tarea como parte de la actividad de lanzamiento del proyecto asegura la revisión y ajuste del proceso y genera evidencia de las prácticas de gestión de definición del proceso y sus recursos tal como se exige en este nivel de capacidad del MMU-ISO.

Esa tarea se complementa con pasos extras en las actividades de planificación y gestión de las iteraciones. En Evaluar resultados se incorpora el paso de revisar el proceso y proponer ajustes si es necesario y en Planificar la iteración se agrega el consecuente paso de incorporar los cambios al proceso que se sugieran en las evaluaciones.

# Capítulo 9

## Conclusiones

### Resumen

En esta Tesis hemos presentado una propuesta para un proceso de desarrollo de software que incorpore prácticas de diseño centrado en usuario y dé conformidad al Modelo de Madurez en Usabilidad de la ISO en los niveles de Capacidad 1, 2 o 3.

Hemos señalado la importancia que la usabilidad ha adquirido como atributo de calidad en los productos de software. También mostramos que tanto desde la Human Computer Interaction como desde la Ingeniería de Software se reconoce que es necesario involucrar activamente a los usuarios en el proceso de desarrollo para aumentar los atributos de usabilidad del producto final.

Pasamos revista a diferentes propuestas que provienen de ambos campos y que buscan implementar procesos de Diseño Centrados en Usuario. Sin embargo, encontramos que existen dificultades de integración entre IS y HCI, debidas a la falta de conocimiento mutuo entre ellas, la diferencia de enfoques sobre el mismo objeto de estudio, la dispar madurez de las disciplinas, etc. Una de las principales dificultades está planteada por el hecho de que el diseño y construcción de software está regido fundamentalmente por las prácticas y conceptos de la ingeniería del software al tiempo que los métodos, técnicas y herramientas de la HCI son poco conocidas o incomprendidas por los ingenieros de software. Esta situación deriva en una falta de comprensión adecuada de las implicancias de la usabilidad en todo el proceso y termina confinada a los aspectos relacionados con la organización visual de pantallas y la formulación del esquema de navegación.

En aquellos casos donde finalmente se instauran las prácticas básicas de diseño de interacción y usabilidad, con poco o nulo poder de decisión de sus actores, no existen guías para impulsar una mejora ordenada y previsible de los procesos que promueva logros de usabilidad más altos. Una de las formas de disponer de hojas de ruta para la mejora sistemática y previsible de procesos está dada por los Modelos de Capacidad y Madurez. Planteamos entonces la utilización de modelos de madurez y capacidad en usabilidad para conformar nuevos procesos de trabajo y modificar los existentes. En particular, describimos el Modelo de Madurez en Usabilidad contenido en las Normas ISO, que identifica siete procesos de Diseño Centrado en Usuario y una escala de Capacidad que comprende seis niveles (desde 0 o Incompleto hasta 5 u Optimizado).

Tomamos el Proceso Unificado como uno de los métodos de desarrollo que ha conseguido mayor adhesión en la comunidad de ingeniería de software. Analizamos su versión open source, conocida como OpenUP para evaluar su conformidad con el Modelo de Madurez en Usabilidad de ISO. Identificamos algunas características que están en la base del Proceso Unificado y que resultan indispensables para procesos Centrados en Usuario, pero también encontramos varias falencias que no pueden ser cubiertas simplemente con la modificación

ad hoc de algunas de sus prácticas.

En consecuencia, la propuesta de esta Tesis es extender OpenUP para que pueda ser instanciado como un proceso centrado en usuario. Se proponen dos contribuciones. Por un lado, se agrega una nueva práctica a la EPL (el framework de buenas prácticas del que OpenUP es instanciación), el Desarrollo Centrado en Usuario o DesCU. Esta Práctica articula los elementos de contenido de método específicos del DCU, sus roles, tareas y productos de trabajo, tomando como referencia el ciclo de vida propuesto por el modelo de ISO 13407 sobre el que se cimenta el MMU-ISO.

La segunda contribución consiste en extender la instanciación de OpenUP con tres configuraciones de métodos que permitirán alcanzar los niveles 1, 2 o 3 de Capacidad en Usabilidad en el marco del MMU-ISO. En estas configuraciones se articula la práctica DesCU con las restantes del framework para permitir instanciaciones del Proceso Unificado que puedan conformar al MMU-ISO.

Finalmente, mostramos una implementación de estas extensiones como plugins para el Eclipse Process Framework Composer. Estos plugins podrán ser utilizados por un equipo de trabajo simplemente publicando sus contenidos tal como son provistos para el nivel de capacidad deseado, pero también podrán ser customizados a las necesidades de un proyecto u organización particular.

## Discusión y lecciones aprendidas

Hemos señalado en varias oportunidades que un proceso centrado en usuarios es la mejor manera identificada hasta el momento para garantizar buenas condiciones de usabilidad del producto. Sin embargo, no todos los procesos de desarrollo de software pueden ser transformados en DCU de manera simple. Es necesario que dispongan al menos de dos características básicas: incrementalidad e iteratividad. Ambas se encuentran en la naturaleza del Proceso Unificado y por tanto fue posible desarrollar una adaptación de OpenUP orientada a lograr mayor centralidad en el usuario.

La extensión que se presenta apunta a las dos carencias más significativas que encontramos en todos los procesos de desarrollo y en las alternativas de incorporación de usabilidad que revisamos. Por un lado, es necesario incorporar nuevos roles habitualmente no tenidos en cuenta en el proceso de desarrollo para transformarlo en un proceso centrado en usuario. Esto implica no sólo comprometer a los stakeholders y usuarios finales, sino agregar una nueva batería de habilidades y destrezas en el proyecto. Esto debe ser incorporado desde el mismo momento de la concepción del proyecto y los diseñadores de experiencia de usuario deben trabajar continuamente acoplados con los analistas, arquitectos y diseñadores de software.

Por otra parte, la manera de convertir a todo el proceso en centrado en usuario es atravesar el ciclo de vida completo con actividades vinculadas a la experiencia del usuario. No hay forma de garantizar buenos niveles de usabilidad en un producto si todos los aspectos de la calidad en el uso se reducen al diseño de la interfaz de usuario.

Poner al DCU al mismo nivel de otras buenas prácticas de la ingeniería de software como el Desarrollo guiado por Casos de Uso, la Planificación en dos niveles, etc. le brinda un espacio necesario en la mesa de las decisiones. La organización de la EPL como un framework de buenas prácticas facilita la organización de todos los elementos de método propios de un proceso centrado en usuarios como un conjunto de alta cohesión y fácilmente acoplable a otras prácticas. Manteniendo el bajo nivel de ceremonial y burocracia que sustenta a la EPL y OpenUP es posible realizar configuraciones centradas en usuarios como hemos podido demostrar. Se agregan ocho productos de trabajo y diez definiciones de tareas que los crean, usan y modifican.

La guía del MMU-ISO como hoja de ruta facilita la incorporación gradual, pero sistemática de mayores niveles de capacidad en usabilidad por parte de un equipo de trabajo

u organización. Las configuraciones de base que se proponen como OpenUP/MMU-ISO Nx pueden ser instanciadas fácilmente con las modificaciones apropiadas a cada organización gracias al EPF Composer.

## Trabajos futuros

Los siguientes trabajos aparecen como continuidad de lo presentado en esta tesis:

- Realizar trabajos de análisis en campo con implementación de procesos que instancien las configuraciones OpenUP/MMU-ISO Nx para evaluar los ajustes que resulten necesarios.
- Evaluar la posibilidad de realizar extensiones de OpenUP que permitan alcanzar los otros niveles de Capacidad y Madurez del MMU-ISO. Los niveles 4 y 5 del MMU-ISO corresponden a aspectos vinculados con la organización, más que con un proyecto o equipo determinado. Sin embargo, debería ser posible determinar qué partes del método deberán ser adaptadas para dar el soporte adecuado a una conformidad con esos niveles de capacidad y madurez.

# Bibliografía

- AMBLER, S. 2006. The Agile Unified Process (AUP). <http://www.ambysoft.com/unifiedprocess/agileUP.html>. Ultimo acceso: marzo 2010.
- AMBLER, S., NALBONE, J., AND VIZDOS, M. 2005. *The Enterprise Unified Process: extending the Rational Unified Process*. Prentice Hall Press, Upper Saddle River, NJ, USA.
- AMBLER, S. W. 2004. *The Object Primer : Agile Model-Driven Development with UML 2.0*. Cambridge University Press.
- ANNET, J. AND DUNCAN, K. 1967. Task analysis and training in design. *Occupational psychology* 41.
- ANSI. 2001. *Common Industry Format for Usability Test Reports*.
- APRIL, A. AND COALLIER, F. 1995. *Trillium: a model for the assessment of telecom software system development and maintenance capability*. Software Engineering Standards, International Symposium on 0, 175.
- ARNOWITZ, J., ARENT, M., AND BERGER, N. 2006. *Effective Prototyping for Software Makers (The Morgan Kaufmann Series in Interactive Technologies)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.
- ASTELS, D. 2003. *Test Driven development: A Practical Guide*. Prentice Hall Professional Technical Reference.
- ATG, I. 2010. Web Enabled UCD v1.0. [http://www.iconatg.com/iconprocess/plugins/web\\_enabled\\_ucd](http://www.iconatg.com/iconprocess/plugins/web_enabled_ucd). Ultimo acceso: marzo 2010.
- BASS, L. AND JOHN, B. E. 2001a. Achieving usability through software architecture. In *ICSE '01: Proceedings of the 23rd International Conference on Software Engineering*. IEEE Computer Society, Washington, DC, USA, 684.
- BASS, L. AND JOHN, B. E. 2001b. Supporting Usability Through Software Architecture. *Computer@* 10, 113–115.
- BECK. 2002. *Test Driven Development: By Example*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.
- BEYER, H. AND HOLTZBLATT, K. 1998. *Contextual design: defining customer centered systems*. Morgan Kaufmann, San Francisco USA.
- BUXTON, B. 2007. *Sketching User Experiences: Getting the Design Right and the Right Design*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.
- CARD, S. K., NEWELL, A., AND MORAN, T. P. 2000. *The Psychology of Human-Computer Interaction*. L. Erlbaum Associates Inc., Hillsdale, NJ, USA.

- CARROLL, J. M., Ed. 1995. *Scenario-based design: envisioning work and technology in system development*. John Wiley & Sons, Inc., New York, NY, USA.
- COCKBURN, A. 2000. *Writing effective use cases*. Addison Wesley.
- COCKBURN, A. 2002. *Agile software development*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.
- CONALLEN, J. 2002. *Building Web Applications with UML*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.
- CONSTANTINE, L. AND LOCKWOOD, L. 1999. *Software for use: a practical guide to the models and methods of usage-centered design*. Addison Wesley, Reading, MA.
- CONSTANTINE, L. L. AND LOCKWOOD, L. A. D. 2002. Usage-Centered Engineering for Web Applications. *IEEE Softw.* 2, 42–50.
- COOPER, A. 2004. *The Inmates Are Running the Asylum: Why High Tech Products Drive Us Crazy and How to Restore the Sanity (2nd Edition)*. Pearson Higher Education.
- COSTABILE, M. F. 2001. *Usability in the software lifecycle*. Vol. 1. World Scientific Publishing Company, 179–192.
- CROSBY, P. B. 1979. *Quality Is Free. The Art of Making Quality Certain*. McGraw-Hill, New York.
- DSDM-CONSORTIUM. 2007. OpenUP/DSDM Plugin. website. [http://www.eclipse.org/epf/downloads/openup/openup\\_downloads.php](http://www.eclipse.org/epf/downloads/openup/openup_downloads.php). Ultimo acceso: marzo 2010.
- DSDM-CONSORTIUM. 2010. DSDM Public Version 4.2. <http://dev.dsdm.org/version4/2/public/default.asp>. Ultimo acceso: marzo 2010.
- EARTHY, J. 1998. Usability Maturity Model: human-centeredness scale. Tech. rep., Lloyds Register of Shipping.
- EARTHY, J. 2000. Usability Maturity Model: Processes. Tech. rep., Lloyd's Register. January.
- ECLIPSE. 2006. Eclipse Process Framework Project. [http://www.eclipse.org/projects/project\\_summary.php?projectid=technology.epf](http://www.eclipse.org/projects/project_summary.php?projectid=technology.epf). Ultimo acceso: marzo 2010.
- ECLIPSE. 2008a. Eclipse Process Framework Practice Library. <http://epf.eclipse.org/wikis/epfpraclib/>. Ultimo acceso: marzo 2010.
- ECLIPSE. 2008b. Method Authoring Method for the EPF Practices Library. <http://epf.eclipse.org/wikis/mam/index.htm>. Ultimo acceso: marzo 2010.
- ECLIPSE. 2008c. OpenUP version 1.5.0.1.
- ECLIPSE. 2008d. Roadmap: Authoring in the Eclipse Practice Library (EPL). Ultimo acceso: marzo 2010.
- EHRlich, K. AND ROHN, J. A. 1994. Cost justification of usability engineering: a vendors' perspective. *Cost-justifying usability*, 73–110.
- FLANAGAN, G. A. AND RAUCH, T. L. 1995. Usability management maturity, part 1 (abstract): self assessment—how do you stack up? In *CHI '95: Conference companion on Human factors in computing systems*. ACM, New York, NY, USA, 336.

- FOLMER, E. AND BOSCH, J. 2005. Case studies on Analyzing Software Architectures for Usability. In *EUROMICRO '05: Proceedings of the 31st EUROMICRO Conference on Software Engineering and Advanced Applications*. IEEE Computer Society, Washington, DC, USA, 206–213.
- GARRETT, J. J. 2002. *The Elements of User Experience: User-Centered Design for the Web*. New Riders Publishing, Thousand Oaks, CA, USA.
- GORANSSON, B., LIF, M., AND GULLIKSEN, J. 2003. Usability design - Extending Rational Unified Process with a new discipline. In *Interactive systems: design, specification and verification. 10th International Workshop*, J. Nunes and J. Cunha, Eds. Springer Verlag, Funchal, Madeira Island, Portugal.
- GOULD, J. AND LEWIS, C. 1985. Designing for usability: key principles and what designers think. *Communications of ACM* 3, 300–311.
- GULLIKSEN, J., G, B., BOIVIE, I., BLOMKVIST, S., PERSSON, J., AND CAJANDER, A. 2003. Key principles for user-centred systems design. *Behaviour and Information Technology* 6, 397–409.
- HACKOS, J. T. AND REDISH, J. C. 1998. *User and task analysis for interface design*. John Wiley & Sons, Inc., New York, NY, USA.
- HIX, D. AND HARTSON, H. R. 1993. *Developing user interfaces: ensuring usability through product & process*. John Wiley & Sons, Inc., New York, NY, USA.
- HOLTZBLATT, K., WENDELL, J. B., AND WOOD, S. 2005. *Rapid Contextual Design. A how to guide to key techniques for user centered design*. Morgan Kaufmann, San Francisco, USA.
- HUMPHREY, W. S. 1989. *Managing the software process*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.
- IBM. 2005a. Rational Unified Process for Large Projects 7.0.1.E.
- IBM. 2005b. RUP for user experience modeling. [http://www.ibm.com/developerworks/rational/library/05/1206\\_rup\\_plugins/rup\\_ue.html](http://www.ibm.com/developerworks/rational/library/05/1206_rup_plugins/rup_ue.html). Ultimo acceso: marzo 2010.
- ISO/IEC. 1995. 12207 Information Technology Software Life Cycle Processes. Tech. rep., International Organization for Standardization and International Electrotechnical Commission.
- ISO/IEC. 1998. 9241-11 Ergonomic requirements for office work with visual display terminals (VDT)s - Part 11 Guidance on usability.
- ISO/IEC. 1999. 13407 Human-centred design processes for interactive systems.
- ISO/IEC. 2000a. 18529 Ergonomics of human system interaction. Human-centred lifecycle process descriptions. Tech. rep.
- ISO/IEC. 2000b. 9126 Software product quality - Quality model.
- ISO/IEC. 2006. 15504 Information Technology Software Process Assessment Part 1: Concepts and Vocabulary, Part 2: Performing an Assessment, Part 3: Guidance on Performing an Assessment, Part 4: Guidance on Use for Process Improvement and Process Capability Determination, Part 5: An Exemplar Process Assessment Model, 2003-2006. Tech. rep., International Organization for Standardization and International Electrotechnical Commission.

- JACOBSON, I., BOOCH, G., AND RUMBAUGH, J. 1999. *The unified software development process*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.
- JACOBSON, I., CHRISTERSON, M., JONSSON, P., AND VERGAARD, G. 1992. *Object-oriented software engineering: a use case driven approach*. Addison Wesley, Reading MA.
- JOHNSON, J. AND HENDERSON, A. 2002. Conceptual models: begin by designing what to design. *interactions@* 1, 25–32.
- JOKELA, T. 2001a. Assessment of user centred design processes as a basis for improvement action. Ph.D. thesis, University of Oulu.
- JOKELA, T. 2001b. The KESSU Usability Design Process Model. Version 2.1. Tech. rep., University of Oulu, Finland.
- KENT BECK, MIKE BEEDLE, A. V. B. A. C. W. C. M. F. 2001. Manifesto for Agile Software Development. <http://agilemanifesto.org/>. Ultimo acceso: marzo 2010.
- KROLL, P. AND MACISAAC, B. 2006. *Agility and Discipline Made Easy: Practices from OpenUP and RUP (Addison-Wesley Object Technology (Paperback))*. Addison-Wesley Professional.
- KRUTCHEN, P. 2000. *The Rational Unified Process: an introduction*, 2nd ed. Addison Wesley.
- KUNIAVSKY, M. 2003. *Observing the User Experience: A Practitioner's Guide to User Research (Morgan Kaufmann Series in Interactive Technologies) (The Morgan Kaufmann Series in Interactive Technologies)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.
- KUVAJA, P. 1995. BOOTSTRAP: A Software Process Assessment and Improvement Methodology. In *Proceedings of the Second Symposium on Software Quality Techniques and Acquisition Criteria on Software Quality Techniques and Acquisition Criteria*. Springer-Verlag, London, UK, 31–48.
- LANDAUER, T. K. 1996. *Trouble with Computers: Usefulness, Usability, and Productivity*. MIT Press, Cambridge, MA, USA.
- LARMAN, C. 2003. *Agile and Iterative Development: A Manager's Guide*. Pearson Education.
- LIMBOURG, Q. AND VANDERDONCKT, J. 2004. Comparing task models for user interface design. In *The Handbook of Task Analysis for Human Computer Interaction*, D. Diaper and N. Stanton, Eds. Lawrence Erlbaum Associates, 135–154.
- LORÉS, J., Ed. 2001. *La interacción persona ordenador*. AIPO, Asociación Interacción Persona Ordenador, Espana.
- LUNDMARK, M. AND TORESSON, J. 2007. Assessing usability. Effective strategies for enhancing usability in projects. Ph.D. thesis, University of Chalmers.
- MAYHEW, D. J. 1999. *The usability engineering lifecycle: a practitioner's handbook for user interface design*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.
- NIELSEN, J. 1993. *Usability engineering*. Academic Press, San Diego.
- NIELSEN, J. 2006a. Corporate usability maturity: stages 1-4. <http://www.useit.com/alertbox/maturity.html>. Ultimo acceso: marzo 2010.

- NIELSEN, J. 2006b. Corporate usability maturity: stages 5-8. [http://www.useit.com/alertbox/process\\_maturity.html](http://www.useit.com/alertbox/process_maturity.html). Ultimo acceso: marzo 2010.
- NUNES, N. J. AND CUNHA, J. F. 2001. *WISDOM-Whitewater interactive system development with object models*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 197-243.
- OMG. 2008. Software & Systems Process Engineering Meta-Model Specification Version 2.0. Tech. rep., OMG.
- PATERNÓ, F. 1999. *Model-Based Design and Evaluation of Interactive Applications*. Springer-Verlag, London, UK.
- PATERNÒ, F., MANCINI, C., AND MENICONI, S. 1997. ConcurTaskTrees: A Diagrammatic Notation for Specifying Task Models. In *INTERACT '97: Proceedings of the IFIP TC13 Interantional Conference on Human-Computer Interaction*. Chapman & Hall, Ltd., London, UK, UK, 362-369.
- PAULK, M. C. 1995. *The capability maturity model : guidelines for improving the software process*. Addison-Wesley Pub. Co, Reading, Mass.
- PRUITT, J. AND ADLIN, T. 2006. *The Persona Lifecycle: Keeping People in Mind Throughout Product Design (Interactive Technologies)*, 1 ed. Morgan Kaufmann.
- RAUCH, T. L. AND FLANAGAN, G. A. 1995. Usability management maturity, part 2 (abstract): usability techniques—what can you do? In *CHI '95: Conference companion on Human factors in computing systems*. ACM, New York, NY, USA, 338.
- RIEMAN, J., FRANZKE, M., AND REDMILES, D. 1995. Usability evaluation with the cognitive walkthrough. In *CHI '95: Conference companion on Human factors in computing systems*. ACM, New York, NY, USA, 387-388.
- ROSSON, M. B. AND CARROLL, J. M. 2002. *Usability engineering: scenario-based development of human-computer interaction*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.
- SCHAFFER, E. 2004. *Institutionalization of Usability: A Step-by-Step Guide*. Addison Wesley Longman Publishing Co., Inc., Redwood City, CA, USA.
- SCHWABER, K. 2004. *Agile Project Management With Scrum*. Microsoft Press, Redmond, WA, USA.
- SEFFAH, A., GULLIKSEN, J., AND DESMARAIS, M. C. 2005. *Human-Centered Software Engineering - Integrating Usability in the Development Process (Human-Computer Interaction Series)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA.
- SEFFAH, A. AND METZKER, E. 2004. The obstacles and myths of usability and software engineering. *Commun. ACM* 12, 71-76.
- SEFFAH, A., VANDERDONCKT, J., AND DESMARAIS, M. C. 2009. *Human-Centered Software Engineering: Software Engineering Models, Patterns and Architectures for HCI*. Springer Publishing Company, Incorporated.
- SEI. 1995. The Capability Maturity Model: Guidelines for Improving the Software Process. Tech. rep., Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA.
- SHARP, H., ROGERS, Y., AND PREECE, J. 2007. *Interaction Design: Beyond Human Computer Interaction*. John Wiley & Sons.

- SHEPHERD, A. 2001. *Hierarchical task analysis*. CRC Press.
- SHERWOOD-JONES, B. 1995. Total Systems Maturity. Internal report, version 2. Tech. rep., BAeSEMA, Glasgow.
- SHNEIDERMAN, B., PLAISANT, C., COHEN, M., AND JACOBS, S. 2009. *Designing the User Interface: Strategies for Effective Human-Computer Interaction*. Addison-Wesley Publishing Company, USA.
- STANTON, D. D. N., Ed. 2004. *The handbook of task analysis for human-computer interaction*. Lawrence Erlbaum Associates.
- STANTON, N. A. 2006. Hierarchical Task Analysis: Developments, Applications and Extensions. *Applied Ergonomics* 37, 55-79.
- WINOGRAD, T. 1997. *The design of interaction*. In *Beyond calculation: the next fifty years*, P. D. R. Metcalfe, Ed. Springer Publishing Company, Incorporated, New York, NY, USA, 149-161.

Parte IV  
Apéndices

# Apéndice A

## MMU-ISO: Procesos DCU

A continuación se transcribe cada uno de los procesos DCU contenidos en el MMU-ISO en términos de su objetivo o propósito, los indicadores que permiten medir el éxito alcanzado y las prácticas de base a llevar a cabo.

### DCU 1. Asegurar el contenido DCU en la estrategia de sistemas

#### Propósito

Establecer y mantener un foco en temas relacionados con los involucrados y los usuarios en cada parte de la organización que aporte al concepto, desarrollo, mantenimiento y soporte del sistema

#### Indicadores de éxito

- El marketing tiene en cuenta temas de usabilidad, ergonomía y socio-técnicos.
- Los sistemas se dirigen a satisfacer necesidades y expectativas de usuarios.
- Los planificadores tienen en cuenta los requerimientos de involucrados y organizaciones durante el planteo de la estrategia de sistema.
- Los sistemas dan mejor respuesta a los cambios provenientes de los usuarios (tareas, expectativas, contexto, necesidades, etc.)
- La empresa responde mejor a los cambios de usuarios.
- El sistema tiene menos probabilidades de ser rechazado por el mercado.

#### Actividades

##### Representar a los involucrados

Actuar como el abogado de los usuarios y todos los interesados en el desarrollo del sistema. El abogado del usuario recuerda al staff que el sistema será usado por gente real y tiene que alcanzar determinados niveles de calidad en el uso. Este rol incluye aproximaciones de logro en diseño centrado en usuario, lograr la participación de usuarios en estudios conceptuales, investigar y divulgar temas de contexto de uso.

**Analizar el mercado**

Realizar investigaciones de anticipación en grupos de usuarios potenciales para identificar necesidades futuras de sistemas, nuevos grupos de usuarios u organizaciones. Identificar el contexto de uso esperado para los sistemas futuros. Plantear procedimientos para extraer información de usuarios sobre los sistemas futuros.

**Definir y planificar la estrategia de sistemas**

Presentar la información del mercado como una visión (por ejemplo para la aprobación de la dirección de la empresa o el proyecto). Operacionalizar la visión en una estrategia de implementación. Establecer el costo de sistemas en términos de su ciclo de vida, para poder valorar el costo de un abordaje centrado en el usuario.

**Reunir feedback del mercado**

Realizar investigaciones para refinar y consolidar la estrategia de sistemas basada en feedback de usuarios y no usuarios en el mercado al que se dirigirá el sistema a desarrollar.

**Analizar las tendencias de los usuarios**

Buscar cambios en usuarios (sus habilidades y entrenamiento, sus necesidades y deseos), las tareas (cambios en el tipo o volumen de trabajo), contexto (cambios en los entornos de vida o de trabajo, nuevas tecnologías, expectativas sociales o políticas). El análisis de esta información permitirá predecir necesidades futuras.

**DCU 2. Planificar y gestionar el proceso DCU****Propósito**

Especificar la forma en que las actividades centradas en la persona se adaptan al ciclo de vida completo de procesos y la empresa

**Indicadores de éxito**

- El plan de proyecto permite iteraciones e incorporaciones de feedback de usuario.
- Se asignan recursos para la efectiva comunicación entre los participantes del equipo de diseño.
- Se concilian potenciales conflictos y compromisos entre temas de HCI y otros aspectos del desarrollo.
- Procesos centrados en la persona se incorporan en los sistemas, procedimientos y estándares de calidad.
- Los temas de HCI y DCU son soportados y promovidos en la organización

**Actividades****Consultar a los involucrados**

Establecer las estructuras, mecanismos y procedimientos para asegurar que todos los involucrados relevantes sean consultados de manera efectiva en cada aspecto significativo para el desarrollo e implementación del sistema

**Identificar y planificar la participación de los usuarios**

Decidir los modos más efectivos para extraer el input del usuario en cada etapa del proyecto, tomando las mayores ventajas de las buenas prácticas sobre trabajo de equipo y la participación de usuario.

**Seleccionar los métodos y técnicas centrados en humanos**

Decidir qué métodos serán incluidos y cómo se relacionarán entre ellos en el proceso de desarrollo. Definir cómo será la interfaz de estos métodos con la metodología particular del ciclo de vida que se usa para el desarrollo de sistema.

**Asegurar la aproximación centrada en la persona**

Establecer una cultura multidisciplinar del equipo de proyecto. Mantener el foco del staff en una aproximación centrada en usuarios. Identificar las habilidades de requeridas para los especialistas y planificar cómo proveerlas.

**Planificar actividades de DCU y conducirlas**

Desarrollar un plan especificando cómo las actividades centradas en el usuario se integrarán en el proceso general de desarrollo. Realizar un abordaje específico de los temas de usuario en la gestión de proyectos y departamentos de desarrollo. Asegura que el proceso de desarrollo de sistemas tome en cuenta el input del usuario. Tomar en cuenta los temas de usuario e involucrados en las actividades de soporte (por ejemplo, en la gestión de contratos y compras).

**Liderar una aproximación centrada en usuario**

Promover un abordaje centrado en el usuario dentro de la empresa o el equipo de proyecto. Establecer y comunicar una política sobre el tema.

**Soportar DCU**

Incluir elementos centrados en el usuario en los procedimientos de soporte (aseguramiento de calidad, control de cambios, procesos y métodos de mantenimiento, gestión de recursos). Asegurar que estos se realizan como parte integral de la gestión de infraestructura de la empresa o del equipo.

**DCU 3. Especificar los requerimientos de los involucrados y de la organización****Propósito**

Establecer los requerimientos del sistema desde la organización y otros interesados, tomando en cuenta las necesidades, competencias y entorno de trabajo de cada involucrado relevante al sistema.

**Indicadores de éxito**

- Se define la performance requerida del sistema contra objetivos operativos y funcionales.
- Se identifican requerimientos normativos y legislativos.

- Se define la cooperación y comunicación entre usuarios y otras partes relevantes.
- Se define el trabajo de los usuarios (alocación de tareas, confort, seguridad, salud, motivación)
- Se establece la performance de tareas de los usuarios con el sistema
- Se define el diseño de trabajo, las prácticas de la organización y la estructura.
- Se identifican la factibilidad de operaciones y mantenimiento.
- Se fijan objetivos para la operación y/o uso de los componentes de software y hardware

## Actividades

### Clarificar y documentar las metas del sistema

Describir los objetivos que los usuarios y la organización desean alcanzar con el uso del sistema.

### Definir a los involucrados

Identificar y analizar los roles de cada grupo de usuarios e involucrados que serán afectados por el sistema. Valorar la significación y relevancia del sistema para cada grupo de involucrados.

### Evaluar los riesgos de las personas y el sistema

Revisar los riesgos para la seguridad, salud y bienestar de los involucrados en el sistema. Relacionarlos con la gestión de riesgo del sistema.

### Definir el sistema

Establecer y acordar las funciones y performance requerida para el sistema en términos de la experiencia total para los involucrados con el sistema (metas para adecuación, aceptabilidad y eficiencia del usuario final). La experiencia total cubre cada aspecto de la relación con el sistema por parte de los involucrados relevantes, desde el encargo del trabajo hasta la finalización completa.

### Generar los requerimientos de los involucrados y la organización

Desarrollar un planteo explícito de los requerimientos de los involucrados y la organización sobre el sistema. Este proceso es iterativo. Los requerimientos pueden ser organizados en orden de importancia. Los requerimientos normativos sobre el entorno y la carga de trabajo deben ser tenidos en cuenta. Los requerimientos de los involucrados y la organización definen una gran parte de los requerimientos operativos y de performance del sistema.

### Establecer los objetivos de calidad en el uso

Generar y acordar criterios mensurables para la calidad de uso requerida al sistema. La calidad de uso se establece como los niveles solicitados de efectividad, productividad y satisfacción del usuario hacia el sistema o sus componentes, en un contexto particular de tareas y sobre la base de requerimientos de performance.

## **DCU 4. Entender y especificar el contexto de uso**

### **Propósito**

Identificar, clarificar y registrar las características de los involucrados, sus tareas y el entorno físico y organizacional en el que operará el sistema.

### **Indicadores de éxito**

- Se han definido las características de los usuarios
- Se han definido las tareas que los usuarios deberán realizar
- Se han definido la organización y el entorno en el que se utilizará el sistema.

### **Actividades**

#### **Identificar y documentar las tareas del usuario**

Describir las actividades que los usuarios realizan para alcanzar las metas del sistema. Las descripciones de tareas no incluyen solamente las funciones o características del equipamiento. Las tareas pueden cambiar o evolucionar durante el ciclo de vida del sistema.

#### **Identificar y documentar los atributos del usuario**

Describir las características relevantes de los usuarios finales del sistema, incluyendo conocimientos, lenguaje, capacidades físicas, nivel de experiencia, etc.

#### **Identificar y documentar el entorno organizacional**

Describir el marco social y organizacional, la estructura y prácticas de gestión, etc.

#### **Identificar y documentar el entorno técnico**

Describir las características relevantes de cualquier equipo que sea utilizado. Para sistemas nuevos las características del equipo dependen de las soluciones a proponer, por lo que es posible que no sean conocidas hasta bastante avanzado el proyecto.

#### **Identificar y documentar el entorno físico**

Describir la ubicación, equipamiento del puesto de trabajo y condiciones ambientales.

## **DCU 5. Producir soluciones de diseño**

### **Propósito**

Crear soluciones potenciales de diseño mediante el uso de prácticas del estado del arte, la experiencia y conocimiento de los participantes y los resultados del análisis del contexto de uso.

## Indicadores de éxito

- Se considera en el diseño el sistema socio-técnico completo en que se inserta el sistema
- Se tienen en cuenta las características y necesidades del usuario en la compra de componentes del sistema y en el diseño
- Se integran en el sistema los conocimientos existentes y buenas prácticas de ingeniería de sistemas socio-técnicos, ergonomía, psicología, ciencias cognitivas, etc.
- Se mejora la comunicación entre los involucrados en el sistema haciendo explícitas las decisiones de diseño.
- El equipo de desarrollo es capaz de explorar varios conceptos de diseño antes de confirmar alguno.
- Se incorpora feedback de todos los involucrados y usuarios desde las primeras fases del proceso de desarrollo
- Es posible evaluar varias iteraciones de un diseño y sus alternativas.
- Se diseña la interfaz de usuario
- Se desarrolla el entrenamiento y soporte para el usuario

## Actividades

### Alocar funciones

Analizar el contexto de uso y las funciones y performance requerida del sistema para distribuir las funciones entre las personas, las máquinas y los componentes organizacionales del sistema que mejor puedan cumplimentar cada una de ellas. La asignación de funciones puede ser dinámica. El objetivo es optimizar la performance del sistema completo para el logro de las metas. En los niveles superiores de la jerarquía del sistema puede ocurrir que las funciones no se puedan asignar a algún componente en particular (persona, equipo, software, organización, etc.), sino a subsistemas constituidos por más de uno de ellos.

### Producir el modelo de tareas compuesto

Desarrollar un modelo factible de las nuevas tareas del usuario a partir del conocimiento de las mejores prácticas, los requerimientos, el contexto de uso, la asignación de funciones y las restricciones de diseño del sistema. Los procesos DCU 5.1 a 5.3 se llevan a cabo en cada nivel de la jerarquía de sistemas., mientras que DCU5.4 a 5.8 se realizan donde se definen y desarrollan componentes de sistema.

### Explorar el diseño del sistema

Generar y analizar un rango de opciones de diseño para cada aspecto del sistema relacionado con su utilización y su efecto en los involucrados.

### Desarrollar soluciones de diseño

Aplicar la información relevante del conocimiento sobre la gente al diseño de sistemas. Incluir en el diseño los requerimientos de involucrados y organización, contexto de uso, estándares internacionales, leyes, patentes, buenas prácticas, guías de estilo, etc.

**Especificar el sistema**

Producir un diseño para cada componente relacionado con el uso. Cambiar el diseño a la luz del feedback surgido de las evaluaciones. Dependiendo del tipo de sistema, la especificación puede incluir: diseñar el trabajo del usuario, sus tareas, el entorno de trabajo, equipamiento, software, documentación, diseño de envases, funcionalidad de interfaces, etc.

**Desarrollar prototipos**

Concretar las soluciones de diseño mediante simulaciones, modelos, maquetas, etc. Desarrollar simulaciones o ensayos de aspectos clave del sistema a los efectos de testearlas con usuarios o sus representantes.

**Desarrollar el entrenamiento para el usuario**

Identificar, especificar y producir el entrenamiento requerido para permitir a los involucrados realizar las tareas en forma efectiva con el sistema nuevo. Cubrir o incluir cualquier cambio propuesto en los procesos de negocio, diseño de trabajo o tareas.

**Desarrollar el soporte para el usuario**

Identificar, especificar y producir los servicios de soporte para el usuario del sistema. Tomar en cuenta los cambios propuestos en los procesos de negocio o diseño de trabajo.

**DCU 6. Evaluar los diseños contra los requerimientos****Propósito**

Reunir feedback sobre el diseño en desarrollo proveniente de los usuarios y otras fuentes representativas.

**Indicadores de éxito**

- Se ha provisto feedback para mejorar el diseño.
- Existe una forma de medir si los objetivos de la organización y los involucrados se alcanzaron o no.
- Existe un monitoreo continuo sobre el uso en el largo plazo

**Actividades****Especificar y validar el contexto de evaluación**

Describir y verificar las condiciones bajo las cuales el sistema será testeado o evaluado de cualquier forma. Describir las relaciones, especialmente las discrepancias, entre el contexto de evaluación y el contexto de uso. Esto debería realizarse antes de cada uno de DCU 6.2 a 6.6

**Evaluar prototipos iniciales para definir los requerimientos del sistema**

Calificar sistemas apropiados usando criterios relevantes. Testear la usabilidad de sistemas competitivos o alternativos. Utilizar prototipos para estimular el input de involucrados a los sistemas de usuarios. Testear la estabilidad de los requerimientos.

**Evaluar prototipos para mejorar el diseño**

Recolectar input de usuario sobre la calidad de uso del sistema en desarrollo. Presentar los resultados al equipo de diseño de la manera más apropiada.

**Evaluar el sistema para verificar que los requerimientos se cumplen**

Testear las versiones en desarrollo o final del sistema para asegurar que cumple los requerimientos de los usuarios, las tareas y el entorno como fueron definidos.

**Evaluar el sistema para verificar que se siguieron las prácticas requeridas**

Chequear los sistemas sobre la adherencia al conocimiento aplicable sobre las ciencias humanas, guías de estilo, estándares, legislación.

**Evaluar el sistema para asegurar que continuamente cumple las metas organizacionales y las necesidades de los usuarios**

Chequear el sistema en uso contra los cambios en las necesidades organizacionales, del usuario u otros involucrados y de usabilidad para asegurar que continúa cumpliendo las mismas. Esto incluye el contacto rutinario con un número representativo de usuarios utilizando algún procedimiento definido para extraer la información necesaria. Incluye también el feedback a todos los involucrados. La evaluación del sistema en uso puede ser empleada también para evaluar si los requerimientos y la especificación resultante fueron correctos.

**DCU 7. Presentar y operar el sistema****Propósito**

Establecer los aspectos centrados en la persona para el soporte e implementación del sistema.

**Indicadores de éxito**

- Se comunican al proyecto las necesidades de los interesados en el sistema.
- Se especifican los procesos de gestión de cambios, incluyendo las responsabilidades de usuarios y desarrolladores.
- Se manejan los requerimientos de soporte para usuarios, encargados de mantenimiento y otros interesados.
- Existe cumplimiento de normas y procedimientos de seguridad e higiene.
- Se soportan adaptaciones locales del sistema
- Se reúnen las reacciones de los usuarios y los cambios resultantes en el sistema son reportados a los interesados.

**Actividades****Gestionar el cambio**

Facilitar, supervisar y asegurar que los aspectos de DCU en la implementación del sistema. Esto incluye reorganizar el diseño de trabajo y las prácticas, equipos, entrenamiento, etc.

**Determinar el impacto sobre la organización e involucrados**

Evaluar el impacto del sistema a introducir sobre la gente y la organización.

**Personalización y diseño local**

Proveer soporte para la customización del sistema que permita cubrir las necesidades culturales u operacionales en algún lugar. Proveer soporte para la customización y configuración del sistema para satisfacer necesidades de usuarios específicos. Proveer detalles de customización a la gestión de configuración.

**Proveer entrenamiento al usuario**

Brindar entrenamiento y talleres a los usuarios para cubrir las necesidades identificadas y facilitar la transmisión al nuevo diseño del trabajo o las nuevas disposiciones de equipos.

**Soportar a los usuarios en las actividades planificadas**

Mantener contacto con los usuarios y la organización de clientes durante la definición, desarrollo e introducción de sistema.

**Asegurar la conformidad con la legislación de ergonomía de puestos de trabajo**

Informe de lugares de trabajo, usuarios y programas de entrenamiento para asegurar que el software, hardware y puesto de trabajo cubre los requerimientos de legislación nacional.

## Apéndice B

# MMU-ISO: Niveles de Capacidad y Madurez

En este apéndice detallamos todos los atributos a identificar en un assessment para cada uno de los seis niveles de capacidad que presenta el MMU-ISO.

### Nivel 0. Proceso incompleto

El proceso no está implementado o falla al tratar de alcanzar su propósito. Sin atributos a medir en este nivel.

### Nivel 1: Proceso realizado

El proceso implementado alcanza el propósito definido. Los siguientes atributos del proceso demuestran el logro de este nivel:

#### Atributos de la performance del proceso (AP1.1)

El grado en el cual los productos de trabajo resultantes son producidos a partir de los productos de input mediante la realización de las prácticas que comprende este proceso. Las prácticas de gestión para lograr este atributo del proceso es:

- **MP1.1.1** Asegurar que las prácticas básicas son realizadas para satisfacer el propósito del proceso.

### Nivel 2: El proceso gestionado

El proceso gestionado entrega productos de trabajo de calidad aceptable dentro de los tiempos definidos y las necesidades de recursos. Los siguientes atributos del proceso demuestran el logro de este nivel:

#### Atributos de la gestión de la performance(PA2.1)

El grado con el cual el proceso se gestiona para producir productos de trabajo dentro del tiempo establecido y los requerimientos de recursos. Las prácticas de gestión relacionadas son:

- MP2.1.1 Identificar requerimientos de recursos para permitir la planificación y el control del proceso
- MP2.1.2 Planificar la performance del proceso identificando las actividades y los recursos alocados de acuerdo con los requerimientos
- MP2.1.3 Implementar las actividades definidas para lograr el propósito del proceso
- MP2.1.4 gestionar la ejecución de las actividades para producir los productos dentro del tiempo establecido y los requerimientos de recursos.

### **Atributos de la gestión de productos de trabajo(PA2.2)**

El grado con el cual los productos de trabajo son documentados y controlados para satisfacer sus requerimientos funcionales y no funcionales. Las prácticas de gestión relacionadas son:

- MP2.2.1 Identificar requerimientos para la integridad y calidad de los productos de trabajo
- MP2.2.2 Identificar las actividades necesarias para alcanzar los requerimientos de integridad y calidad de los productos de trabajo.
- MP2.2.3 Gestionar la configuración de los productos de trabajo para asegurar la integridad.
- MP2.2.4 Gestionar la calidad de los productos de trabajo para asegurar que satisfacen sus requerimientos funcionales y no funcionales.

## **Nivel 3: El proceso establecido**

El proceso establecido asegura el despliegue de un proceso definido basado en buenos principios de ingeniería de sistemas. Los siguientes atributos del proceso demuestran el logro de este nivel:

### **Atributos de definición de proceso (PA3.1)**

El grado con el cual el proceso contribuye a las metas definidas para el negocio de la organización a través de la definición de un proceso estándar. Las prácticas de gestión relacionadas son:

- MP3.1.1 Identificar entre las disponibles en la organización la definición de proceso estándar que es apropiada al propósito del proceso y las metas de negocio de la organización
- MP3.1.2 Adaptar el proceso estándar para obtener un proceso definido apropiado al contexto
- MP3.1.3 Implementar el proceso definido para lograr su propósito consistente y repetidamente y soportar la meta de negocio definida para la organización
- MP3.1.4 Proveer feedback en el proceso estándar desde la experiencia de usar el proceso definido

### **Atributos de recursos de proceso (PA3.2)**

El grado con el cual el proceso contribuye efectivamente a las metas definidas del negocio de la organización mediante el uso de recursos humanos calificados e infraestructura del proceso. Las prácticas de gestión relacionadas son:

- MP3.2.1 Definir las competencias de recursos humanos requeridas para soportar la implementación del proceso definido
- MP3.2.2 Definir los requerimientos de infraestructura del proceso para soportar la implementación del proceso definido
- MP3.2.3 Proveer recursos humanos calificados adecuadamente para satisfacer las competencias definidas
- MP3.2.4 Proveer infraestructura adecuada para el proceso de acuerdo con las necesidades definidas.

### **Nivel 4: El proceso predecible**

El proceso predecible se realiza en forma consistente dentro de límites de control para alcanzar sus metas.

Los siguientes atributos del proceso demuestran el logro de este nivel:

#### **Atributos de la medición del proceso (PA4.1)**

El grado al cual las metas y métricas se usan para asegurar que la implementación del proceso contribuye al logro de los objetivos.

Las prácticas de gestión relacionadas son:

- MP4.1.1 Definir las metas de proceso y métricas asociadas que soporten los objetivos del negocio de la organización.
- MP4.1.2 Proveer recursos e infraestructura adecuados para la recolección de datos.
- MP4.1.3 Recolectar los datos de medición especificados en la implementación del proceso definido.
- MP4.1.4 Evaluar el logro de metas de proceso por comparación con las medidas registradas.

#### **Atributos del control del proceso (PA4.2)**

El grado en el cual se alcanza el logro confiable de las metas del proceso definido mediante la recolección y análisis de las medidas para controlar y corregir la performance del proceso.

Las prácticas de gestión relacionadas son:

- MP4.2.1 Identificar técnicas de análisis y control apropiadas al contexto del proceso.
- MP4.2.2 Proveer recursos e infraestructura adecuados para el análisis y control del proceso.
- MP4.2.3 Analizar las métricas disponibles para identificar parámetros de control del proceso.
- MP4.2.4 Identificar desviaciones y tomar las acciones de control requeridas para mantener el control del proceso.

## Nivel 5: El proceso optimizado

El proceso optimizado adapta su performance para satisfacer necesidades actuales y futuras del negocio y cumplir sus metas de negocio confiablemente.

Los siguientes atributos del proceso demuestran el logro de este nivel:

### Atributos de cambio del proceso (PA5.1)

El grado en el cual las metas de negocio de la organización se alcanzan mediante cambios en la definición, gestión y performance del proceso.

Las prácticas de gestión relacionadas son:

- MP5.1.1 Identificar y aprobar cambios a la definición estandar del proceso sobre la base de una comprensión cuantitativa del mismo.
- MP5.1.2 Proveer recursos adecuados para implementar de manera efectiva los cambios aprobados en el proceso.
- MP5.1.3 Implementar los cambios aprobados al proceso customizado para lograr los resultados esperados.
- PM5.1.4 Validar la efectividad del cambio de proceso sobre la base de la performance real versus la metas del proceso y del negocio.

### Atributos de mejora continua (PA5.2)

El grado al cual la mejora continua en el logro de las metas de negocio definidas para la organización se asegura mediante cambios al proceso.

Las prácticas de gestión relacionadas son:

- MP5.2.1 Identificar oportunidades de mejora de forma sistemática y proactiva para enriquecer continuamente el proceso.
- MP5.2.2 Establecer una estrategia de implementación basada en las oportunidades de mejora de performance del proceso identificadas de acuerdo con las metas del negocio .
- MP5.2.3 Implementar cambios a las areas seleccionadas del proceso adaptado de acuerdo con la estrategia de implementación.
- MP5.2.4 Validar la efectividad del cambio del proceso sobre la base de la performance real del proceso versus las metas del negocio y el retroalimentar al proceso estandar.



## Apéndice C

# Assessment de OpenUP/Basic

### Evidencia encontrada

En este Apéndice se exhiben las planillas que recogen la evidencia de capacidad de OpenUP/Basic en cada uno de los niveles de MMU-ISO y el grado de logro para cada atributo.

DCU.1 Asegurar el contenido DCU en la estrategia de sistemas				
Prácticas y procesos	Prácticas	Contenidos de proceso	Contenidos de Método	Score
<b>AP1.1 Atributo de performance del proceso</b>				
<b>Nivel 1 Proceso realizado</b>				
<b>DCU.1.1 Representar los stakeholders</b>	Equipo integrado, Visión compartida, Planificación en dos niveles, Desarrollo conducido por UC	Iniciar el proyecto, Identificar y refinar requerimientos	Definir visión, Planificar proyecto, Encontrar y bocetar requerimientos	0.3
<b>DCU.1.2 Analizar el mercado</b>				
<b>DCU.1.3 Definir y planificar la estrategia de sistemas</b>				
<b>DCU.1.4 Reunir respuesta de mercado</b>				
<b>DCU.1.5 Analizar tendencias de usuarios</b>				
Score combinado para los atributos (DCU.1.1 to 1.5):				0.3
Score combinado para este nivel				0.3
<b>Nivel 2 Proceso Cuestioneado</b>				
<b>AP2.1 Atributo de gestión de la performance</b>				
<b>PG2.1.1 Identificar requerimientos de recursos</b>	Equipo integrado, Visión compartida, Planificación en dos niveles, Desarrollo conducido por UC	Iniciar el proyecto, Identificar y refinar requerimientos	Definir visión, Planificar proyecto, Encontrar y bocetar requerimientos	0.3
<b>PG2.1.2 Planificar la performance del proceso</b>	Equipo integrado, Visión compartida, Planificación en dos niveles, Desarrollo conducido por UC	Iniciar el proyecto, Identificar y refinar requerimientos	Definir visión, Planificar proyecto, Encontrar y bocetar requerimientos	0.3
<b>PG2.1.3 Implementar las actividades definidas</b>	Equipo integrado, Visión compartida, Planificación en dos niveles, Desarrollo conducido por UC	Iniciar el proyecto, Identificar y refinar requerimientos	Definir visión, Planificar proyecto, Encontrar y bocetar requerimientos	0.3
<b>PG2.1.4 Gestionar la ejecución de las actividades</b>	Equipo integrado, Visión compartida, Planificación en dos niveles, Desarrollo conducido por UC	Iniciar el proyecto, Identificar y refinar requerimientos	Definir visión, Planificar proyecto, Encontrar y bocetar requerimientos	0.3
Score combinado para los atributos (PG2.1.1 to 2.1.4):				0.3
<b>AP 2.2 Atributo de Gestión de producto de trabajo</b>				
<b>PG2.2.1 Identificar requisitos</b>	Equipo integrado, Visión compartida, Planificación en dos niveles, Desarrollo conducido por UC	Iniciar el proyecto, Identificar y refinar requerimientos	Definir visión, Planificar proyecto, Encontrar y bocetar requerimientos	0.3
<b>PG2.2.2 Identificar las actividades necesarias</b>	Equipo integrado, Visión compartida, Planificación en dos niveles, Desarrollo conducido por UC	Iniciar el proyecto, Identificar y refinar requerimientos	Definir visión, Planificar proyecto, Encontrar y bocetar requerimientos	0.3
<b>PG2.2.3 Gestionar la configuración de productos de trabajo</b>	Equipo integrado, Visión compartida, Planificación en dos niveles, Desarrollo conducido por UC	Iniciar el proyecto, Identificar y refinar requerimientos	Definir visión, Planificar proyecto, Encontrar y bocetar requerimientos	0.3
<b>PG2.2.4 Gestionar la calidad de productos de trabajo</b>	Equipo integrado, Visión compartida, Planificación en dos niveles, Desarrollo conducido por UC	Iniciar el proyecto, Identificar y refinar requerimientos	Definir visión, Planificar proyecto, Encontrar y bocetar requerimientos	0.3
Score combinado para los atributos (PG2.2.1 to 2.2.4):				0.3
Score combinado para este nivel				0.3
<b>Nivel 3 Proceso establecido</b>				
<b>AP3.1 Atributo de definición de proceso</b>				
<b>PG3.1.1 Identificar la definición del proceso standard</b>				
<b>PG3.1.2 Adaptar el proceso standard</b>				
<b>PG3.1.3 Implementar el proceso definido</b>				
<b>PG3.1.4 Proveer feedback</b>				
Score combinado para los atributos (PG3.1.1 to 3.1.4):				0
<b>AP 3.2 Atributo de recursos de proceso</b>				
<b>PG3.2.1 Definir las competencias de rhh</b>				
<b>PG3.2.2 Definir los requisitos de infraestructura del proceso</b>				
<b>PG3.2.3 Proveer rhh cualificados</b>				
<b>PG3.2.4 Proveer infraestructura de proceso adecuada</b>				
Score combinado para los atributos (PG3.2.1 to 3.2.4):				0
Score combinado para este nivel				0

Figura C.1: Resultados para el Proceso DCU 1

Prácticas y procesos		DCU2 Planificar y Gestionar el proceso	DCU	Score
Nivel 1 Proceso realizado		Prácticas	Contenidos de proceso	Contenidos de Método
<b>AP1.1 Atributo de performance del proceso</b>				
<b>DCU 2.1 Consultar a los stakeholders</b>				
	Equipo integrado. Visión compartida. Planificación en dos niveles. Desarrollo conducido por UC. Desarrollo iterativo. diseño evolutivo. Testing concurrente. Gestion de cambios.	Planificar el proyecto. Planificar y gestionar la iteración. Identificar y refinar requerimientos. Acordar el abordaje técnico.	Planificar proyecto. Planificar iteración. Gestionar iteración. Evaluar resultados. Definir la visión. Encontrar y bocetar requerimientos. Detallar arquitectura. Diseñar la solución. Pedir cambio. Crear casos de prueba.	0.7
<b>DCU.2.2 Planificar la participación de usuarios</b>				
	Planificación en dos niveles. Desarrollo iterativo. Visión compartida. Equipo integrado. Desarrollo conducido por UC. Testing concurrente	Iniciar el proyecto. Planificar y gestionar la iteración. Identificar y refinar requerimientos. Testear la solución	Planificar proyecto. Planificar iteración. Definir la visión. Encontrar y bocetar requerimientos. Crear casos de prueba.	0.7
<b>DCU.2.3 Seleccionar metodos y tecnicas centradas en usuarios</b>				
	Planificación en dos niveles. Desarrollo iterativo	Iniciar el proyecto. Planificar y gestionar la iteración.	Planificar proyecto. Planificar iteración.	0.3
<b>DCU.2.4 Asegurar el enfoque centrado en usuario en el equipo</b>				
	Planificación en dos niveles	Iniciar el proyecto	Planificar proyecto.	0.3
<b>DCU 2.5 Planificar el proceso de diseño centrado en usuario</b>				
	Planificación en dos niveles. Desarrollo iterativo	Iniciar el proyecto. Planificar y gestionar la iteración.	Planificar proyecto. Planificar iteración.	0.3
<b>DCU 2.6 Gestionar el proceso centrado en usuario</b>				
	Planificación en dos niveles. Desarrollo iterativo	Iniciar el proyecto. Planificar y gestionar la iteración.	Planificar proyecto. Planificar iteración.	0.3
<b>DCU.2.7 Abogar por las actividades centradas en usuario</b>				
	Planificación en dos niveles. Desarrollo iterativo	Iniciar el proyecto. Planificar y gestionar la iteración.	Planificar proyecto. Planificar iteración.	0.3
<b>DCU 2.8 Proveer soporte para el diseño centrado en usuario</b>				
	Planificación en dos niveles. Desarrollo iterativo	Iniciar el proyecto. Planificar y gestionar la iteración.	Planificar proyecto. Planificar iteración.	0.3
Score combinado para los atributos (DCU 2.1 to 2.8)				0.4
Score combinado para esta nivel				0.4
<b>Nivel 2 Proceso Gestionado</b>				
<b>AP2.1 Atributo de gestión de la performance</b>				
<b>PG2 1.1 Identificar requerimientos de recursos</b>				
	visión compartida. Planificación en dos niveles. Desarrollo iterativo	Iniciar el proyecto. Planificar y gestionar la iteración.	Planificar proyecto. Planificar iteración. Definir la visión.	0.3
<b>PG2 1.2 Planificar la performance del proceso</b>				
	visión compartida. Planificación en dos niveles. Desarrollo iterativo	Iniciar el proyecto. Planificar y gestionar la iteración.	Planificar proyecto. Planificar iteración. Definir la visión.	0.3
<b>PG2 1.3 Implementar las actividades definidas</b>				
	Equipo integrado. Visión compartida. Planificación en dos niveles. Desarrollo conducido por UC. Desarrollo iterativo. diseño evolutivo. Testing concurrente. Gestion de cambios.	Planificar el proyecto. Planificar y gestionar la iteración. Identificar y refinar requerimientos. Acordar el abordaje técnico	Planificar proyecto. Planificar iteración. Gestionar iteración. Evaluar resultados. Definir la visión. Encontrar y bocetar requerimientos. Detallar arquitectura. Diseñar la solución. Pedir cambio. Crear casos de prueba.	0.3
<b>PG2 1.4 Gestionar la ejecución de las actividades</b>				
	Desarrollo iterativo	Iniciar el proyecto. Identificar y refinar requerimientos	Gestionar iteración	0.3
Score combinado para los atributos (PG2 1.1 to 2.1.4)				0.3
<b>AP 2.2 Atributo de Gestión de productos de trabajo</b>				
<b>PG2 2.1 Identificar requisitos</b>				
	visión compartida. Planificación en dos niveles. Desarrollo iterativo	Iniciar el proyecto. Planificar y gestionar la iteración.	Planificar proyecto. Planificar iteración. Definir la visión.	0.3
<b>PG2.2.2 Identificar las actividades necesarias</b>				
	visión compartida. Planificación en dos niveles. Desarrollo iterativo	Iniciar el proyecto. Planificar y gestionar la iteración.	Planificar proyecto. Planificar iteración. Definir la visión.	0.3
<b>PG2.2.3 Gestionar la configuración de productos de trabajo</b>				
	visión compartida. Planificación en dos niveles. Desarrollo iterativo	Iniciar el proyecto. Planificar y gestionar la iteración.	Gestionar iteración. Evaluar resultados. Pedir cambio.	0.3
<b>PG2.2.4 Gestionar la calidad de productos de trabajo</b>				
	Desarrollo iterativo	Planificar y gestionar la iteración.	Gestionar iteración	0.3
Score combinado para los atributos (PG2 2.1 to 2.2.4)				0.3
Score combinado para esta nivel				0.3
<b>Nivel 3 Proceso automatizado</b>				
<b>AP3.1 Atributo de definición de proceso</b>				
<b>PG3 1.1 Identificar la definición del proceso standard</b>				
<b>PG3.1.2 Adaptar el proceso standard</b>				
<b>PG3 1.3 Implementar el proceso definido</b>				
<b>PG3.1.4 Proveer feedback</b>				
Score combinado para los atributos (PG3 1.1 to 3.1.4)				0
<b>AP 3.2 Atributo de recursos de proceso</b>				
<b>PG3.2.1 Definir las competencias de rhh</b>				
	Planificación en dos niveles	Iniciar el proyecto.	Planificar el proyecto.	0.3
<b>PG3.2.2 Definir los requisitos de infraestructura del proceso</b>				
	Planificación en dos niveles	Iniciar el proyecto.	Planificar el proyecto.	0.3
<b>PG3 2.3 Proveer rhh cualificados</b>				
	Planificación en dos niveles	Iniciar el proyecto.	Planificar el proyecto.	0.3
<b>PG3.2.4 Proveer infraestructura de proceso adecuada</b>				
	Planificación en dos niveles	Iniciar el proyecto.	Planificar el proyecto.	0.3
Score combinado para los atributos (PG3 2.1 to 3.2.4)				0.3
Score combinado para esta nivel				0.3

Figura C.2: Evidencia para determinar Capacidad en DCU2

DCU 3 Especificar los requerimientos de usuario y de organización					
Nivel 1 Proceso realizado	Prácticas y procesos	Prácticas	Contenidos de proceso	Contenidos de Método	Score
<b>AP1.1 Atributo de performance del proceso</b>					
DCU 3.1 Clarificar y documentar las metas del sistema	visión integrada.		Iniciar el proyecto.	Definir visión.	0.7
DCU 3.2 Definir los stakeholders	visión integrada. Desarrollo conducido por UC		Iniciar el proyecto. Identificar y refinar requerimientos.	Definir visión. Encontrar y bocetar requerimientos.	0.3
DCU 3.3 Evaluar los riesgos para stakeholders	Planificación en dos niveles. Desarrollo iterativo. Visión integradora. Desarrollo conducido por UC. Gestión de cambios.		Iniciar el proyecto. Identificar y refinar requerimientos. Planificar y gestionar la iteración.	Planificar proyecto. Planificar iteración. Definir visión. Encontrar y bocetar requerimientos. Detallar requerimientos. Pedir cambios.	0.7
DCU 3.4 Definir el sistema	Planificación en dos niveles. Desarrollo iterativo. Visión integradora. Desarrollo conducido por UC. Gestión de cambios.		Iniciar el proyecto. Identificar y refinar requerimientos. Planificar y gestionar la iteración.	Planificar proyecto. Planificar iteración. Definir visión. Encontrar y bocetar requerimientos. Detallar requerimientos.	0.3
DCU 3.5 Generar requerimientos de stakeholders y de la organización	Planificación en dos niveles. Desarrollo iterativo. Visión integradora. Desarrollo conducido por UC. Testing concurrente. Gestión de cambios.		Iniciar el proyecto. Identificar y refinar requerimientos. Planificar y gestionar la iteración. Desarrollar incremento de solución. Testear solución.	Definir visión. Encontrar y bocetar requerimientos. Detallar requerimientos. Implementar solución. Pedir cambios. Ejecutar pruebas.	0.7
DCU 3.6 Establecer la calidad en objetivos de uso	Planificación en dos niveles. Desarrollo iterativo. Visión integradora. Desarrollo conducido por UC. Testing concurrente. Gestión de cambios.		Iniciar el proyecto. Identificar y refinar requerimientos. Planificar y gestionar la iteración. Desarrollar incremento de solución. Testear solución.	Planificar proyecto. Planificar iteración. Definir visión. Encontrar y bocetar requerimientos. Detallar requerimientos. Implementar solución. Pedir cambios. Ejecutar pruebas.	0.3
Score combinado para los atributos (DCU.3.1 to 3.6):					0.5
Score combinado para este nivel					0.5
<b>Nivel 2 Proceso Gestionado</b>					
<b>AP2.1 Atributo de gestión de la performance</b>					
PG2.1.1 Identificar requerimientos de recursos	Planificación en dos niveles. Desarrollo iterativo. Visión integradora.		Iniciar el proyecto. Identificar y refinar requerimientos.	Planificar proyecto. Planificar iteración. Definir la visión.	0.3
PG2.1.2 Planificar la performance del proceso	Planificación en dos niveles. Desarrollo iterativo. Visión integradora.		Iniciar el proyecto. Identificar y refinar requerimientos.	Planificar proyecto. Planificar iteración. Definir la visión.	0.3
PG2.1.3 Implementar las actividades definidas	Planificación en dos niveles. Desarrollo iterativo. Visión integradora. Desarrollo conducido por UC. Arquitecturas evolutiva. diseño evolutivo. Testing concurrente. Gestión de cambios.		Iniciar el proyecto. Identificar y refinar requerimientos.	Planificar proyecto. Planificar iteración. Gestionar iteración. Evaluar resultados. Definir la visión. Encontrar y bocetar requerimientos. Detallar requerimientos. Bocetar la arquitectura. Diseñar la solución. Pedir cambio. Crear casos de prueba.	0.3
PG2.1.4 Gestionar la ejecución de las actividades	Desarrollo iterativo.		Planificar y gestionar la iteración.	Gestionar iteración.	0.3
Score combinado para los atributos (PG2.1.1 to 2.1.4):					0.3
Score combinado para este nivel					0.3
<b>AP 2.2 Atributo de Gestión de producto de trabajo</b>					
PG2.2.1 Identificar requisitos	Planificación en dos niveles. Desarrollo iterativo. Visión integradora.		Iniciar el proyecto. Identificar y refinar requerimientos.	Planificar proyecto. Planificar iteración. Definir la visión.	0.3
PG2.2.2 Identificar las actividades necesarias	Planificación en dos niveles. Desarrollo iterativo. Visión integradora.		Iniciar el proyecto. Identificar y refinar requerimientos.	Planificar proyecto. Planificar iteración. Definir la visión.	0.3
PG2.2.3 Gestionar la configuración de productos de trabajo	Planificación en dos niveles. Desarrollo iterativo. Visión integradora.		Iniciar el proyecto. Identificar y refinar requerimientos.	Gestionar iteración. Evaluar resultados. Pedir cambio.	0.3
PG2.2.4 Gestionar la calidad de productos de trabajo			Planificar y gestionar la iteración.	Gestionar iteración.	0.3
Score combinado para los atributos (PG2.2.1 to 2.2.4):					0.3
Score combinado para este nivel					0.3
<b>Nivel 3 Proceso establecido</b>					
<b>AP3.1 Atributo de definición de proceso</b>					
PG3.1.1 Identificar la definición del proceso standard					
PG3.1.2 Adaptar el proceso standard					
PG3.1.3 Implementar el proceso definido					
PG3.1.4 Proveer feedback					
Score combinado para los atributos (PG3.1.1 to 3.1.4):					0
<b>AP 3.2 Atributo de recursos de proceso</b>					
PG3.2.1 Definir las competencias de mhh	Planificación en dos niveles.			Planificar el proyecto.	0.3
PG3.2.2 Definir los requisitos de infraestructura del proceso	Planificación en dos niveles.			Planificar el proyecto.	0.3
PG3.2.3 Proveer mhh cualificados	Planificación en dos niveles.			Planificar el proyecto.	0.3
PG3.2.4 Proveer infraestructura de proceso adecuada	Planificación en dos niveles.			Planificar el proyecto.	0.3
Score combinado para los atributos (PG3.2.1 to 3.2.4):					0.3
Score combinado para este nivel					0.15

Figura C.3: Resultados del assessment de DCU 3

DCU 4 Entender y especificar el contenido de uso				
Prácticas y procesos	Prácticas	Contenidos de proceso	Contenidos de Método	Score
<b>Nivel 1: Proceso realizado</b>				
<b>AP1.1 Atributo de performance del proceso</b>				
DCU.4.1 Identificar y documentar tareas de usuario	Visión integradora. Desarrollo conducido por UC	Iniciar el proyecto. Identificar y refinar requerimientos	Definir visión. Encontrar y bocetar requerimientos. Detallar requerimientos	0.7
DCU.4.2 Identificar y documentar atributos significativos de usuario	Visión integradora. Desarrollo conducido por UC	Iniciar el proyecto. Identificar y refinar requerimientos	Definir visión. Encontrar y bocetar requerimientos. Detallar requerimientos	0.3
DCU.4.3 Identificar y documentar entorno organizacional	Visión integradora. Desarrollo conducido por UC	Iniciar el proyecto. Identificar y refinar requerimientos	Definir visión. Encontrar y bocetar requerimientos. Detallar requerimientos	0.3
DCU.4.4 Identificar y documentar el entorno técnico	Visión integradora. Desarrollo conducido por UC	Iniciar el proyecto. Identificar y refinar requerimientos	Definir visión. Encontrar y bocetar requerimientos. Detallar requerimientos	0.3
DCU.4.5 Identificar y documentar entorno físico	Visión integradora. Desarrollo conducido por UC	Iniciar el proyecto. Identificar y refinar requerimientos	Definir visión. Encontrar y bocetar requerimientos. Detallar requerimientos	0.3
Score combinado para los atributos (DCU 4.1 to 4.5)				0.4
Score combinado para este nivel				0.4
<b>Nivel 2: Proceso Gestionado</b>				
<b>AP2.1 Atributo de gestión de la performance</b>				
PG2.1.1 Identificar requerimientos de recursos	Visión integradora. Desarrollo conducido por UC	Iniciar el proyecto. Identificar y refinar requerimientos	Definir visión. Encontrar y bocetar requerimientos. Detallar requerimientos	0.7
PG2.1.2 Planificar la performance del proceso	Visión integradora. Desarrollo conducido por UC	Iniciar el proyecto. Identificar y refinar requerimientos	Definir visión. Encontrar y bocetar requerimientos. Detallar requerimientos	0.3
PG2.1.3 Implementar las actividades definidas	Visión integradora. Desarrollo conducido por UC	Iniciar el proyecto. Identificar y refinar requerimientos	Definir visión. Encontrar y bocetar requerimientos. Detallar requerimientos	0.3
PG2.1.4 Gestionar la ejecución de las actividades	Visión integradora. Desarrollo conducido por UC	Iniciar el proyecto. Identificar y refinar requerimientos	Definir visión. Encontrar y bocetar requerimientos. Detallar requerimientos	0.3
Score combinado para los atributos (PG2.1.1 to 2.1.4)				0.3
<b>AP 2.2 Atributo de Gestión de producto de trabajo</b>				
PG2.2.1 Identificar requisitos	Visión integradora. Desarrollo conducido por UC	Iniciar el proyecto. Identificar y refinar requerimientos	Definir visión. Encontrar y bocetar requerimientos. Detallar requerimientos	0.7
PG2.2.2 Identificar las actividades necesarias	Visión integradora. Desarrollo conducido por UC	Iniciar el proyecto. Identificar y refinar requerimientos	Definir visión. Encontrar y bocetar requerimientos. Detallar requerimientos	0.3
PG2.2.3 Gestionar la configuración de productos de trabajo	Visión integradora. Desarrollo conducido por UC	Iniciar el proyecto. Identificar y refinar requerimientos	Definir visión. Encontrar y bocetar requerimientos. Detallar requerimientos	0.3
PG2.2.4 Gestionar la calidad de productos de trabajo	Visión integradora. Desarrollo conducido por UC	Iniciar el proyecto. Identificar y refinar requerimientos	Definir visión. Encontrar y bocetar requerimientos. Detallar requerimientos	0.3
Score combinado para los atributos (PG2.2.1 to 2.2.4)				0.4
Score combinado para este nivel				0.4
<b>Nivel 3: Proceso establecido</b>				
<b>AP3.1 Atributo de definición de proceso</b>				
PG3.1.1 Identificar la definición del proceso standard				
PG3.1.2 Adaptar el proceso standard				
PG3.1.3 Implementar el proceso definido				
PG3.1.4 Proveer feedback				
Score combinado para los atributos (PG3.1.1 to 3.1.4)				0
<b>AP 3.2 Atributo de recursos de proceso</b>				
PG3.2.1 Definir las competencias de rhh	Visión integradora. Desarrollo conducido por UC	Iniciar el proyecto. Identificar y refinar requerimientos	Definir visión. Encontrar y bocetar requerimientos. Detallar requerimientos	0.7
PG3.2.2 Definir los requisitos de infraestructura del proceso	Visión integradora. Desarrollo conducido por UC	Iniciar el proyecto. Identificar y refinar requerimientos	Definir visión. Encontrar y bocetar requerimientos. Detallar requerimientos	0.3
PG3.2.3 Proveer rhh cualificados	Visión integradora. Desarrollo conducido por UC	Iniciar el proyecto. Identificar y refinar requerimientos	Definir visión. Encontrar y bocetar requerimientos. Detallar requerimientos	0.3
PG3.2.4 Proveer infraestructura de proceso adecuada				
Score combinado para los atributos (PG3.2.1 to 3.2.4)				0.4
Score combinado para este nivel				0.2

Figura C.4: Evidencia de capacidades OpenUP en DCU 4

DCU 5 Producir soluciones de diseño				
Prácticas y procesos	Prácticas	Contenidos de proceso	Contenidos de Método	Score
<b>Nivel 1: Proceso realizado</b>				
<b>AP1.1 Atributo de performance del proceso</b>				
DCU 5.1 Alocar funciones	Arquitectura evolutiva, diseño evolutivo	Acordar abordaje técnico, Desarrollar la arquitectura, Diseñar incremento de solución	Bocetar arquitectura, Diseñar solución	0.3
DCU 5.2 Producir modelo de tareas compuesto				
DCU 5.3 Explorar el diseño del sistema	Arquitectura evolutiva, diseño evolutivo	Acordar abordaje técnico, Desarrollar la arquitectura, Diseñar incremento de solución	Bocetar arquitectura, Diseñar solución	0.7
DCU 5.4 Desarrollar soluciones de diseño	diseño evolutivo, Testing concurrente	Diseñar incremento de solución, Testear solución	Diseñar solución, Ejecutar pruebas	0.3
DCU 5.5 Especificar el sistema				
DCU 5.6 Desarrollar prototipos	Testing concurrente	Testear solución	Ejecutar pruebas	0.3
DCU 5.7 Desarrollar entrenamiento de usuarios				
DCU 5.8 Desarrollar soporte de usuarios				
Score combinado para los atributos (DCU 5.1 to 5.8)				0.3
Score combinado para este nivel				0.3
<b>Nivel 2: Proceso testado</b>				
<b>AP2.1 Atributo de gestión de la performance</b>				
PG2.1.1 Identificar requerimientos de recursos	Arquitectura evolutiva, diseño evolutivo	Acordar abordaje técnico, Desarrollar la arquitectura, Diseñar incremento de solución	Bocetar arquitectura, Refinar la arquitectura, Diseñar solución	0.3
PG2.1.2 Planificar la performance del proceso	Arquitectura evolutiva, diseño evolutivo, Desarrollo iterativo	Acordar abordaje técnico, Desarrollar la arquitectura, Diseñar incremento de solución	Bocetar arquitectura, Refinar la arquitectura, Diseñar solución, Planificar la iteración	0.3
PG2.1.3 Implementar las actividades definidas	Arquitectura evolutiva, diseño evolutivo	Acordar abordaje técnico, Desarrollar la arquitectura, Diseñar incremento de solución	Bocetar arquitectura, Refinar la arquitectura, Diseñar solución	0.3
PG2.1.4 Gestionar la ejecución de las actividades	Desarrollo iterativo	Diseñar incremento de solución	Gestionar la iteración	0.3
Score combinado para los atributos (PG2.1.1 to 2.1.4)				0.3
<b>AP 2.2 Atributo de Gestión de producto de trabajo</b>				
PG2.2.1 Identificar requisitos	Arquitectura evolutiva, diseño evolutivo	Acordar abordaje técnico, Desarrollar la arquitectura, Diseñar incremento de solución	Bocetar arquitectura, Refinar la arquitectura, Diseñar solución	0.3
PG2.2.2 Identificar las actividades necesarias	Arquitectura evolutiva, diseño evolutivo	Acordar abordaje técnico, Desarrollar la arquitectura, Diseñar incremento de solución	Bocetar arquitectura, Refinar la arquitectura, Diseñar solución	0.3
PG2.2.3 Gestionar la configuración de productos de trabajo	Arquitectura evolutiva, diseño evolutivo	Acordar abordaje técnico, Desarrollar la arquitectura, Diseñar incremento de solución	Bocetar arquitectura, Refinar la arquitectura, Diseñar solución	0.3
PG2.2.4 Gestionar la calidad de productos de trabajo	Arquitectura evolutiva, diseño evolutivo	Acordar abordaje técnico, Desarrollar la arquitectura, Diseñar incremento de solución	Bocetar arquitectura, Refinar la arquitectura, Diseñar solución	0.3
Score combinado para los atributos (PG2.2.1 to 2.2.4)				0.3
Score combinado para este nivel				0.3
<b>Nivel 3: Proceso establecido</b>				
<b>AP3.1 Atributo de definición de proceso</b>				
PG3.1.1 Identificar la definición del proceso standard				
PG3.1.2 Adaptar el proceso standard				
PG3.1.3 Implementar el proceso definido				
PG3.1.4 Proveer feedback				0
Score combinado para los atributos (PG3.1.1 to 3.1.4)				0
<b>AP 3.2 Atributo de recursos de proceso</b>				
PG3.2.1 Definir las competencias de rhh				
PG3.2.2 Definir los requisitos de infraestructura del proceso				
PG3.2.3 Proveer rhh cualificados				
PG3.2.4 Proveer infraestructura de proceso adecuada				
Score combinado para los atributos (PG3.2.1 to 3.2.4)				0
Score combinado para este nivel				0

Figura C.5: Evidencia de capacidad OpenUP en DCU 5

DCU 6 Evaluar el diseño contra los requerimientos				
Prácticas y procesos	Prácticas	Contenidos de proceso	Contenidos de Método	Score
<b>Nivel 1: Proceso realizado</b>				
<b>AP1.1 Atributo de performance del proceso</b>				
DCU 6.1 Especificar y validar el contexto de evaluación	Testing concurrente	Testear la solución	Crear casos de prueba. Implementar guiones de pruebas.	0.3
DCU 6.2 Evaluar prototipos tempranos para definir los requerimientos de diseño				
DCU 6.3 Evaluar prototipos para mejorar el diseño	Testing concurrente. Gestión de cambios.	Testear la solución. Tareas continuas.	Crear casos de prueba. Ejecutar pruebas. Pedir cambios.	0.3
DCU 6.4 Evaluar el sistema para chequear que se satisfacen los requerimientos	Testing concurrente. Gestión de cambios.	Testear la solución. Tareas continuas.	Crear casos de prueba. Ejecutar pruebas. Pedir cambios.	0.3
DCU 6.5 Evaluar el sistema para chequear que se siguieron las prácticas requeridas				0.3
DCU 6.6 Evaluar el sistema en uso para asegurar que continua satisfaciendo necesidades de usuario y organización				
Score combinado para los atributos (DCU 6.1 to 6.6):				0.3
Score combinado para este nivel:				0.3
<b>Nivel 2: Proceso Gestionado</b>				
<b>AP2.1 Atributo de gestión de la performance</b>				
PG2.1.1 Identificar requerimientos de recursos	Planificación en dos niveles. Desarrollo iterativo. Testing concurrente	Iniciar el proyecto. Identificar y refinar requerimientos.	Planificar el proyecto. Planificar iteración. Crear casos de prueba. Implementar guiones de pruebas.	0.3
PG2.1.2 Planificar la performance del proceso	Planificación en dos niveles. Desarrollo iterativo. Testing concurrente	Iniciar el proyecto. Identificar y refinar requerimientos.	Planificar el proyecto. Planificar iteración. Crear casos de prueba. Implementar guiones de pruebas.	0.3
PG2.1.3 Implementar las actividades definidas	Planificación en dos niveles. Desarrollo iterativo. Testing concurrente	Iniciar el proyecto. Identificar y refinar requerimientos.	Planificar el proyecto. Planificar iteración. Crear casos de prueba. Implementar guiones de pruebas.	0.3
PG2.1.4 Gestionar la ejecución de las actividades	Planificación en dos niveles. Desarrollo iterativo. Testing concurrente	Iniciar el proyecto. Identificar y refinar requerimientos.	Planificar el proyecto. Planificar iteración. Crear casos de prueba. Implementar guiones de pruebas.	0.3
Score combinado para los atributos (PG2.1.1 to 2.1.4):				0.3
Score combinado para este nivel:				0.3
<b>AP 2.2 Atributo de Gestión de producto de trabajo</b>				
PG2.2.1 Identificar requisitos	Planificación en dos niveles. Desarrollo iterativo. Testing concurrente	Iniciar el proyecto. Identificar y refinar requerimientos.	Planificar el proyecto. Planificar iteración. Crear casos de prueba. Implementar guiones de pruebas.	0.3
PG2.2.2 Identificar las actividades necesarias	Planificación en dos niveles. Desarrollo iterativo. Testing concurrente	Iniciar el proyecto. Identificar y refinar requerimientos.	Planificar el proyecto. Planificar iteración. Crear casos de prueba. Implementar guiones de pruebas.	0.3
PG2.2.3 Gestionar la configuración de productos de trabajo	Planificación en dos niveles. Desarrollo iterativo. Testing concurrente	Iniciar el proyecto. Identificar y refinar requerimientos.	Planificar el proyecto. Planificar iteración. Crear casos de prueba. Implementar guiones de pruebas.	0.3
PG2.2.4 Gestionar la calidad de productos de trabajo	Planificación en dos niveles. Desarrollo iterativo. Testing concurrente	Iniciar el proyecto. Identificar y refinar requerimientos.	Planificar el proyecto. Planificar iteración. Crear casos de prueba. Implementar guiones de pruebas.	0.3
Score combinado para los atributos (PG2.2.1 to 2.2.4):				0.3
Score combinado para este nivel:				0.3
<b>Nivel 3: Proceso establecido</b>				
<b>AP3.1 Atributo de definición de proceso</b>				
PG3.1.1 Identificar la definición del proceso standard				
PG3.1.2 Adaptar el proceso standard				
PG3.1.3 Implementar el proceso definido				
PG3.1.4 Proveer feedback				
Score combinado para los atributos (PG3.1.1 to 3.1.4):				0
<b>AP 3.2 Atributo de recursos de proceso</b>				
PG3.2.1 Definir las competencias de mhh				
PG3.2.2 Definir los requisitos de infraestructura del proceso				
PG3.2.3 Proveer mhh cualificados				
PG3.2.4 Proveer infraestructura de proceso adecuada				
Score combinado para los atributos (PG3.2.1 to 3.2.4):				0
Score combinado para este nivel:				0

Figura C.6: Resultados del assessment para DCU 6

# Apéndice D

## Software adjunto

Se adjunta al documento de la Tesis un CD que contiene los siguientes productos:

- El documento PDF con el texto de la Tesis
- Tres sitios web publicando el despliegue de las Configuraciones OpenUP/MMU-ISO Nx
- Un sitio web con el despliegue de todas las Prácticas de la EPL incluyendo DesCU
- El EPF Composer y los plugins descritos en el Capítulo 8

**Nota:** Para facilitar la identificación del contenido que representa la extensión desarrollada en la Tesis el mismo se redactó en español y se inserta en la versión original en inglés de la EPL y OpenUP.