



# Ingeniería de Procesos Software

Universidad de Castilla-La Mancha

Escuela Superior de Informática

Departamento de Tecnologías y Sistemas de Información

Francisco Ruiz



## Objetivos

- **Conocer** los principios e importancia de la **IPS**.
- Comprender el interés de trabajar con esta nueva tecnología en las investigaciones metodológicas en ingeniería del software.
- Aprender las características del **estándar SPEM** como base para el modelado de ciclos de vida, metodologías, procesos, métodos, buenas prácticas, técnicas, etc.
- Presentar la herramienta **EPF Composer** como editor basado en SPEM.



- Introducción
  - Procesos Software
  - Modelos de Procesos Software
  - Principios de la Ingeniería de Procesos Software
- Fundamentos de SPEM
  - Características básicas
  - Metamodelo vs perfil UML
  - Usos
- Elementos de SPEM
  - Organización
  - Tareas
  - Roles
  - Productos de Trabajo
  - Guías
  - Categorías.
- Composición de Procesos.
  - Estructura
  - Actividades
  - Fases
  - Iteraciones
  - Hitos
  - Patrones de Proceso
  - Procesos para Despliegue
- EPF Composer
  - Funcionalidad básica
  - Demostración
  - Características avanzadas



- OMG (2007): Software & Systems Process Engineering Metamodel Specification (**SPEM**); version 2.0, agosto 2007.
  - <http://www.omg.org/cgi-bin/doc?ptc/07-08-07>
- Ruiz, F. y Verdugo, J. (2008): Guía de Uso de SPEM 2 con EPF Composer, versión 3. UCLM.
  - Web de la asignatura.
- Software Process Engineering in the Real World
  - <http://www.softwareprocessengineering.com/>



• **Introducción**

- **Procesos Software**
- **Modelos de Procesos Software**
- **Principios de la Ingeniería de Procesos Software**

• **Fundamentos de SPEM**

- Características básicas
- Metamodelo vs perfil UML
- Usos

• **Elementos de SPEM**

- Organización
- Tareas
- Roles
- Productos de Trabajo
- Guías
- Categorías.

• **Composición de Procesos.**

- Estructura
- Actividades
- Fases
- Iteraciones
- Hitos
- Patrones de Proceso
- Procesos para Despliegue

• **EPF Composer**

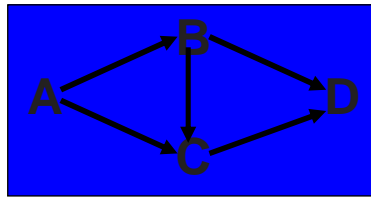
- Funcionalidad básica
- Demostración
- Características avanzadas



Un Proceso Software (PS) es

**Un conjunto coherente de políticas, estructuras organizacionales, tecnologías, procedimientos y artefactos que son necesarios para concebir, desarrollar, instalar y mantener un producto software.**

**(Fugetta, 2000)**



**Métodos y Procedimientos que definen la relaciones entre las Tareas.**

**Personal**



**PROCESO SW**

**Herramientas y Metodologías.**



- Son complejos:
- No son procesos de producción:
  - Dirigidos por excepciones,
  - Muy determinados por circunstancias impredecibles,
  - Cada uno con sus peculiaridades.
- No son procesos de ingeniería “pura”:
  - Desconocemos las abstracciones adecuadas,
  - Dependen demasiado de demasiada gente,
  - Diseño y producción no están claramente separados,
  - Presupuestos, calendarios, calidad no pueden ser planificados de forma fiable.



- No son (*completamente*) procesos creativos:
  - Algunas partes pueden ser descritas en detalle,
  - Algunos procedimientos han sido impuestos.
- Están basados en descubrimientos que dependen de la comunicación, coordinación y cooperación dentro de marcos de trabajo predefinidos:
  - Los entregables generan nuevos requerimientos,
  - Los costes del cambio del software no suelen reconocerse,
  - El éxito depende de la implicación del usuario y de la coordinación de muchos roles (ventas, desarrollo técnico, cliente, etc.).



### Gestión y Mejora:

- Años 80
- Creciente Importancia en la industria Software por la calidad
- Aparecen estándares como la familia ISO 9000 y modelos de madurez como CMM (finales de los 80)
- Estándares ISO 9000
  - Certificación Calidad → Garantía de que una organización software entregará productos de calidad
- Estos estándares y modelos incluyen prácticas que facilitan la gestión de los procesos software
- Aparecen ciertas limitaciones:
  - ¿una organización con certificación de calidad obtendrá siempre productos de alta calidad?
  - Incremento de Burocracia



## Utilidad en **Procesos Software**

- Para poder integrar varios PS, cada uno con su MP.
- Para la mejora de PS.
  - Permitiendo la evolución del modelo de un PS.
  - Pudiendo gestionar de forma integrada el proceso y su ciclo de vida (diseño, despliegue, ejecución, automatización, mejora, ..).
- Para construir Plataformas más potentes.
  - Haciendo que el repositorio sea más genérico y tenga más capacidad semántica.
  - Permitiendo que todas las herramientas (CASE, gestión de proyectos, ...) compartan los modelos.
  - Pudiendo realizar procesamiento y transformaciones directamente sobre los modelos.

11



Pasar de la **gestión de modelos** “contemplativa” a la “productiva” significa que la mayoría de los pasos de la cadena de producción y mantenimiento de software puede considerarse como operaciones definidas de forma precisa sobre artefactos del modelo.

Es muy diferente lo que se puede hacer con la descripción de una tarea según el formato en que esté:

- Texto en PDF o DOC
- Archivo XML basado en SPEM
- Web navegable
- ...

12



**Tarea DSI 7.1: Verificación de las Especificaciones de Diseño**

El objetivo de esta tarea es asegurar la calidad formal de los distintos modelos, conforme a la técnica seguida para la elaboración de cada producto y a las normas y estándares especificados en el catálogo de normas.

**Productos**

De entrada

- Catálogo de Requisitos (DSI 1.2)
- Catálogo de Excepciones (DSI 1.3)
- Catálogo de Normas (DSI 1.4)
- Diseño de la Arquitectura del Sistema (DSI 1.5)
- Entorno Tecnológico del Sistema (DSI 1.6)
- Diseño Detallado de Subsistemas de Soporte (DSI 2.1)
- Modelo Físico de Datos Optimizado (DSI 6.3)
- Esquemas Físicos de Datos (DSI 6.4)
- Asignación de Esquemas Físicos de Datos a Nodos (DSI 6.4)

En Diseño Estructurado:

- Diseño de la Arquitectura Modular (DSI 5.2)
- Diseño de Interfaz de Usuario (DSI 5.3)

En Diseño Orientado a Objetos:

- Diseño de la Realización de los Casos de Uso (DSI 3.4)
- Diseño de Interfaz de Usuario (DSI 3.3)
- Modelo de Clases de Diseño (DSI 4.6)
- Comportamiento de Clases de Diseño (DSI 4.4)

De salida

- Entorno Tecnológico del Sistema
- Diseño de la Arquitectura del Sistema
- Diseño Detallado de Subsistemas de Soporte
- Modelo Físico de Datos Optimizado
- Esquemas Físicos de Datos
- Asignación de Esquemas Físicos de Datos a Nodos
- Diseño de Interfaz de Usuario

En Diseño Estructurado:

- Diseño de la Arquitectura Modular
- En Diseño Orientado a Objetos:
- Diseño de la Realización de los Casos de Uso
- Modelo de Clases de Diseño
- Comportamiento de Clases de Diseño

**Participantes**

- Equipo de Arquitectura
- Equipo del Proyecto

PDF de METRICA 3 => Leerlo



```
<?xml version="1.0" encoding="UTF-8"?>
<org.eclipse.epf.uma:TaskDescription xmi:version="2.0"
  xmlns:xmi="http://www.omg.org/XMI"
  xmlns:org.eclipse.epf.uma="http://www.eclipse.org/epf/uma/1.0.4/uma.ecore"
  xmlns:epf="http://www.eclipse.org/epf" epf:version="1.2.0" xmi:id="-
vF3xMvNwJZwzNdl7nMXUCA"
...
  <mainDescription>&lt;p>&#xD;
  Identificador de tarea: 55&lt;br />&#xD;
  Origen de la tarea: &lt;em>Métrica 3.&lt;/em> Tarea &lt;em>DSI 6.1: Diseño
del modelo físico de datos.&lt;/em>&#xD;
&lt;/p>&#xD;
&lt;p align="center"&lt;br />&#xD;
  En esta tarea se determina cómo se van a convertir las clases en tablas,
considerando las relaciones existentes entre&#xD; ellas y los identificadores,
definiendo sus claves primarias, ajenas, alternativas u otros medios de acceso
en general.&#xD;
...
  así como para hacer una estimación del&#xD; espacio de
almacenamiento.&#xD;
&lt;/p>&lt;/mainDescription>
  <sections xmi:id="_9oDDkHMaEdyRedV3y8FdVg" name="El Administrador
de Bases de Datos y el Equipo de Desarrollo analizan el gestor de bases de
datos o el sistema de ficheros."
  guid="_9oDDkHMaEdyRedV3y8FdVg"/>
  <sections xmi:id="_D_cDcHMbEdyRedV3y8FdVg" name="El Equipo de
Desarrollo y el Equipo de Arquitectura analizan las estimaciones de utilización y
volumen de las ocurrencias de cada clase del modelo de clases."
  guid="_D_cDcHMbEdyRedV3y8FdVg"/>
  ...
  <purpose>&lt;p align="center"&lt;br />&#xD;
  El objetivo de esta tarea es realizar el diseño del modelo&nbsp;&nbsp;&nbsp;lógico de
datos a partir del modelo de clases.&#xD;
&lt;/p>&lt;/purpose>
</org.eclipse.epf.uma:TaskDescription>
```

Especificación en XML de la misma tarea de METRICA 3 usando el metamodelo SPEM =>

ahora se puede realizar cualquier tipo de procesamiento automático que interese



**Tarea: DSI 07.1: Diseño del modelo lógico de datos**

En esta tarea se realiza el diseño del modelo lógico de datos.  
Disciplinas: Diseño

Expandir todas las secciones    Contraer todas las secciones

**Objetivo**  
El objetivo de esta tarea es realizar el diseño del modelo lógico de datos a partir del modelo de clases.  
Volver al inicio

**Relaciones**

<b>Roles</b>	Realizador principal: • Equipo de Desarrollo	Otras actividades adicionales que realiza: • Administrador de Bases de Datos • Equipo de Arquitectura
<b>Entradas</b>	Obligatoria: • Características específicas del SGBD o sistema de ficheros que se utilizará • PPL: Plan de migración y de carga inicial de datos • SDT: Especificaciones de migración y carga inicial de datos • SDT: Modelo de clases de diseño	Opcional: • Ninguno
<b>Salidas</b>	• SDT: Modelo lógico de datos	

Volver al inicio

**Descripción principal**  
Identificador de tarea: 55  
Origen de la tarea: Métrica 3. Tarea DSI 6.1: Diseño del modelo físico de datos.  
En esta tarea se determina cómo se van a convertir las clases en tablas, considerando las relaciones



Los procesos de diferentes proyectos tienden a seguir patrones comunes.

Es necesario intentar capturar estos aspectos comunes en una **representación del proceso**, la cuál describe estas características comunes y fomenta la homogeneidad.

**Modelo de Procesos (MP):** representación abstracta de una familia de procesos expresada en una adecuada notación de modelado de procesos (formalismo).



## Un Modelo de PS

Es una abstracción o representación (textual, gráfica o formal) en la que se capturan los aspectos más importantes de un PS

Es aplicable a un proyecto particular o a una familia de proyectos

Es una representación descriptiva de:

- las actividades,
- los recursos,
- los productos,
- los actores y
- las reglas que el proceso requiere para alcanzar sus objetivos.

Está basada en un lenguaje de modelado (metamodelo+sintaxis)

17

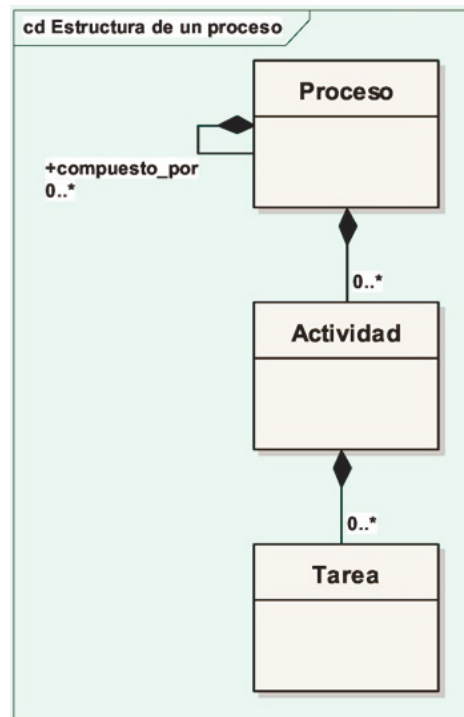


Según los estándares ISO, un PS tiene una **estructura jerárquica** con varios niveles de agregación:

- Subprocesos (opcionales)
- Actividades
- Tareas

Los procesos, subprocessos y actividades tienen asociado un flujo de trabajo.

Las tareas son las unidades básicas de trabajo (atómicas).



18



La disponibilidad de un MP (*computerizado*) proporciona capacidades para:

- Facilitar la comprensión y comunicación humana.
- Facilitar la reutilización.
- Dar soporte a la mejora de procesos.
- Dar soporte a la gestión de procesos.
- Guiar la automatización de procesos.
- Dar soporte para la ejecución automática.

Para poder ofrecer lo anterior, los MPS deben:

- Representar la forma en que el proceso es (o debería ser) realizado;
- Proveer un marco de trabajo flexible y fácil de comprender, aunque con potencia para representar y mejorar el proceso; y
- Permitir refinar hasta llegar al nivel de detalle que sea necesario.

19



Expresan un punto de interés particular en vez del MP completo (similar a vistas en BD):

- Sub-modelos (en modelado *bottom-up*).
- Modelos parciales (en modelado *top-down*).

Las más habituales son:

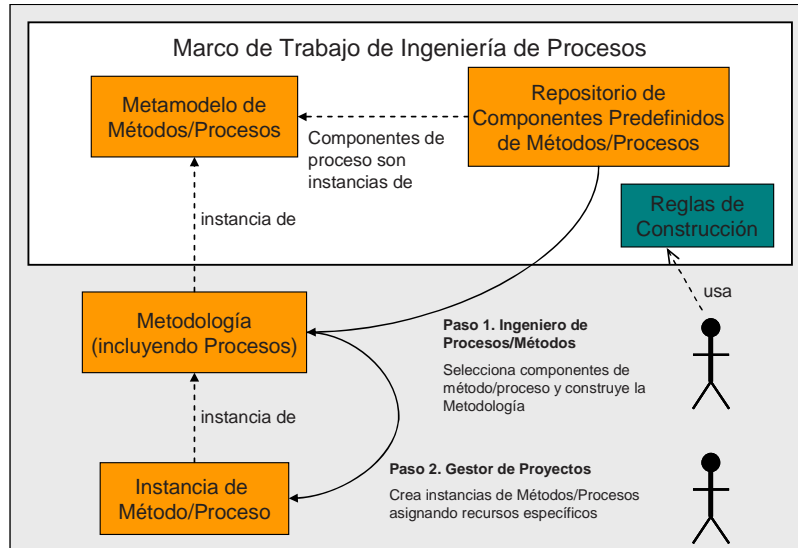
- **De actividades:** tipos, estructura y propiedades de las actividades y sus relaciones
- **De productos:** tipos, estructura y propiedades de los ítems software de un proceso;
- **De recursos:** describe los recursos que se necesitan o se suministran a los procesos;
- **De roles:** describe un peculiar conjunto de recursos, como son las habilidades que los desarrolladores suministran y las responsabilidades que aceptan.

Nos son disjuntas: una vista no puede ser definida sin usar conceptos de otras.

20

La **Ingeniería de Procesos** trata de aplicar a los procesos maneras y técnicas que antes han demostrado su utilidad en los productos (software).

Sinónimo: Ingeniería de Métodos



- Introducción
  - Procesos Software
  - Modelos de Procesos Software
  - Principios de la Ingeniería de Procesos Software
- **Fundamentos de SPEM**
  - **Características básicas**
  - **Metamodelo vs perfil UML**
  - **Usos**
- Elementos de SPEM
  - Organización
  - Tareas
  - Roles
  - Productos de Trabajo
  - Guías
  - Categorías.
- Composición de Procesos.
  - Estructura
  - Actividades
  - Fases
  - Iteraciones
  - Hitos
  - Patrones de Proceso
  - Procesos para Despliegue
- EPF Composer
  - Funcionalidad básica
  - Demostración
  - Características avanzadas

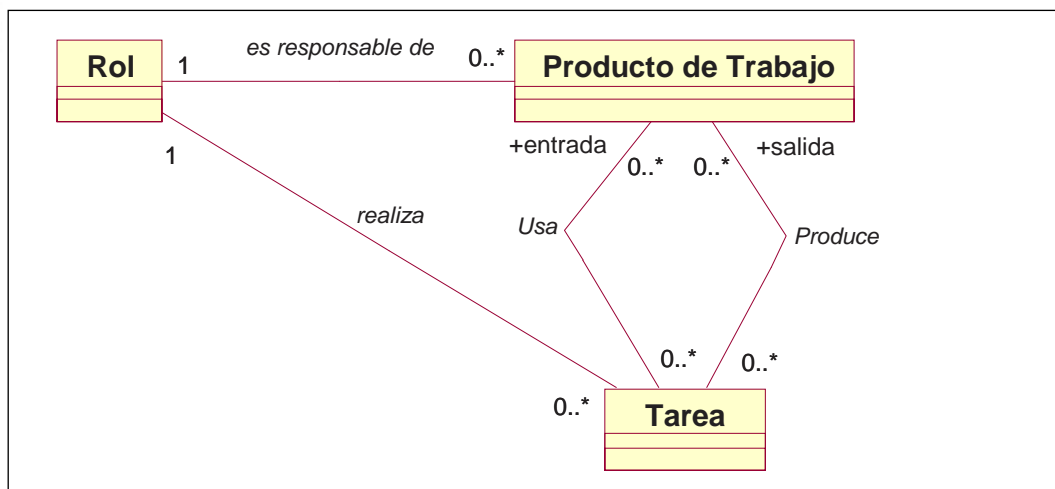
# SPEM 2

## Software & Systems Process Engineering Metamodel Specification, v2.0

Metamodelo para modelos de procesos de ingeniería del software y de ingeniería de sistemas

- Se describe de dos maneras:
  - como un metamodelo MOF-compliant, y
  - como un perfil UML 2.

Idea básica de proceso:





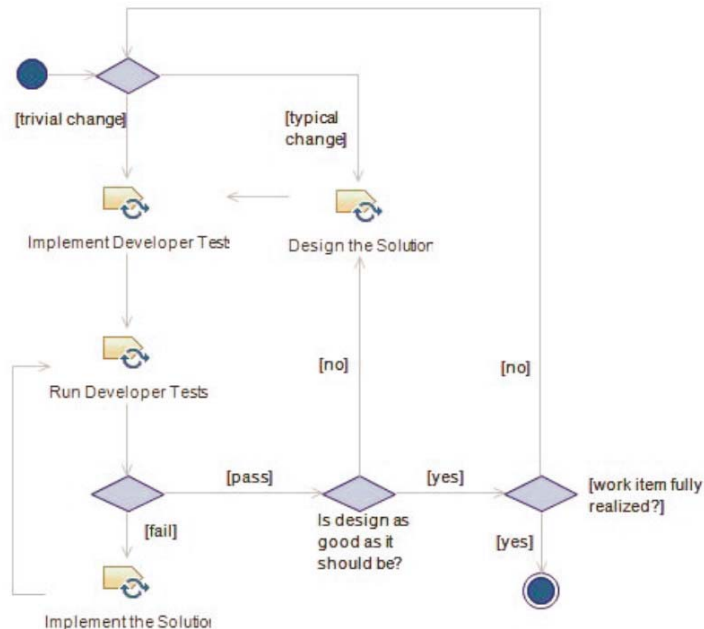
# Las descripciones de trabajo tienen asociada una estructura de desglose de trabajo

- Metrica 3 (Versión Estructurado)
  - Proceso ASI (Versión Estructurado)
    - ASI 1: Definición del Sistema (Versión Estructurado)
      - ASI 1.1: Determinación del Alcance del Sistema (Versión Estructurado)
        - ASI 1.1.1: Identificación del Entorno Tecnológico (Versión Estructurado)
        - ASI 1.1.3: Especificación de Estándares y Normas (Versión Estructurado)
        - ASI 1.1.4: Identificación de los Usuarios Participantes y Finales (Versión Estructurado)
      - ASI 2: Establecimiento de Requisitos (Versión Estructurado)
        - ASI 2.1: Obtención de Requisitos (Versión Estructurado)
        - ASI 2.2: Especificación de Casos de Uso (Versión Estructurado)
        - ASI 2.3: Análisis de Requisitos
        - ASI 2.4: Validación de Requisitos
      - ASI 3: Identificación de Subistemas de Análisis (Versión Estructurado)
        - ASI 3.1: Determinación de Subistemas de Análisis (Versión Estructurado)
        - ASI 3.2: Integración de Subistemas de Análisis (Versión Estructurado)
      - ASI 6: Elaboración del Modelo de Datos
        - ASI 6.1: Elaboración del Modelo Conceptual de Datos
        - ASI 6.2: Elaboración del Modelo Lógico de Datos
        - ASI 6.3: Normalización del Modelo Lógico de Datos
        - ASI 6.4: Especificación de Necesidades de Migración y Carga de Datos
      - ASI 7: Elaboración del Modelo de Procesos
        - ASI 7.1: Obtención del Modelo de Procesos del Sistema
        - ASI 7.2: Especificación de Interfaces con Otros Sistemas
      - ASI 8: Definición de Interfaces de Usuario (Versión Estructurado)
        - ASI 8.1: Especificación de Principios Generales de la Interfaz
        - ASI 8.2: Identificación de Perfiles y Diálogos
        - ASI 8.3: Especificación de Formatos Individuales de la Interfaz de Pantalla (Versión Estructurado)
        - ASI 8.4: Especificación del Comportamiento Dinámico de la Interfaz (Versión Estructurado)
        - ASI 8.5: Especificación de Formatos de Impresión (Versión Estructurado)
      - ASI 9: Análisis de Consistencia y Especificación de Requisitos (Versión Estructurado)
        - ASI 9.1: Verificación de los Modelos (Versión Estructurado)
        - ASI 9.2: Análisis de Consistencia Entre Modelos (Versión Estructurado)
        - ASI 9.3: Validación de los Modelos (Versión Estructurado)
        - ASI 9.4: Elaboración de la Especificación de Requisitos Software (ERS) (Versión Estructurada)

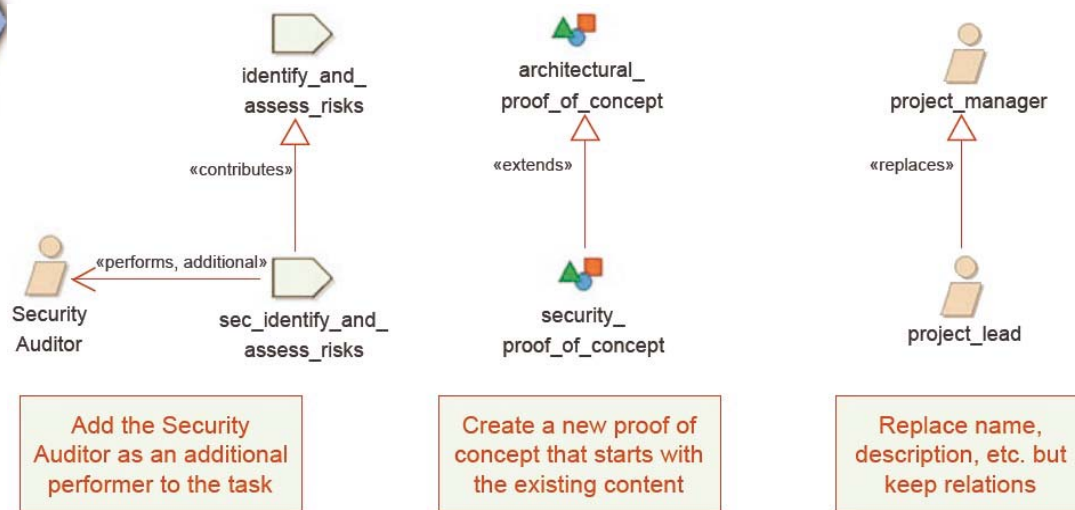
**Work Breakdown Structure (WBS)**



# Las descripciones de trabajo también pueden tener asociado un flujo de trabajo



## Existen mecanismos de **variabilidad** para modificar un proceso base



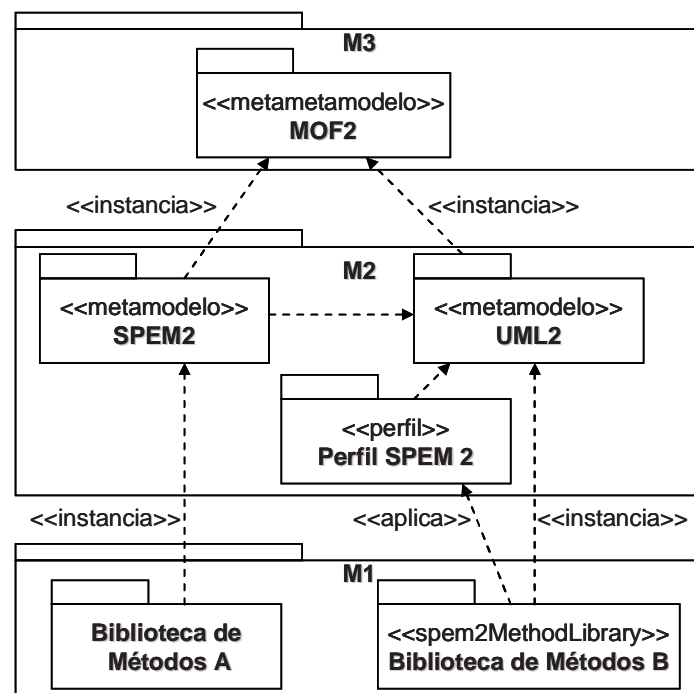
- SPEM 2 sirve para definir procesos de desarrollo de software y sistemas y sus componentes.
- =>
  - Su **alcance** se limita a los elementos mínimos necesarios para definir dichos procesos sin añadir características específicas de un dominio o disciplina particular.
  - Pero sirve para métodos y procesos de **diferentes** estilos, culturas, niveles de formalismo, o modelos de ciclos de vida.
  - No es un lenguaje de modelado de procesos en general.
  - No provee conceptos para **modelado del comportamiento**, pero incluye mecanismos para encajar el elegido (diagramas de actividad de UML 2, BPMN/BPDM, ..).



- **Metamodelo MOF-compliant**
  - Define todas las estructuras y reglas de estructuración para representar contenidos de métodos y procesos.
  - Es completo en sí mismo.
  - Está definido como un metamodelo del nivel M2 de MOF.
  - Reutiliza algunas clases de UML 2.
  - Define la notación de diagramas de proceso específicos.
- **Perfil de UML 2.**
  - Define un conjunto de estereotipos UML 2 que permiten representar métodos y procesos usando UML 2.
  - La definición sólo abarca la presentación, ya que las definiciones semánticas y restricciones están en el metamodelo.

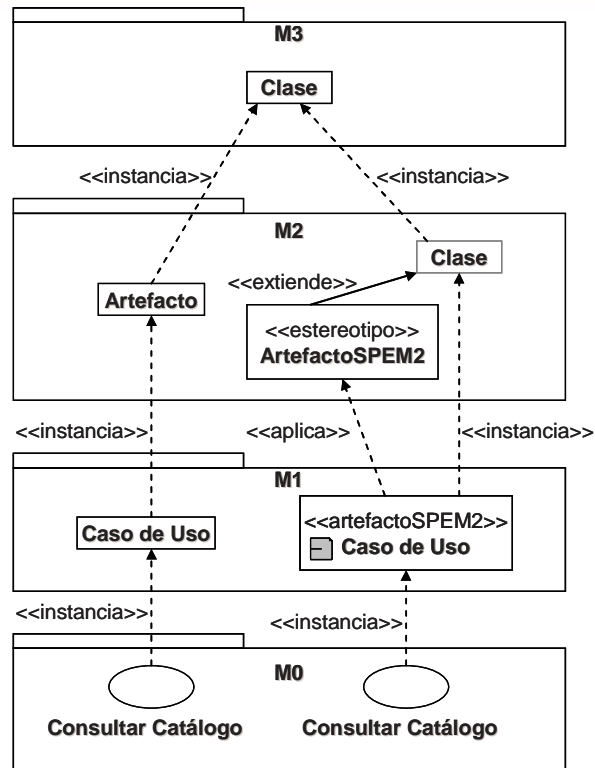


- **Capas de modelado**





- Ejemplos de instancias



### Problemas relacionados con los procesos, que enfrentan las organizaciones que desarrollan software:

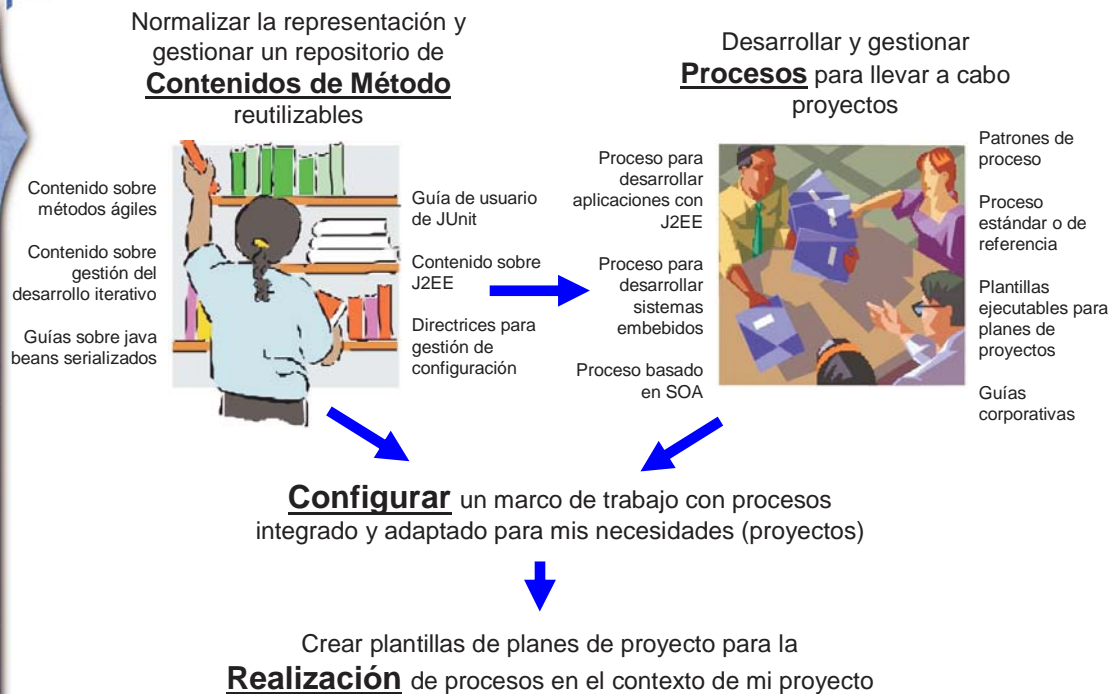
- Miembros de los equipos no tienen acceso fácil y centralizado a la información de procesos que necesitan
  - Diferentes desarrolladores manejan versiones diferentes fuentes o versiones de la misma información.
- Es difícil combinar e integrar informaciones y procesos que están en formatos propietarios diferentes
  - Cada libro, manual , herramienta utiliza un lenguaje y estilo diferente.
- Es duro definir una aproximación de desarrollo organizada y sistemática que se adapte a las necesidades
  - Cultura, prácticas establecidas, requisitos de certificación, legales, etc.



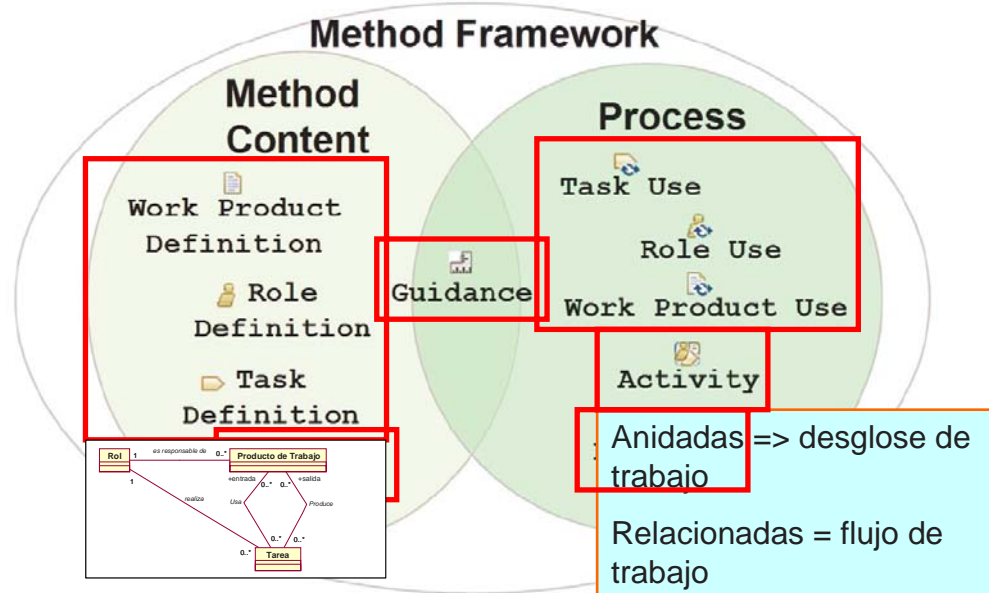
- SPEM es un metamodelo para ingeniería de procesos y un marco de trabajo conceptual que provee los conceptos necesarios para **modelar, documentar, presentar, publicar, gestionar, intercambiar y realizar métodos y procesos software.**
- Está destinado a ingenieros de procesos, jefes de proyectos, gestores de proyectos y programas que son **responsables de mantener e implementar procesos** para sus organizaciones o para proyectos concretos.



### Marco de Trabajo General



Separación clara entre la definición de contenidos de método y su aplicación en procesos.



Conceptos de contenido de método vs proceso

Mantenimiento consistente de muchos procesos alternativos.

- Para ello, SPEM incluye:
  - Un conjunto extendido de interrelaciones de reutilización y variabilidad con semántica de herencia y orientación a aspectos.
  - Conceptos de patrones de proceso, y
  - Plugins de métodos.
- Esto permite tener diferentes variantes de procesos específicos, basados en los mismos contenidos de método y estructuras de procesos, pero aplicados con diferente detalle y escala.



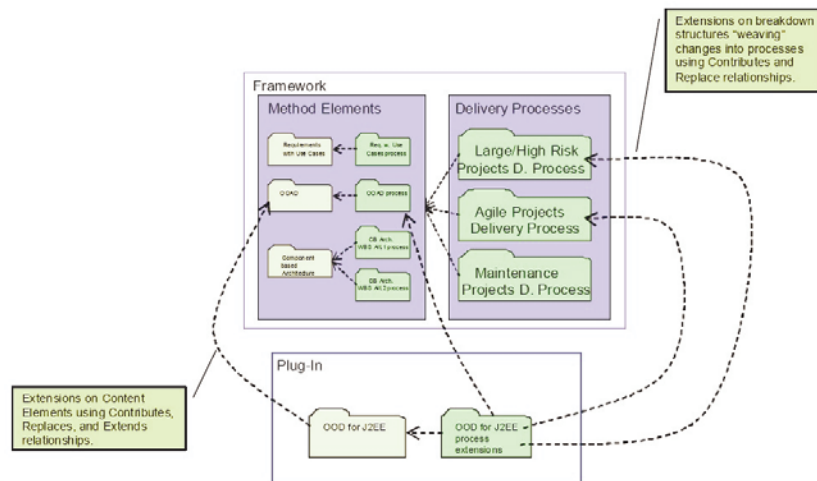
### Muchos ciclos de vida diferentes.

- SPEM permite trabajar con distintos tipos de ciclos de vida del software:
  - Cascada, Iterativo, Incremental, Evolutivo, ..
- Para ello incluye un conjunto de atributos que permiten especificar aspectos temporales para los elementos de proceso que luego pueden ser asociados a los planes de proyectos.
- Ejemplo de atributo para clases de ciclos de vida:
  - Iteración => la ejecución de una o varias descripciones se trabajo se puede repetir más de una vez.



### Variabilidad y extensibilidad.

- Para esto SPEM incluye un mecanismo de **plugins**:
  - **Method plug-ins**: para particularizar y adaptar contenidos de método sin modificar el original.
  - **Process plug-ins**: para procesos, pudiendo añadir o sustituir en el WBS sin afectar al original.





## Patrones de proceso.

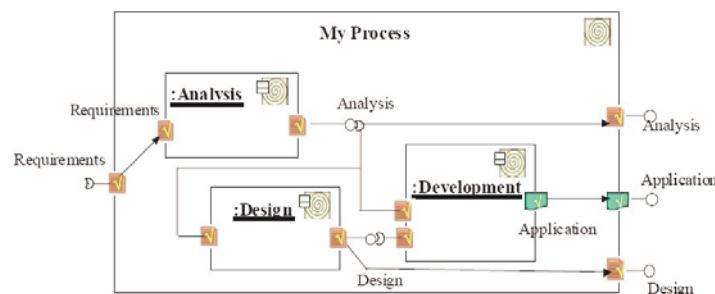
- Son bloques (trozos de proceso) reutilizables para crear nuevos procesos.
- La selección y aplicación de un patrón de proceso puede ser hecha de dos formas:
  - Puede ser **copiado y modificado**, permitiendo individualizar el contenido del patrón según las necesidades de cada momento.
  - Puede ser aplicado por medio del mecanismo de **Actividad en Uso**, que es una forma avanzada de reutilizar estructuras de proceso.
- Una Actividad en Uso define tipos de interrelaciones para que cuando el patrón esté siendo revisado o modificado, todos los cambios se reflejen automáticamente en todos los procesos en que se aplica el patrón.

39



## Componentes de proceso.

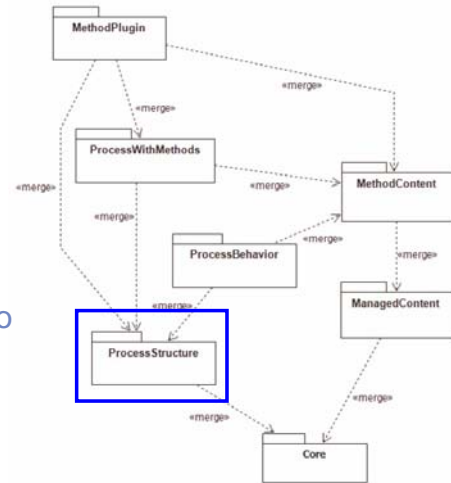
- Piezas de proceso sustituibles y reutilizables basadas en los principios de **encapsulación** y **caja negra**:
  - No se especifica la descripción de trabajo interna del componente.
  - Sólo se especifican los productos de trabajo de entrada y salida que habrá (**puertos** de productos de trabajo).
- Permiten manejar las situaciones en que un proyecto requiere que partes del proceso no sean decididas hasta la ejecución (caso típico: outsourcing).



40

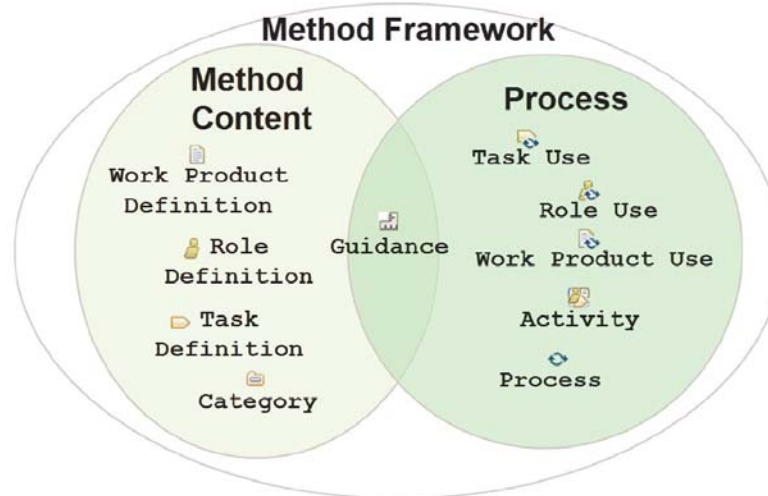
Paquete **Process Structure:**

- Define la base para la creación de modelos de proceso (MP) flexibles y sencillos.
- Define la **estructura de desglose de trabajo** estática mediante anidamiento de **actividades** y dependencias de precedencia entre ellas.
- Dicha estructura también incluye referencias a la lista de **Roles** que realiza cada actividad y a los **Productos de Trabajo** que son entradas y/o salidas.
- Provee capacidades para:
  - **Reutilización** mediante ensamblado de procesos usando conjuntos de actividades enlazadas de forma dinámica.



- Introducción
  - Procesos Software
  - Modelos de Procesos Software
  - Principios de la Ingeniería de Procesos Software
- Fundamentos de SPEM
  - Características básicas
  - Metamodelo vs perfil UML
  - Usos
- Elementos de SPEM
  - **Organización**
  - **Tareas**
  - **Roles**
  - **Productos de Trabajo**
  - **Guías**
  - **Categorías.**
- Composición de Procesos.
  - Estructura
  - Actividades
  - Fases
  - Iteraciones
  - Hitos
  - Patrones de Proceso
  - Procesos para Despliegue
- EPF Composer
  - Funcionalidad básica
  - Demostración
  - Características avanzadas

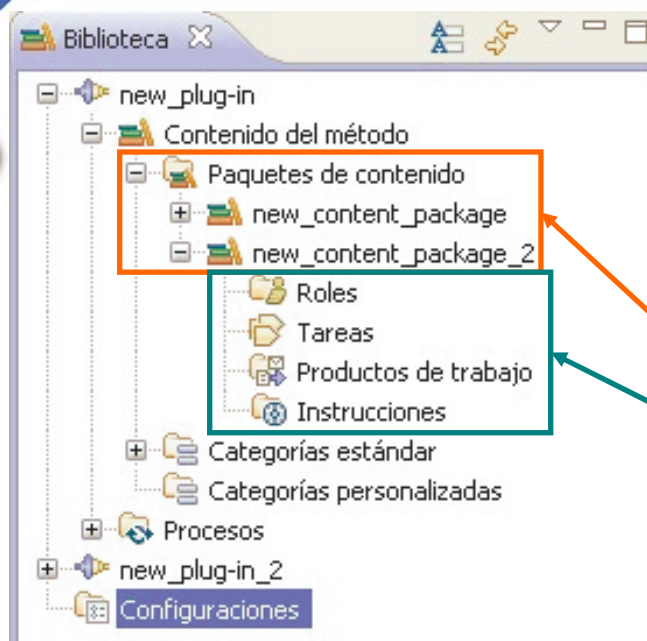
- Con SPEM se distinguen **dos etapas** a la hora de implementar un proceso o metodología
  - Se puebla el **Contenido de Método** con **Elementos de Método**, es decir, los elementos primarios o constructores básicos.
  - Se combinan y reutilizan dichos elementos para ensamblar **Actividades** y **Procesos**.



43

## Contenido de Método

### Organización



El contenido de método puede ser organizado a voluntad del usuario mediante:

- una jerarquía de paquetes de contenido
- cada paquete incluye roles, tareas, productos de trabajo y guías (instrucciones)

44



## Tarea – Task Definition



- Describe una unidad de trabajo asignable y gestionable.
- Cada Tarea se asigna a ciertos Roles.
- Su granularidad es de unas pocas horas a unos pocos días.
- Afecta a un o unos pocos Productos de Trabajo.
- Es la unidad atómica de trabajo para definir procesos.
- Es un Elemento de Método que define el trabajo realizado por roles.
- También es una Definición de Trabajo (en procesos).



## Una Tarea está asociada con:

- 1..\* **Roles** distinguiendo entre:
  - 1 realizador principal obligatorio [responsable]
  - 0..\* realizadores adicionales opcionales
- 1..\* **Productos de Trabajo** como:
  - Entradas obligatorias
  - Entradas opcionales
  - Salidas
- 0..\* **Herramientas** que se recomienda usar.
- 0..\* **Pasos**, que describen de forma secuencial el trabajo a realizar.
- 0..\* **Habilidades** que se requieren habitualmente para llevar a cabo la tarea.



## Existen tres tipos de Productos de Trabajo:



### ▪ Artefacto - Artifact

- De naturaleza tangible (modelo, documento, código, ..)



### ▪ Entregable – Deliverable

- Provee una descripción y definición para empaquetar otros productos de trabajo con fines de entrega a un cliente interno o externo.
- Representa una salida de un proceso que tiene valor para un usuario, cliente u otro participante.
- Esta asociado con 0.\* **componentes de entregable**, que son los productos de trabajo que lo forman.



### ▪ Resultado - Outcome

- Un producto de trabajo de naturaleza intangible (resultado o estado), o
- Que no está formalmente definido.



### • Consideraciones sobre el Cálculo – Estimation Considerations

- Indicaciones para estimar el esfuerzo asociado con cierto trabajo, incluyendo consideraciones sobre cómo hacer la estimación y las métricas a utilizar.



### • Práctica – Practice

- Manera o estrategia predefinida de hacer un trabajo que tiene un impacto positivo sobre la calidad de un producto de trabajo o de un proceso.
- Son ortogonales a los métodos y procesos; de forma que una práctica resume aspectos que pueden impactar en diferentes partes de un método o proceso.
- Ejemplos: “gestionar riesgos”, “desarrollo basado en componentes”...



### • Informe – Report

- Plantilla predefinida de un resultado que se obtiene de forma automática mediante alguna herramienta.



## Categorías Estándar

- **Disciplina – Discipline**
  - Permite categorizar el trabajo (tareas).
  - Una disciplina es una colección de tareas que están relacionadas con un área principal de esfuerzo dentro de un proyecto completo.
  - Suelen estar basadas en una perspectiva tradicional de proyectos en cascada: requisitos, análisis, diseño, construcción, pruebas, mantenimiento, ...
- **Conjunto de Roles – Rol Set**
  - Agrupar roles que tienen algo en común (usan técnicas similares, requieren habilidades parecidas, ...).
    - Ejemplo: Analista englobando a Analista de Sistemas e Ingeniero de Requisitos.

49



La definición de contenidos en SPEM 2 mediante plug-ins permite reutilizar los contenidos definidos en una librería

### Reutilización de plug-ins:

- Al crear un plug-in se puede referenciar a otros plug-ins para reutilizar su contenido en la manera que sea conveniente.
- Reutilización directa del contenido de un plug-in en otro.

50



A veces nos interesa reutilizar el contenido pero con ciertas modificaciones. Para ello existe la **variabilidad de contenido**:

- Permite reutilizar y modificar elementos de método sin modificar directamente el contenido original.
- Permite definir mediante otros elementos las diferencias (adiciones, cambios, omisiones) con el elemento original.
- Afecta a los atributos (como la descripción principal) y a las asociaciones con otros elementos de método.
- Se evalúa en el momento de publicar/examinar una configuración.



SPEM 2.0 contempla 5 tipos de variabilidad:

- No asignada
- Contribuye
- Amplía
- Reemplaza
- Amplía y Sustituye



Un elemento que contribuye añade sus atributos y sus relaciones al elemento base sin modificar directamente las propiedades de dicho elemento base.

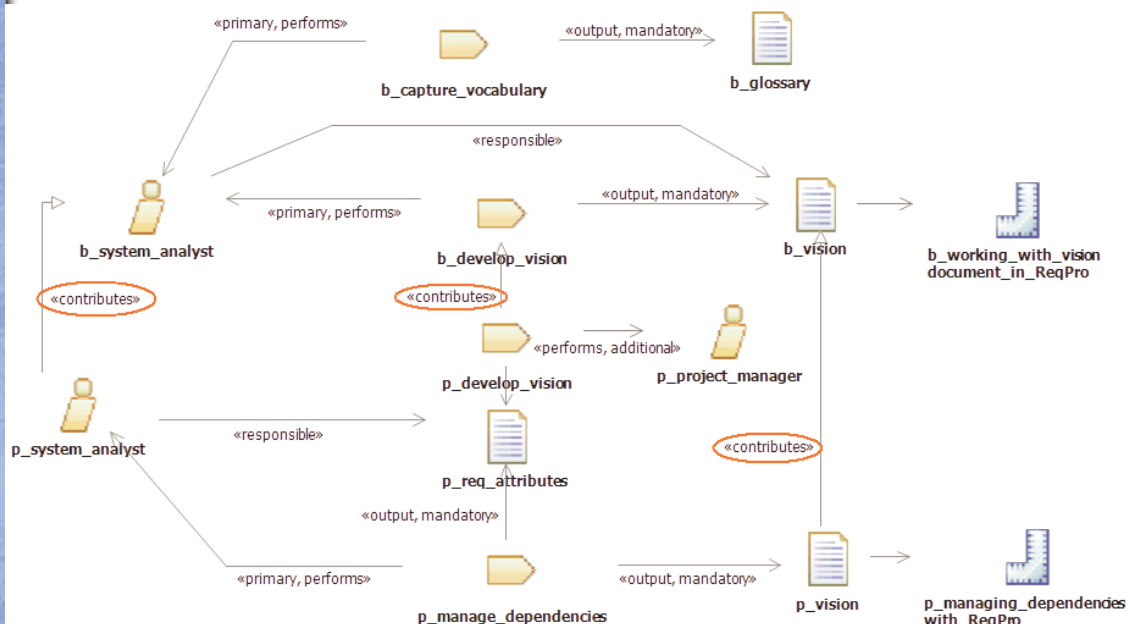
Al publicar los elementos, el elemento base aparece en la web publicada combinado con los atributos y relaciones del elemento que contribuye, mientras que éste último no se publica.

Reglas de contribución:

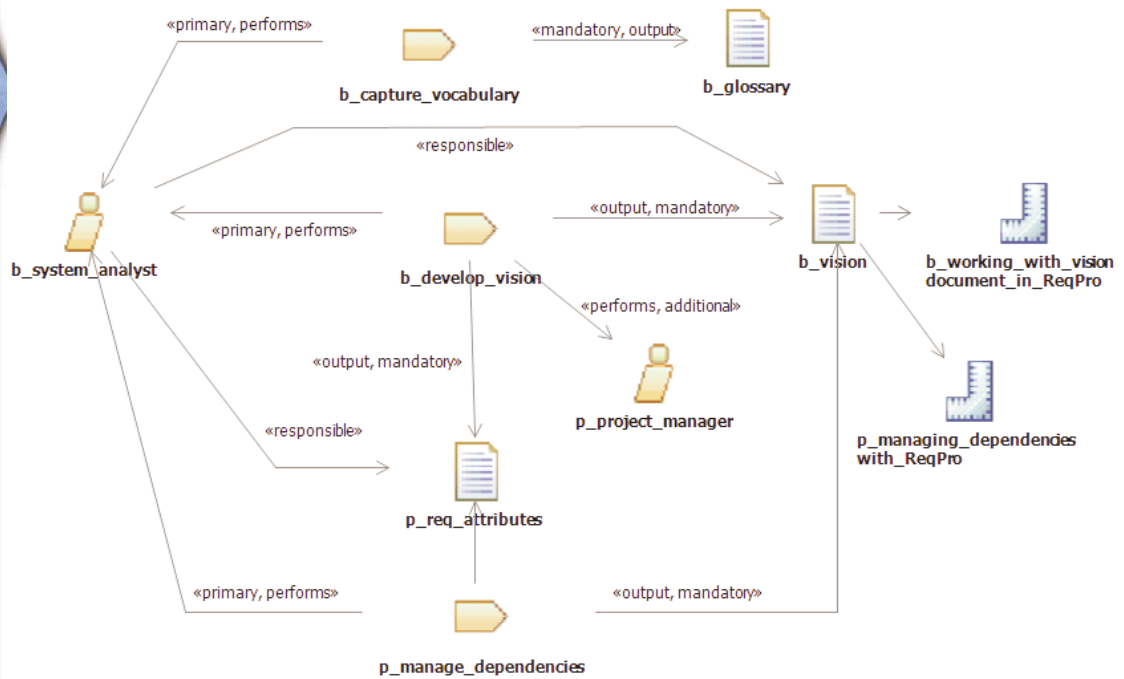
- Los atributos (campos de texto) del elemento que contribuye se concatenan al final de los respectivos campos en el elemento base.
- Las asociaciones de entrada y de salida "a muchos" del elemento que contribuye se añaden al elemento base.
- En las asociaciones de entrada y de salida "a 1" (como la relación "realizador principal" entre una tarea y un rol) definidas, la relación del elemento que contribuye se ignora si la asociación existe en el elemento base.
- Un elemento base puede recibir varias contribuciones.
- La contribución es transitiva.



Ejemplo de contribución:



Resultado en la publicación:



•Introducción

- Procesos Software
- Modelos de Procesos Software
- Principios de la Ingeniería de Procesos Software

•Fundamentos de SPEM

- Características básicas
- Metamodelo vs perfil UML
- Usos

•Elementos de SPEM

- Organización
- Tareas
- Roles
- Productos de Trabajo
- Guías
- Categorías.

•Composición de Procesos.

- Estructura
- Actividades
- Fases
- Iteraciones
- Hitos
- Patrones de Proceso
- Procesos para Despliegue

•EPF Composer

- Funcionalidad básica
- Demostración
- Características avanzadas



## Elemento de Desglose – Breakdown Element

- Es una generalización abstracta para cualquier tipo de elemento que aparece en un proceso y es parte de una estructura de desglose.
- Tienen tres propiedades importantes:
  - Admite **Varias Apariciones** (Has Multiple Occurrences): al realizar el proceso puede haber mas de una instancia del elemento.
  - Es **Opcional** (Is Optional): no es obligatoria su inclusión cuando se lleva a cabo el proyecto.
  - **Planeado** (Is Planned): El elemento es incluido al generar los planes de proyecto que se exportan a las herramientas de gestión de proyectos.

57



## Elemento de Desglose de Trabajo - Work Breakdown Element

- Es un Elemento de Desglose que representa Trabajo.
- Existen dos tipos:
  - Actividad
  - Hito
- Propiedades:
  - Se **Puede Repetir** (Is Repeatable): habrá varias iteraciones o repeticiones.
  - **Continuo** (Is Ongoing): es un trabajo sin duración fija o estado final. Ejemplo: trabajo de un gestor de proyecto que 1 hora al día se dedica a revisar el estado de avance de las tareas.
  - **Condicionado por Sucesos** (Is Event Driven): su inicio no está determinado por eventos normales (cuando acaba el trabajo que lo precede, o cuando se concluye algún producto de trabajo), sino por otro evento especial.

58



- El **Flujo** entre los Elementos de Desglose de Trabajo se representa por medio de **Secuencias de Trabajo**.
- Cada Secuencia de Trabajo conecta dos Elementos de Desglose de Trabajo: **predecesor** y **sucesor**.
- Cada Secuencia de Trabajo, entre un predecesor P y un sucesor S, es de una de las siguientes clases:
  - Acabar para Empezar: S no puede empezar hasta que no concluye P.
  - Acabar para Acabar: S no puedo acabar mientras no esté acabado P.
  - Empezar para Empezar: S no puede comenzar hasta que no lo ha hecho P.
  - Empezar para Acabar: S no puede concluir hasta que no se inicia P.
    - Se emplea en Just-in-Time.

NOTA: Son los mismos tipos de precedencias usados en gestión de proyectos (MS Project)

59

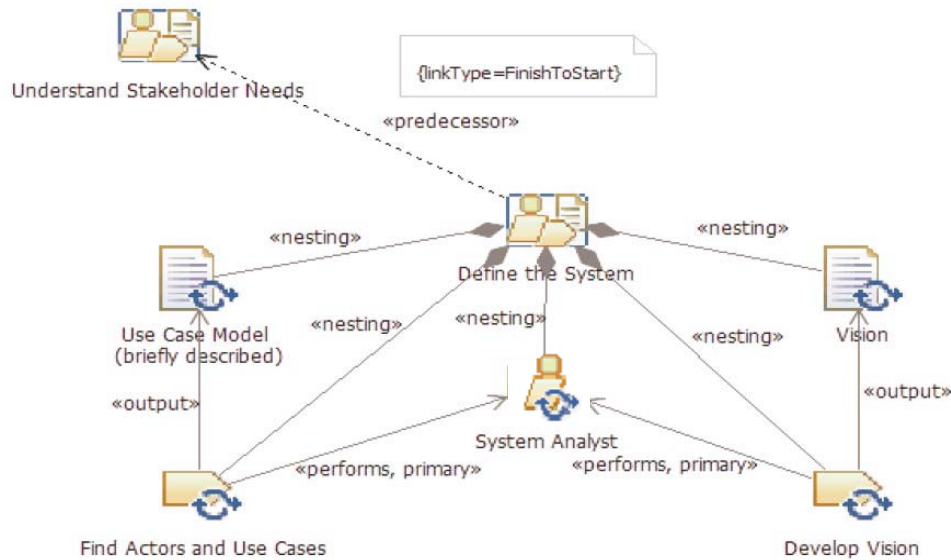


- Estructura (**WBS**) asociada a un Elemento de Desglose de Trabajo

Nombre de presentación	Índice	Predecesores	Tipo
[-] Fase: CENTRAL	0		Patrón de posibilidad
[-] ASI 06: DISEÑO DE CLASES	1		Patrón de posibilidad
[-] ASI 06.1: Selección de casos de uso	2		Descriptor de tareas
[-] ASI 06.2: Identificación de clases asociadas a los casos de uso	3	2	Descriptor de tareas
[-] ASI 06.3: Descripción de la interacción entre objetos	4	3	Descriptor de tareas
[-] ASI 07: DISEÑO DE LA ARQUITECTURA DE MÓDULOS DEL SISTEMA	5	1	Patrón de posibilidad
[-] ASI 07.1: Identificación de responsabilidades y atributos	6		Descriptor de tareas
[-] ASI 07.2: Identificación de asociaciones y agregaciones	7	6	Descriptor de tareas
[-] ASI 07.3: Identificación de generalizaciones	8	7	Descriptor de tareas
[-] ASI 08: ESPECIFICACIÓN DE LA INTERFAZ DE USUARIO	9	5	Patrón de posibilidad
[-] ASI 08.1: Identificación de perfiles y diálogos	10		Descriptor de tareas
[-] ASI 08.2: Especificación de formatos individuales de la interfaz de pantalla	11	10	Descriptor de tareas
[-] ASI 08.3: Especificación del comportamiento dinámico de la interfaz	12	11	Descriptor de tareas
[-] ASI 08.4: Especificación de formatos de impresión	13	12	Descriptor de tareas
[-] ASI 09: ESPECIFICACIÓN DE ELEMENTOS DE SEGURIDAD	14	9	Patrón de posibilidad
[-] ASI 09.1: Identificación de elementos de seguridad	15		Descriptor de tareas
[-] ASI 10: APROBACIÓN E INTEGRACIÓN DE CLASES DE ANÁLISIS	16	14	Patrón de posibilidad
[-] ASI 10.1: Aprobación del modelo de clases de análisis	17		Descriptor de tareas
[-] ASI 10.2: Integración de clases de análisis con las de iteraciones anteriores	18	17	Descriptor de tareas
[-] DSI 03: DISEÑO DE CASOS DE USO REALES	19	16	Patrón de posibilidad
[-] DSI 03.1: Identificación de clases de diseño asociadas a los casos de uso seleccionados	20		Descriptor de tareas
[-] DSI 03.2: Diseño de la realización de casos de uso	21	20	Descriptor de tareas
[-] DSI 04: DISEÑO DE CLASES	22	19	Patrón de posibilidad
[-] DSI 04.1: Identificación de clases adicionales	23		Descriptor de tareas
[-] DSI 04.2: Diseño de asociaciones y agregaciones	24	23	Descriptor de tareas
[-] DSI 04.3: Identificación de atributos de las clases	25	24	Descriptor de tareas
[-] DSI 04.4: Identificación de operaciones de las clases	26	25	Descriptor de tareas

60

- Ejemplo de una actividad con sus asociaciones



### Fase - Phase



- Representa un periodo de tiempo que es significativo para un proyecto, y que acaba de alguna de las siguientes maneras:
  - Con un punto de control de gestión importante,
  - un hito, o
  - Un conjunto de entregables concluidos.
- En la práctica en SPEM es una Actividad que cumple  
Es Repetible = Falso



### Iteración - Iteration

- Representa un conjunto de actividades anidadas que se repiten más de una vez.
- Permite organizar ciclos repetitivos de trabajo.
- En la práctica en SPEM es una Actividad que cumple  
Es Repetible = Cierto



### Hito - Milestone

- Representa un evento significativo para el desarrollo de un proyecto:
  - Decisión importante
  - Conclusión de un entregable
  - Conclusión de una fase, ...
- Es un Elemento de Desglose de Trabajo, por tanto
  - Aparece en la estructura de desglose de trabajo, y
  - Puede tener relaciones de precedencia.



## Patrón de Proceso – Capability Pattern



- También llamado Patrón de Posibilidad
- Es un “fragmento de proceso” que describe un grupo de actividades reutilizable como solución a algún tipo de problema o situación habitual.
- Casos típicos de aplicación de Patrones de Proceso:
  - Servir como bloques para construir Procesos para Despliegue o Patrones de Proceso más complejos.
  - Ayudar a la ejecución de proyectos que **no** siguen un proceso bien definido, sino que trabajan en base a fragmentos de proceso (buenas prácticas) de una manera flexible (métodos ágiles).
  - En formación, para describir el conocimiento de una cierta área clave, buena práctica, disciplina, etc.

65



## Proceso para Despliegue – Delivery Process



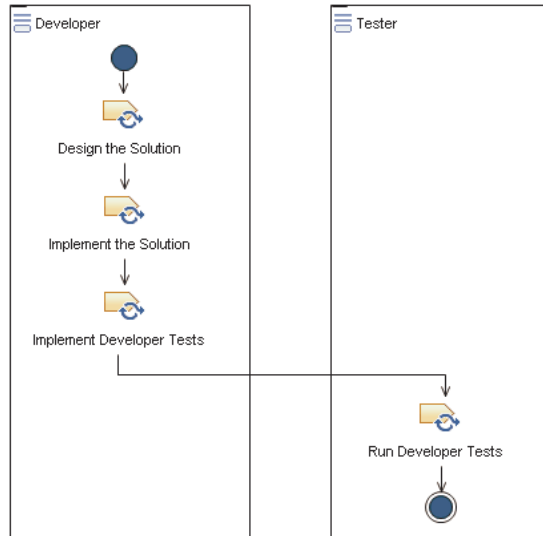
- Describe una aproximación completa e integrada para realizar un tipo específico de proyecto.
- Cubre un ciclo de vida de desarrollo o mantenimiento completo.
- Es usado como plantilla para planificar y ejecutar los proyectos.
- En un Proceso para Despliegue se ensamblan Patrones de Proceso y Elementos en Uso (Tareas, Roles, Productos de Trabajo).

66

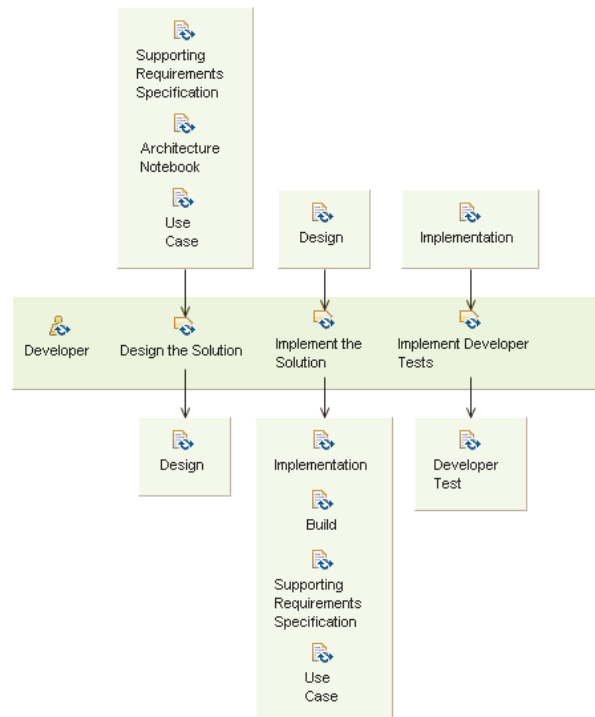
**Diagramas de actividad:** Estos diagramas muestran el flujo entre los elementos hijos de un proceso, fase, actividad o iteración. Suelen utilizarse para mostrar la secuencia de las tareas de una actividad, la secuencia de las actividades de un proceso, etc.

La precedencia de las tareas puede indicarse directamente en el diagrama de actividad.

Los cambios en la precedencia realizados en el diagrama son equivalentes a realizarlos en la estructura de desglose de trabajo.

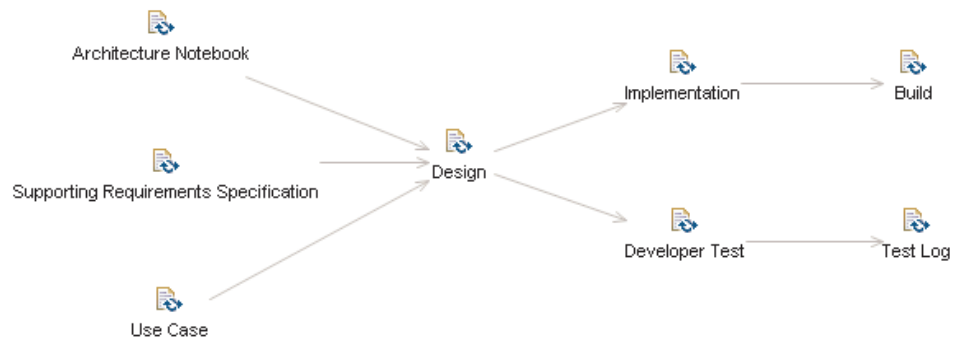


**Diagrama de detalles de la actividad:** Este diagrama muestra las tareas organizadas según el rol responsable que las realiza, y además, muestra las entradas y salidas de cada tarea.



## Diagrama de dependencia del producto de trabajo:

Estos diagramas se utilizan para mostrar la trazabilidad de los productos de una actividad de un proceso. También muestra cómo uno o varios productos se utilizan en la creación de otros.



### •Introducción

- Procesos Software
- Modelos de Procesos Software
- Principios de la Ingeniería de Procesos Software

### •Fundamentos de SPEM

- Características básicas
- Metamodelo vs perfil UML
- Usos

### •Elementos de SPEM

- Organización
- Tareas
- Roles
- Productos de Trabajo
- Guías
- Categorías.

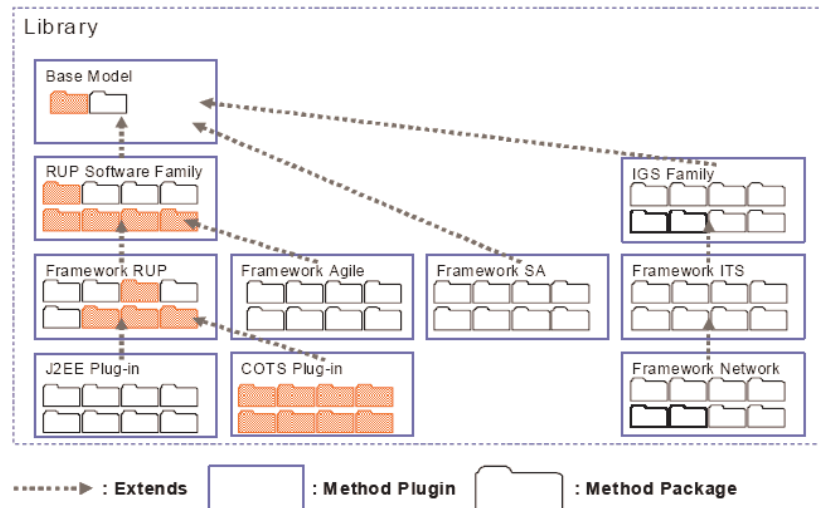
### •Composición de Procesos.

- Estructura
- Actividades
- Fases
- Iteraciones
- Hitos
- Patrones de Proceso
- Procesos para Despliegue

### •EPF Composer

- **Funcionalidad básica**
- **Demostración**
- **Características avanzadas**

Una configuración de método es una selección de contenidos de los plug-ins de una librería, de forma que se limita la vista de la librería al subconjunto seleccionado.



Se pueden definir varias vistas para una misma configuración.

El contenido de una vista se define mediante categorías. Dos opciones:

- Una vista se corresponde con una categoría estándar.
- Una vista se corresponde con una categoría personalizada.

La segunda opción es más potente, ya que en una categoría personalizada podemos incluir todos los elementos de todos los tipos que queramos.

Hay que tener en cuenta que los elementos contenidos en la categoría que se elegirá como vista estén incluidos en la configuración.



## Preferencias generales de publicación:

Ventana → Preferencias → Método → Publicar/Examinar

Recomendaciones:

Marcar la casilla “Incluir el contenido del método en las páginas del descriptor”.

*Diagramas de roles:* Ajustar el espaciado horizontal y vertical y el número de líneas en función de la longitud de los nombres de las tareas y los productos.

*Diagramas de actividad:* No marcar las dos casillas, a no ser que queramos que se publiquen los diagramas por defecto (que no hayamos creado manualmente con el editor).

73

## Creación de Métodos

### Pasos recomendados

Procedimiento recomendado para crear métodos desde cero, atendiendo a su mantenibilidad, reusabilidad y escalabilidad.

1. Crear una **biblioteca de métodos**.
2. Crear los **plugins de método** necesarios.
3. Crear los **elementos de método** organizándolos en **paquetes de contenido**. Es recomendable seguir el siguiente orden:
  1. Crear las **guías**, ya que pueden ser referenciadas desde cualquier otro elemento de método.
  2. Crear los **productos de trabajo**. Al crear cada producto, asociarle las guías convenientes.
  3. Crear los **roles**. Al crear cada rol, indicar los productos de trabajo de los que son responsables y asociarle las guías convenientes.
  4. Crear las **tareas**. Al crear cada tarea, indicar los roles que participan en la tarea, los productos de trabajo que son entrada o salida de la tarea y las guías convenientes.

74



- Información de los Procesos que se exporta a MS Project:
  - **EDT:** Estructura de tareas y actividades (tareas resumen).
  - **Relaciones de precedencia** entre elementos,
  - **Tipo de dependencia** (finalización a inicio, finalización a finalización, etc.)
  - **Roles:** se exportan como recursos y se conectan con las tareas o tareas resumen correspondientes.
- Se crean unas fechas de inicio y finalización por defecto que se deben modificar.
- Hay que comprobar en la pestaña **Estructura de desglose de trabajo** que los elementos que queremos exportar tienen activada la casilla **Planeado**.



- ¿Cómo generar distintas vistas de una metodología?
  - Utilizando **configuraciones**.
  - En cada configuración se incluyen sólo los elementos que se consideren necesarios.
  - Por ejemplo, se podría crear una vista para desarrolladores en la que no aparezcan actividades, tareas, productos, etc. que no tengan relación directa y estricta con el desarrollo del código (como por ejemplo gestión del proyecto, implantación,...)
  - Hay que tener en cuenta que cuando se publica contenido, lo que se publica es una configuración. Por tanto, si queremos publicar varias vistas habrá que hacerlo por separado, dando como resultado varios sitios web distintos.



- ¿Cómo preparar versiones reducidas para entregar a terceros?
  - Utilizando **configuraciones** y la capacidad de **exportarlas**.
  - Se incluyen en la configuración todos los elementos que deban entregarse a terceros.
  - Se exporta la configuración y posteriormente se importa en una nueva biblioteca de métodos vacía que únicamente contendrá la configuración exportada y todos los elementos seleccionados en dicha configuración.
  - De esta forma, se puede entregar a terceros una biblioteca de métodos que únicamente contiene los elementos que se desean entregar.
  - Hay que tener en cuenta que las referencias a los elementos que no se incluyen en la configuración exportada se pierden.



- ¿Cómo incorporar archivos y enlaces en elementos o procesos?
  - Se pueden incorporar archivos o enlaces externos en la página de cualquier elemento o proceso publicado.
  - Se añaden desde la vista de edición del elemento o proceso, en cualquiera de los campos de texto que permitan formato RTF (normalmente en la Descripción principal).
  - Para ello, se abre el editor RTF pulsando el icono  y se pulsa el icono  que aparece en la barra superior del editor de RTF.