



php | Motores de Workflow

Más allá de las Aplicaciones CRUD

Carlos Buenosvinos (carlos.buenosvinos@gmail.com)
Zend PHP Certified – CIO at Latam Training

Carlos Crespo (thecresp@gmail.com)
PHP Consultant at Latam Training



Presentación

▪ Ponente y Ayudante

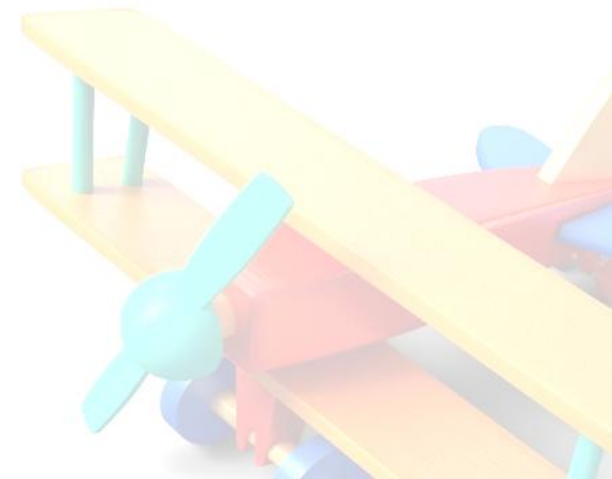
- Carlos Buenosvinos
 - 25 años, Ingeniero Informático Superior por la UPC (Universitat Politècnica de Catalunya)
 - Zend PHP Certified, Ponente en la 1ª phpBarcelona Conference
 - CIO at Latam Training (Consultoría de eRRHH – www.latamtraining.com)
- Carlos Crespo
 - 25 años, Ingeniero Informático Superior por la UPC (Universitat Politècnica de Catalunya)
 - PHP Consultant at Latam Training (Consultoria de eRRHH – www.latamtraining.com)

▪ Objetivos del Taller

- **Conocer** los Principios de un Motor de Workflow
- **Identificar** cuándo es interesante aplicar un Motor de Workflow
- **Conocer** las Soluciones de Motores de Workflow existentes en PHP
- **Practicar** la implementación de un Motor de Workflow en una pequeña Aplicación Web

▪ Metodología del Taller

- Duración: 1 hora y media
 - Ponencia: 30 minutos
 - Parte Práctica: 1 hora
- Guiado a través de Ejercicios Prácticos (Guía Online)
- ¿Qué pasa si tengo un problema o duda?



Conceptos Previos

- **Flujo de Trabajo (Workflow)**

- El **Flujo de trabajo** es el estudio de los aspectos operacionales de una actividad de trabajo: cómo se estructuran las tareas, cómo se realizan, cuál es su orden correlativo, cómo se sincronizan, cómo fluye la información que soporta las tareas y cómo se le hace seguimiento al cumplimiento de las tareas.

- **Proceso de Negocio (Business Process)**

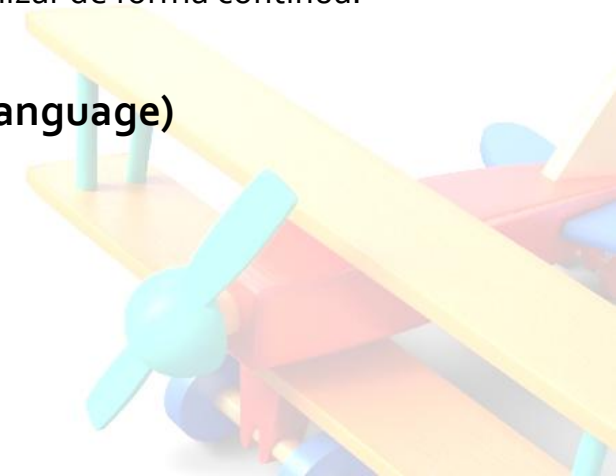
- Un **proceso de negocio** es un conjunto de tareas relacionadas lógicamente llevadas a cabo para lograr un resultado de negocio definido. Cada proceso de negocio tiene sus entradas, funciones y salidas. Las entradas son requisitos que deben tenerse antes de que una función pueda ser aplicada. Cuando una función es aplicada a las entradas de un método, tendremos ciertas salidas resultantes.

- **BPM (Business Process Management)**

- Se llama **Business Process Management** a la metodología empresarial cuyo objetivo es mejorar la eficiencia a través de la gestión sistemática de los procesos de negocio (BPR), que se deben modelar, automatizar, integrar, monitorear y optimizar de forma continua.

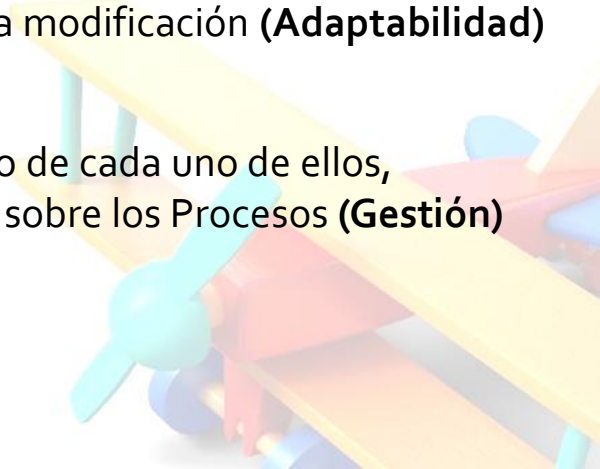
- **Lenguaje de Definición de Workflow (Workflow Language)**

- **BPEL (Business Process Execution Language)**
- **XPDL (XML Process Definition Language)**
- **YAWL (Yet Another Workflow Language)**

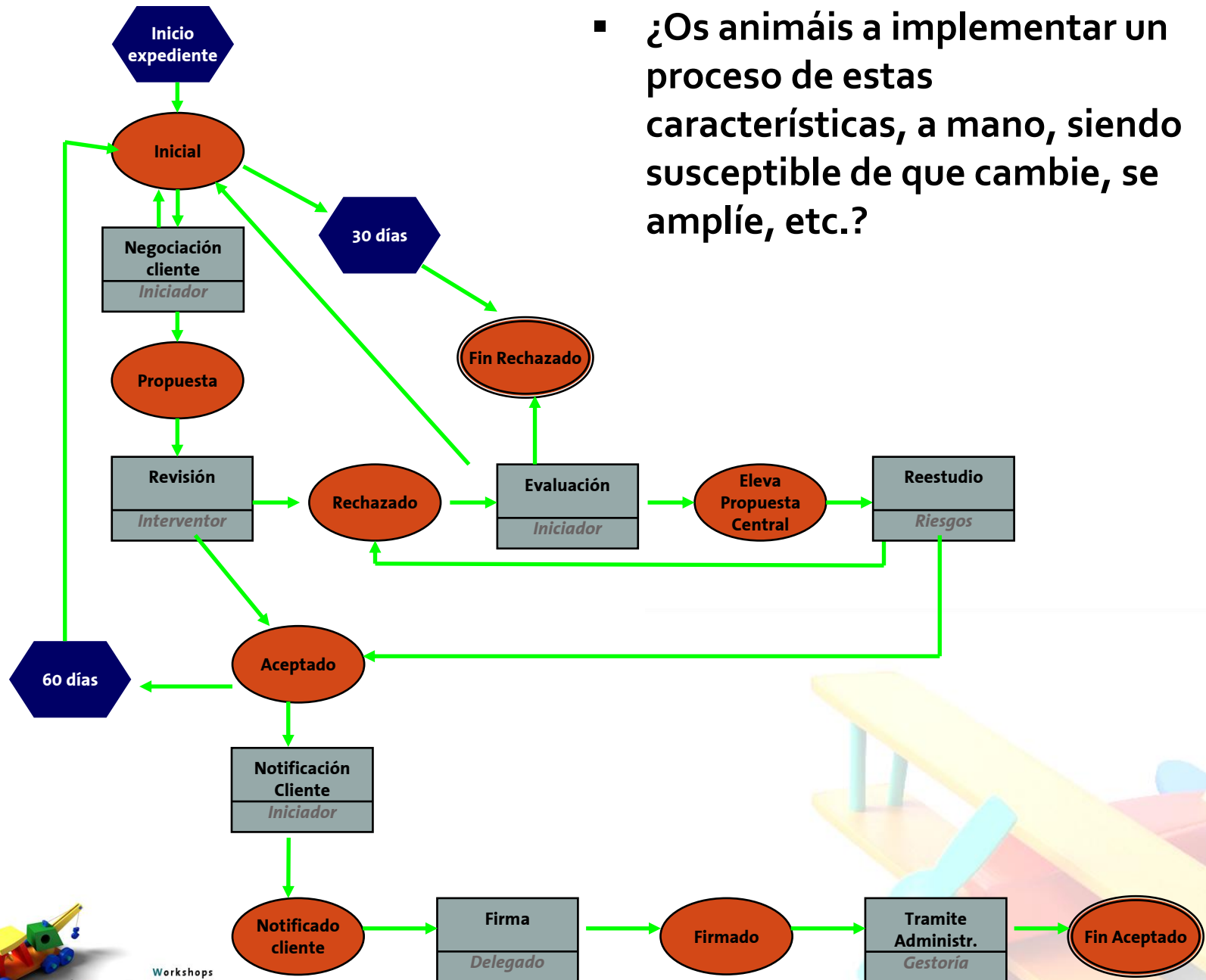


Case Study: MeGustaRefactorizar S.L.

- **Introducción**
- MeGustaRefactorizar S.L dispone de una Aplicación Web a medida (*rucu-rucu*) para gestionar la Elaboración, Presentación y Aprobación de Proyectos para Clientes.
- **Problema nº 1**
- Con las sucesivas expansiones de la Empresa, el proceso se ha ido y se irá haciendo cada vez más complejo → Facilidad de ampliación e interrelación (**Extensibilidad**)
- **Problema nº 2**
- En función del Director de Ventas, el proceso cambia alterando los pasos, algunos aspectos del proceso, etc. Para ello, se refactoriza mucho cuando realmente sólo cambia el orden de los pasos (proceso) → Facilidad en la modificación (**Adaptabilidad**)
- **Problema nº 3**
- Cuando los procesos cambian, el control sobre el estado de cada uno de ellos, monitorización, etc. se ha de reimplementar → Control sobre los Procesos (**Gestión**)



- ¿Os animáis a implementar un proceso de estas características, a mano, siendo susceptible de que cambie, se amplíe, etc.?



Case Study: MeGustaRefactorizar S.L.

■ Carta a los Reyes Magos

- Quiero poder definir un proceso fácilmente (entradas, salidas, acciones, etc.) a través de un fichero de configuración o de definición.
- Quiero modificar el proceso fácilmente a través de modificar el fichero de configuración y que todo siga funcionando.
- Quiero tener una herramienta que me permita crear una instancia de un proceso, ejecutarla, pararla, reanudarla, etc.

■ Solución

- Utilizar un Gestor de Workflows o Motor de Workflows, en su defecto.



¿Qué es un Gestor de Workflows?

■ Gestor de Workflows (Workflow Management)

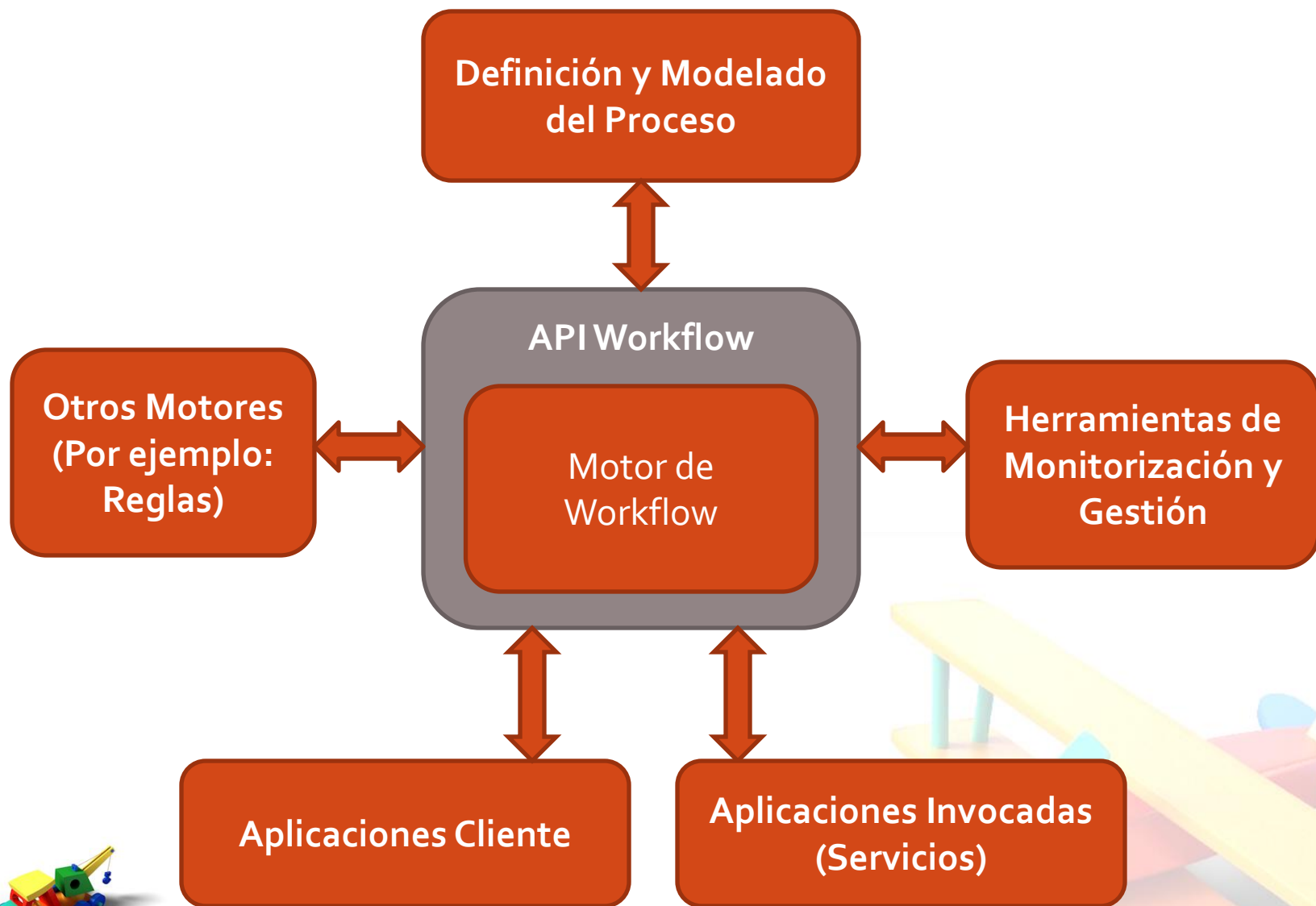
- Una aplicación que automatiza la secuencia de acciones, actividades o tareas utilizadas para la ejecución del proceso, incluyendo el seguimiento del estado de cada una de sus etapas y la aportación de las herramientas necesarias para gestionarlo

■ Objetivos de un Gestor de Workflows

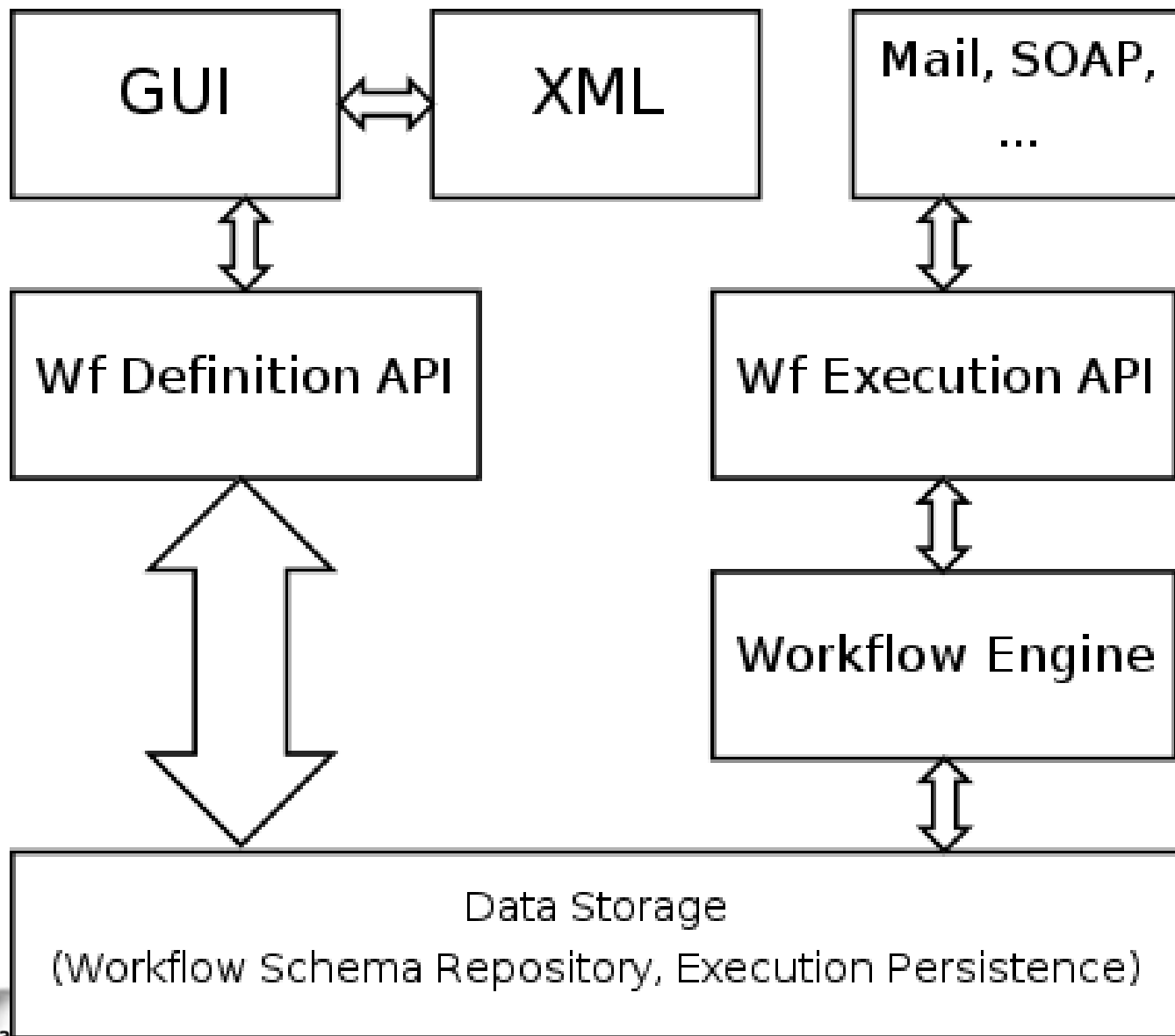
- Reflejar, mecanizar y automatizar los métodos y organización en el sistema de información
- Establecer los mecanismos de control y seguimiento de los procedimientos organizativos
- Independizar el método y flujo de trabajo de las personas que lo ejecutan
- Facilitar la movilidad del personal
- Soportar procesos de reingeniería de negocio



Elementos de un Gestor de Workflows



Arquitectura de un Motor de WFs



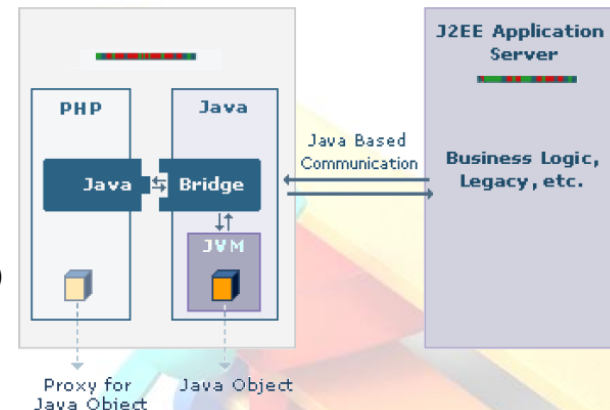
Soluciones Motores de WF en PHP

■ Soluciones Nativas en PHP

- Alto Nivel: **CuteFlow**: <http://www.cuteflow.org/> (Workflow de "Documentos", no se ajusta exactamente)
- Medio Nivel: **Galaxia Workflow** (forma parte de TikiWiki Project): <http://workflow.tikiwiki.org/> (Stand-alone complejo)
- Bajo Nivel: Usar **ezcWorkflow Components (ezComponents - <http://ezcomponents.org>)**: Subconjunto de los ezComponents (eZ Systems AS – Derick Rethans) orientado a ejecutar flujos de trabajo representados a través de Grafos
- Alternativas:
 - **Esperar** a que los frameworks MVC más utilizados (Zend, Symfony, CakePHP, CodeIgnition, etc.) implementen unos componentes parecidos a los ezcWorkflows
 - **Desarrollar nuestro propio Motor de Workflow**

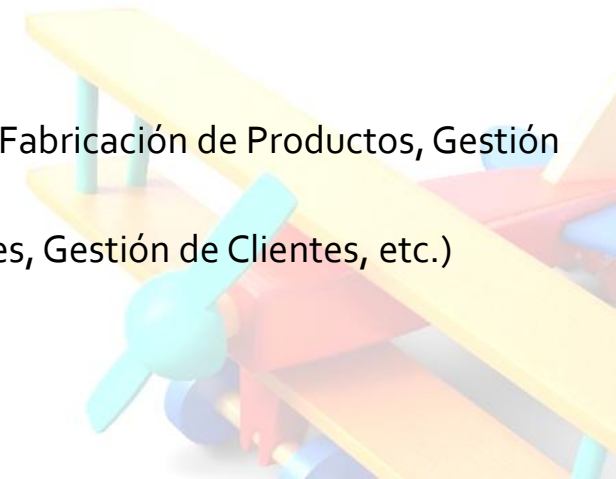
■ Otras soluciones

- Java dispone de infinitos sistemas de Workflow
- Podemos utilizarlos a través de un PHP-Java Bridge
 - **php/Java Bridge** (<http://php-java-bridge.sourceforge.net>)
- Algunos de los Motores de WF en JAVA más usados:
 - **OpenWFE** (www.openwfe.org)
 - **jBPM** (<http://jbpm.org/>)
 - **JavaFlow** (<http://commons.apache.org/sandbox/javaflow>)



Aplicaciones para los Motores de WF

- **Sistemas de Reserva Online**
 - Hoteles, Viajes, etc. (Introducir fechas y destino, elegir disponibilidad, datos bancarios, confirmación del pago, etc.)
- **ECM (Enterprise Content Management)**
- **Simulación / Árboles de Decisión**
- **Implementación de Autómatas (Reconocer Lenguajes, IA, etc.)**
- **Gestores Documentales**
 - Generación, Validación y Revisión de Documentos
- **CMS (Content Management System)**
- **Procesos de Instalación**
- **BPM (Business Process Management)**
 - Procesos de Negocio Internos (Selección de Personal, Fabricación de Productos, Gestión de Stocks, etc.)
 - Procesos de Negocio Externos (Gestión de Proveedores, Gestión de Clientes, etc.)



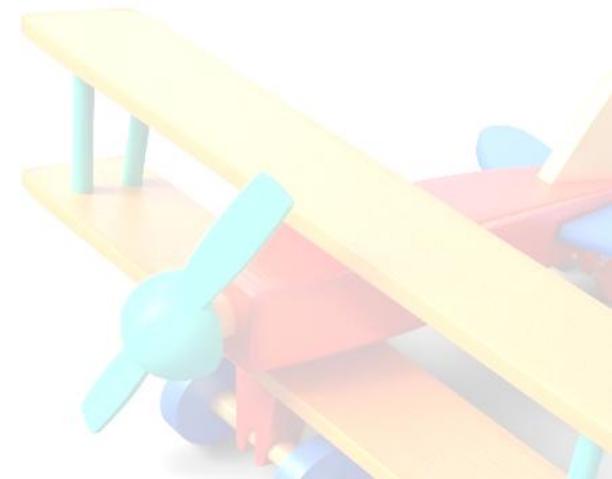
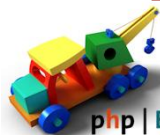
Ventajas y Desventajas

■ Ventajas

- Ahorro de Tiempo en la redefinición de Procesos (sólo modificar un XML)
- Ahorro de Tiempo en la creación de nuevos Procesos (definir proceso e implementar los Servicios), utilización de Servicios ya existentes
- Más Claridad en la definición de la Lógica de la Aplicación
- Ayuda a definir el Propio proceso de Negocio

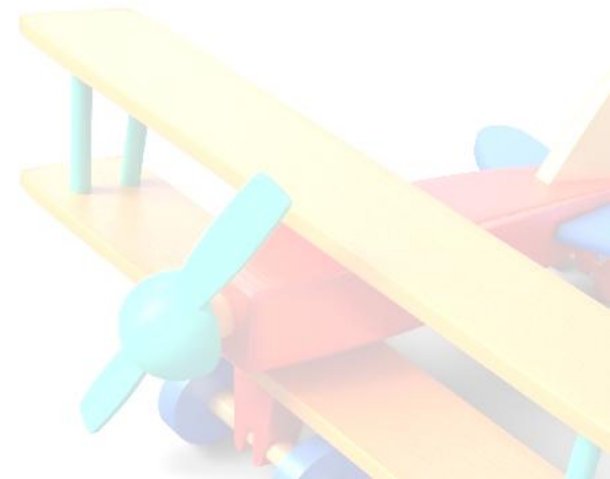
■ Desventajas

- Capa Adicional (Mayor Complejidad)
- Configuraciones Adicionales



Conclusión

- Por el tiempo/coste de Implementación, **vale la pena** usar un Motor de Workflow en Aplicaciones Web que gestionen Procesos susceptibles de modificarse y/o ampliarse en el tiempo.





php | Conceptos Prácticos

Entrando en los ezWorkflow Components



Conceptos Prácticos: ezcWorkflow

Los ezcWorkflows disponen de 2 APIs:

- **Workflow Definition API**

- API para crear, modificar y borrar definiciones de Workflows

- **Workflow Execution API**

- Funcionalidades para iniciar, suspender, parar y reanudar la ejecución de un Workflow
- Funcionalidades para monitorizar la ejecución de un Workflow

eZ Components



Workflow de Ejemplo

Diferentes Tipos de Nodo

Start, End, Cancel, Finish

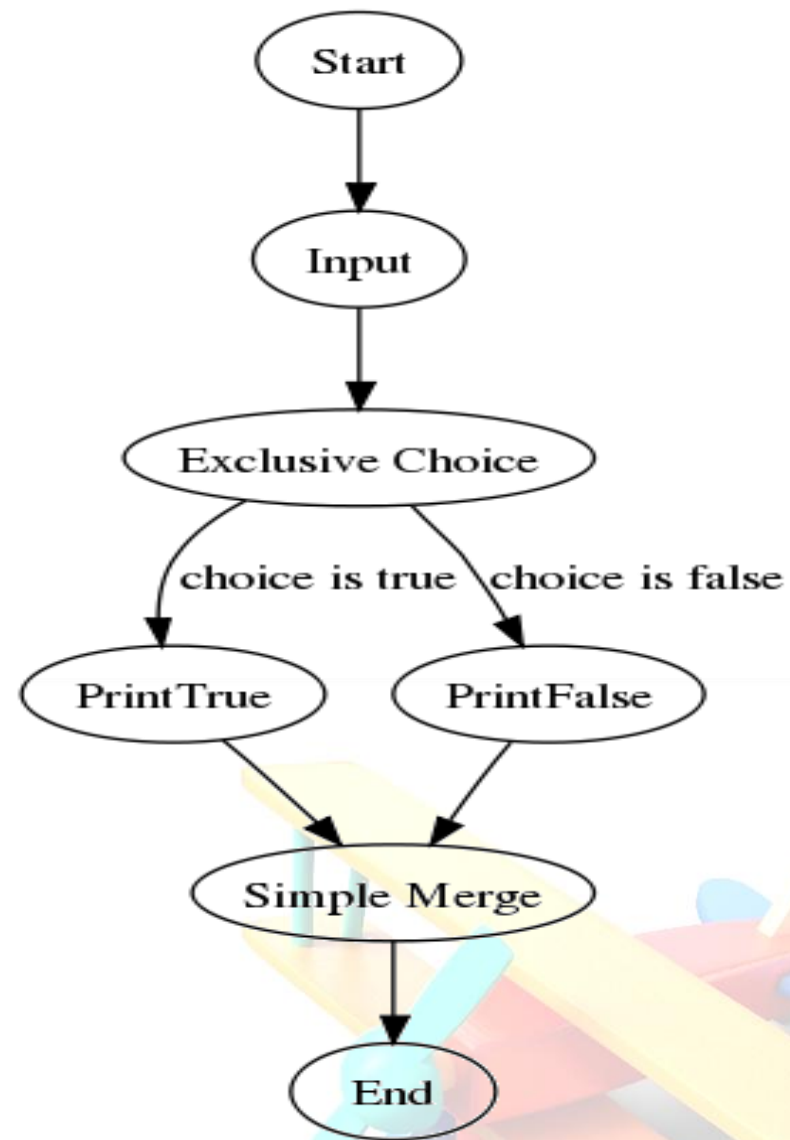
Nodos Acción

Parallel Split, Exclusive Choive, Simple Merge, Synchronization, Loop, etc.

Input Nodes

Subworkflow Nodes

Variable Modify Nodes



Código PHP en Nodos Action

1. Interficie a Implementar: execute(\$execution) y __toString()

```
class MyServiceObject implements excWorkflowServiceObject
{
    private $message;

    public function __construct( $message )
    {
        $this->message = $message;
    }

    public function execute( excWorkflowExecution $execution )
    {
        echo $this->message;

        // Manipulate the workflow.
        // Does not affect the workflow, for illustration only.
        $execution->setVariable( 'choice', true );

        // Return true to signal that the service object has finished
        // executing.
        return true;
    }

    public function __toString()
    {
        return "MyServiceObject, message {$this->message}";
    }
}
```



Definir el Workflow (Código PHP)

1. Creamos un Workflow llamado 'Test'
2. Creamos un Nodo Input que espera una variable booleana llamada 'choice'
3. Asignamos, como salida del nodo inicial, el nuevo nodo creado

```
<?php
// Create new workflow of name "Test".
$workflow = new escWorkflow( 'Test' );

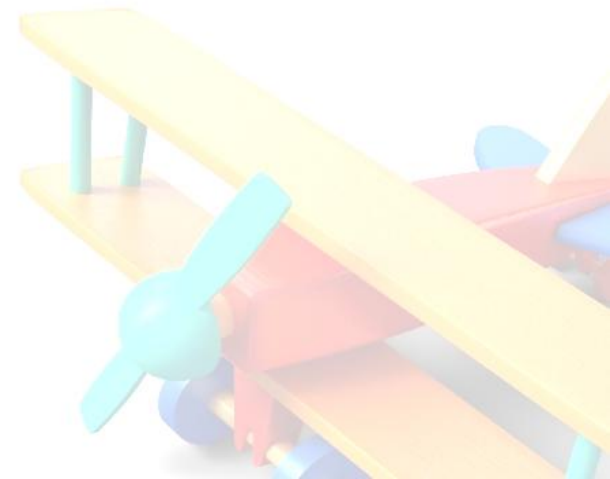
// Create an Input node that expects a boolean workflow variable of name "choice".
$input = new escWorkflowNodeInput(
    array( 'choice' => new escWorkflowConditionIsBool )
);

// Add the previously created Input node
// as an outgoing node to the start node.
$workflow->startNode->addOutNode( $input );
```

Definir el Workflow (Código PHP)

1. Creamos dos Nodos Action del Servicio 'MyServiceObject' pasando argumentos diferentes.

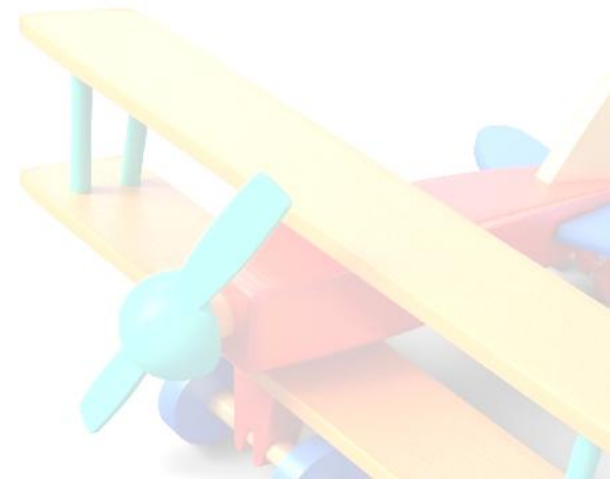
```
$trueNode = new excWorkflowNodeAction( array( 'class' => 'MyServiceObject',  
                                             'arguments' => array( 'message: TRUE' ) )  
                                       );  
  
$falseNode = new excWorkflowNodeAction( array( 'class' => 'MyServiceObject',  
                                                'arguments' => array( 'message: FALSE' ) )  
                                           );
```



Definir el Workflow (Código PHP)

1. Creamos un Nodos de Selección Exclusiva cuyo nodo precedente es el que hemos creado dos diapositivas antes.

```
// Create a new Exclusive Choice node and add it as an  
// outgoing node to the previously created Input node.  
// This node will choose which output to run based on the  
// choice workflow variable.  
  
$branch = new escWorkflowNodeExclusiveChoice;  
$branch->addInNode( $input );
```



Definir el Workflow (Código PHP)

1. Acabamos de Configurar el resto del Workflow.

```
// Branch
// Condition: Variable "choice" has boolean value "true".
// Action:    PrintTrue service object.
$branch->addConditionalOutNode(
    new escWorkflowConditionVariable( 'choice', new escWorkflowConditionIsTrue ),
    $trueNode );

// Branch
// Condition: Variable "choice" has boolean value "false".
// Action:    PrintFalse service object.
$branch->addConditionalOutNode(
    new escWorkflowConditionVariable( 'choice', new escWorkflowConditionIsFalse ),
    $falseNode
);

// Create SimpleMerge node and add the two possible threads of
// execution as incoming nodes of the end node.
$merge = new escWorkflowNodeSimpleMerge;
$merge->addInNode( $trueNode );
$merge->addInNode( $falseNode );
$merge->addOutNode( $workflow->endNode );
```



Leer y Grabar el Workflow en XML

1. Grabar el XML del Workflow.

```
<?php
// Set up workflow definition storage (XML).
$definition = new escWorkflowDefinitionStorageXml( '/path/to/directory' );

// Save workflow definition to database.
$definition->save( $workflow );
?>
```

2. Leer el XML del Workflow.

```
<?php
// Set up workflow definition storage (XML).
$definition = new escWorkflowDefinitionStorageXml( '/path/to/directory' );

// Load latest version of workflow named "Test".
$workflow = $definition->loadByName( 'Test' );
?>
```



Fichero de Definición en XML

```
<?xml version="1.0" encoding="UTF-8"?>
<workflow name="Test" version="1">
  <node id="1" type="Start">
    <outNode id="2"/>
  </node>
  <node id="2" type="Input">
    <variable name="choice">
      <condition type="IsBool"/>
    </variable>
    <outNode id="3"/>
  </node>
  <node id="3" type="ExclusiveChoice">
    <condition type="Variable" name="choice">
      <condition type="IsTrue"/>
      <outNode id="4"/>
    </condition>
    <condition type="Variable" name="choice">
      <condition type="IsFalse"/>
      <outNode id="5"/>
    </condition>
  </node>
  <node id="4" type="Action" serviceObjectClass="PrintTrue">
    <outNode id="6"/>
  </node>
  <node id="5" type="Action" serviceObjectClass="PrintFalse">
    <outNode id="6"/>
  </node>
  <node id="6" type="SimpleMerge">
    <outNode id="7"/>
  </node>
  <node id="7" type="End"/>
</workflow>
```

Definición de un Workflow
!=
Ejecución de un Workflow



Guardar el Workflow en BBDD

1. Creamos el DB Adapter
2. Nos creamos una definición del tipo de Almacenaje
3. Invocamos el método "save" pasándole el Workflow

```
<?php
// Set up database connection.
$db = escDbFactory::create( 'mysql://test@localhost/test' );

// Set up workflow definition storage (database).
$definition = new escWorkflowDatabaseDefinitionStorage( $db );

// Save workflow definition to database.
$definition->save( $workflow );
?>
```



Leer el Workflow de una BBDD

1. Creamos el DB Adapter
2. Nos creamos una definición del tipo de Almacenaje
3. Buscamos el Workflow a través de su Nombre

```
// Set up database connection.  
$db = escDbFactory::create( 'mysql://test@localhost/test' );  
  
// Set up workflow definition storage (database).  
$definition = new escWorkflowDatabaseDefinitionStorage( $db );  
  
// Load latest version of workflow named "Test".  
$workflow = $definition->loadByName( 'Test' );
```



Ejecutar un Workflow

1. Después de obtener el Workflow, se asigna a la ejecución y se invoca el método "start"

```
// Set up database-based workflow executer.  
$execution = new excWorkflowDatabaseExecution( $db );  
  
// Pass workflow object to workflow executer.  
$execution->workflow = $workflow;  
  
// Start workflow execution.  
$id = $execution->start();
```



Resumir y Cancelar un Workflow

1. Cancelamos invocando el Método "cancel".

```
// Cancel workflow execution.  
$execution->cancel();
```

2. Después de obtener el Workflow utilizamos el método "resume"

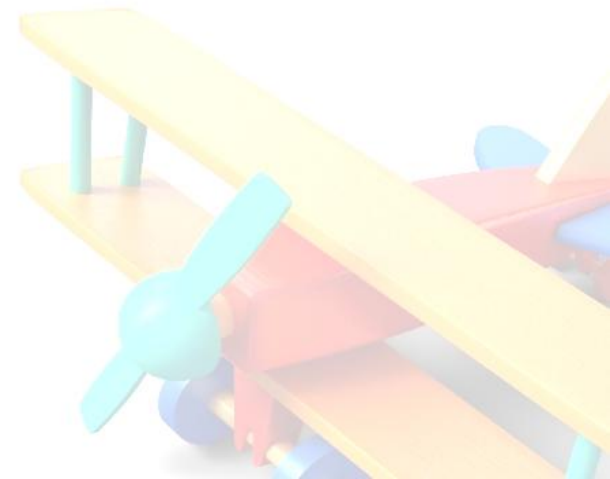
```
// Set up database connection.  
$db = escDbFactory::create( 'mysql://test@localhost/test' );  
  
// Set up database-based workflow executer.  
$execution = new escWorkflowDatabaseExecution( $db, $id );  
  
// Resume workflow execution.  
$execution->resume(  
    array( 'choice' => true )  
);
```



Material Teórico

- Teoría de Workflows

- <http://martinfowler.com/articles/languageWorkbench.html>
- <http://docs.jboss.com/jbpm/v3/userguide/graphorientedprogramming.html>
- <http://www.esw.inesc.pt/~ars/ps/sofsem2004.pdf>
- <http://www.yawlfoundation.org/documents/yawls.pdf>
- http://www.wfmc.org/standards/docs/TC-1025_xpdl_2_2005-09-07_xpdl_2.pdf





php | Muchas Gracias

¡ Muchas Gracias por vuestra Atención !

