PHP4DB: Evolucionando hacia nuevos objetivos para el Desarrollo Ágil de Aplicaciones Web

Lisandro Delía[1], Germán Cáseres[2], Hugo Ramón[3], Pablo Thomas[4], Rodolfo Bertone[5]

Instituto de Investigación en Informática LIDI (III-LIDI)[6]
Facultad de Informática - UNLP

Abstract

Initially, a group of software developers faced the problem of having a software layer capable of allowing the flexible development of Web applications, basically using PHP as working language and classic database motors, in this case, MySQL and MSSQL Server. The basic purpose of the first version, generated in 2006, was to automatically apply the basic functionalities (additions, modifications, deletions, commonly known as AMD) on the database tables in question. The result of the work developed was a product called PHP4DB, name that was used to present the product in several research and development papers.

Based on the use of PHP4DB, the functionality that was provided initially was insufficient. The continuous evolution in the development of management systems and the needs of users/customers caused the framework to evolve as well, presenting new characteristics, features and abilities. The purpose of this work is to describe the improvements introduced and also present a series of objectives that are being currently developed.

Keywords: Software Engineering, Flexible Developments, WEB Applications, CASE Tools

Resumen

Inicialmente un grupo de desarrolladores de software enfrentaron el problema de disponer de una capa de software capaz de permitir el desarrollo ágil de aplicaciones Web, básicamente utilizando a PHP como lenguaje de trabajo y motores clásicos de Base de Datos, en este caso MySQL y MSSQL Server. La primera versión generada durante el 2006, tenía como finalidad esencial generar en forma automática la funcionalidad elemental (Altas, Bajas y Modificaciones, comúnmente denominado ABM) sobre las tablas de Base de Datos del problema. El resultado del trabajo desarrollado fue un producto denominado PHP4BD, nombre bajo el cual fue presentado en diversos trabajos de investigación y desarrollo.

A partir de la utilización de PHP4DB, la funcionalidad brindada inicialmente quedó limitada. La continua evolución en el desarrollo de sistemas de gestión y las necesidades de los usuarios/clientes hicieron que el framework evolucionara, presentando nuevas características, rasgos y habilidades. El propósito de este trabajo es describir las mejoras introducidas y presentar, además, una serie de objetivos que en este momento están siendo desarrollados.

Palabras Claves: Ingeniería de Software, Desarrollos Ágiles, Aplicaciones WEB. Herramientas CASE

---

[1] {ldelia@lidi.info.unlp.edu.ar} Ayudante Diplomado DS, UNLP - Fac. de Informática.

[2] {gcaseres@lidi.info.unlp.edu.ar} Becario III-LIDI UNLP - Fac. de Informática-

[3] {hramon@lidi.info.unlp.edu.ar} Profesor Adjunto DE, UNLP - Fac. de Informática.

[4] {pthomas@lidi.info.unlp.edu.ar} Profesor Adjunto DE, UNLP - Fac. de Informática.

[5] {rbertone@lidi.info.unlp.edu.ar} Profesor Titular DE, UNLP - Fac. de Informática

[6] III-LIDI UNLP - Fac. Informática, calle 50 y 115, La Plata (1900), Buenos Aires, Argentina, Tel/Fax +54 221 4337707 http://www.lidi.info.unlp.edu.ar

1. **Introducción**

A comienzos de la década del 2000 un de grupo personas de la comunidad informática presenta lo que se conocería como un manifiesto, que describe las bases técnicas y de proceso, las cuales se deberían seguir para desarrollar Sistemas de Software. Este grupo utilizó "Métodos Ágiles" para definir a los métodos que aparecían como alternativa a las metodologías formales, consideradas como excesivamente rígidas y colmadas de pasos minuciosos y exhaustivos que debían cumplirse. Estas metodologías necesitaban de una definición muy precisa de requerimientos, planificaciones y estimaciones que pudieran guiar el desarrollo del software [1].

Las bases presentadas oportunamente apuntaban a un cambio importante en el razonamiento de los sistemas [2]:

- Trabajo individual con alta interacción sobre procesos y herramientas.
- Software que funcione rápidamente como algo más prioritario que una documentación exhaustiva.
- Colaboración total del cliente y participación abierta del mismo, como elemento más importante del contrato que rige del desarrollo
- Alta respuesta al cambio y adaptación al cambio sin un alto estrés, más que seguir un plan de cambios preestablecido.

Desde esta perspectiva se puede generar una amplia discusión entre lo que se denominan métodos clásicos y los métodos de desarrollos ágiles. No es menester de este trabajo discutir estas diferencias. Con el solo fin de presentar una opinión de los autores, cada metodología tiene una concepción diferente y está ligada a desarrollo de diferentes tipos de sistema.

Para proyectos de gran envergadura, de más de 5000 puntos función (PF), las metodologías ágiles presentan más inconvenientes que ventajas. En este tipo de problemas, la complejidad es alta, las actividades son amplias y minuciosas, el personal para el desarrollo es numeroso, por lo tanto es necesario contar con las metodologías clásicas que guíen el proceso. Una metodología ágil rápidamente generará un caos importante entre el grupo de desarrollo, fricciones con el cliente, problemas de costos, agendas, etc. [3] [4].

Además, existen desarrollos de sistemas con productos finales de algunos cientos o pocos miles de PF, generalmente denominados software de mediana o pequeña envergadura, donde las características básicas son: alta participación del cliente, demanda cambiante de requerimientos dado que el usuario/cliente "aprende" y aumenta sus necesidades a partir del uso del sistema, grupo de desarrollo limitado (con un máximo de entre 8 y 10 personas). En este contexto, la utilización de una metodología clásica resulta muy lenta y tediosa. Aquí, entonces, los métodos ágiles (Programación Extrema o Scrum entre otros) surgen como soluciones viables. En estos últimos casos, el grupo de trabajo se encuentra inmerso.

Otro detalle de importancia tiene que ver con el tema de calidad. Se plantea, además de la discusión anterior, cuestiones como: los métodos ágiles se "alejan" rápidamente de cualquier establecimiento de calidad. Este tema no pasó desapercibido, oportunamente se realizaron estudios y evaluación de alternativas y se llegó a la conclusión que se puede trabajar con metodologías ágiles y aún así conseguir CMM nivel 2 y parcialmente nivel 3 [5]. Además, estudios posteriores, compararon las metodologías ágiles contra otras normas, Moprosoft primero y CompetiSoft luego, arribando a las mismas conclusiones [6] [7].

El desarrollo de sistemas siguiendo una metodología ágil requiere, entonces, contar con herramientas que puedan minimizar, entre otros, el tiempo necesario de la codificación de tareas rutinarias. Si bien este tiempo es mínimo dentro del ciclo de desarrollo de un sistema, las tareas repetitivas no específicas del dominio de aplicación ocupan generalmente entre un 50 y un 60% del

tiempo total asignado. Además, la puesta a punto y depuración de la funcionalidad, y la generación de interfaz de usuario resulta en valores temporales que no pueden considerarse despreciables [8] [9].

Una vez definidos los requerimientos iniciales del usuario/cliente es posible desarrollar un modelo de datos completo, ágil y dinámico que los represente de la forma más adecuada. Desde este punto, un sistema que implante la funcionalidad requerida necesitará de módulos básicos que administren la información contenida en la BD (Base de Datos). El desarrollo y mantenimiento de cada uno de estos módulos requiere dedicar tiempo a tareas rutinarias manteniendo la consistencia de interfaz y correctitud [10].

Asimismo, el desarrollo de sistemas de software requiere aplicar ciertos principios de seguridad, tales como la protección contra accesos no autorizados o la modificación de información contenida en la BD, la negación de servicios a usuarios desautorizados y la provisión de servicios a usuarios acreditados, incluyendo además las medidas necesarias para detectar, documentar, y contrarrestar las amenazas.

El equipo de desarrollo debe concentrarse en la programación de las funcionalidades mínimas (utilizando un lenguaje de programación), actualizar la BD (generalmente con otro lenguaje específico), armar los formularios de carga de datos, grillas, combinando componentes visuales, entre otras actividades [11]. Además, un aspecto importante de toda aplicación, en particular cuando se trata de un sistema de información, es la coherencia en el desarrollo de la interfaz, debiendo presentar la información e interactuar con el usuario en forma homogénea y consistente.

Desde estas perspectivas, y como se mencionó previamente, durante el año 2006 surgió la necesidad de crear PHP4DB como herramienta para el desarrollo rápido de aplicaciones, con las siguientes características:

- Facilidad de uso por parte del usuario programador.
- Curva de aprendizaje rápida
- Cumplir con los niveles establecidos de calidad.
- Estabilidad.
- Permitir al usuario cliente participar y eventualmente utilizar el producto.
- Interactuar con las BD más utilizadas en el desarrollo de software de pequeña y mediana envergadura.
- Seguir parámetros establecidos y aceptados en la comunidad del desarrollo del software [12].
- Permitir un mantenimiento sencillo y, básicamente, rápido del software realizado.

2. El problema original

En su primera etapa el problema consistió en desarrollar un entorno que permitiera a los programadores generar las operaciones clásicas sobre una BD, tales como listados, filtros, reportes, altas, bajas y modificaciones (ABM) de datos. A cada tabla junto a sus operaciones clásicas asociadas, desde aquí serán denominadas *repositorios*.

Los proyectos involucrados, sistemas Web en esencia, están desarrollados con herramientas open-source, en ambientes LAMP (Linux + Apache + MySQL + PHP) [13] e interactúan con BD's integradas por información heterogénea. Esto último provoca un gran esfuerzo para generar la interfaz de cada *repositorio*. Por lo tanto, es fundamental contar con una capa de software genérica que automatice el desarrollo de los repositorios y otra capa encargada de llevar el control del acceso

al sistema y sus funcionalidades, que actúe como un “molde” para todos los proyectos de similares características.

La complejidad en desarrollar estas capas, depende del tipo de aplicación en cuestión. En aplicaciones tipo RAD (Rapid Application Development) tales como Delphi, PowerBuilder, VisualBasic, es posible parametrizar componentes para lograr *repositorios* con poco esfuerzo [14]. En aplicaciones Web, basadas en tecnología cliente-servidor [15], la solución se presenta un tanto más compleja. Esto es, se debe resolver el proceso tanto del lado del cliente (mediante JavaScript, Java Applets), como del lado del servidor (mediante PHP, ASP, JSP, etc), en conjunto con un modo de mostrar la información (HTML, XML + CSS).

PHP4DB se diseñó para resolver los problemas presentados. PHP4DB es una herramienta orientada a objetos desarrollada íntegramente en PHP, con el objetivo de generalizar lo más posible la capa de software que permita automatizar tareas rutinarias de codificación en un ambiente LAMP y brindar un esquema de seguridad ágil, dinámico y confiable [16].

3. Arquitectura y descripción

3.1 Funcionalidad

PHP4DB fue un desarrollo netamente evolutivo: se plantearon una serie de objetivos básicos los cuales, una vez alcanzados, permitieron “evolucionar” tanto en complejidad como en completitud. En la versión actual es posible realizar las siguientes tareas automáticamente para cada repositorio:

- Visualizar los datos de un *repositorio* mediante una grilla paginada.
- Filtrar dinámicamente los datos del *repositorio* según atributos definidos para tal fin
- Obtener una vista rápida de una fila de la grilla
- Visualizar el manual de ayuda de un *repositorio*
- Generar ABM de datos mediante un formulario pre-establecido
- Generar un reporte formato PDF de todos los datos que se visualiza en la grilla o de un dato en particular
- Exportar a archivos CSV todos los datos que se visualiza en la grilla
- Relacionar un dato en particular con otra funcionalidad externa al *repositorio*
- Auditar en formato XML cada operación que el usuario lleva adelante en el *repositorio*

Para esto, PHP4DB se comunica dinámicamente con la BD del problema a resolver, para recuperar o actualizar la información allí contenida y presentar en el navegador del usuario los resultados mediante código HTML. La intención es que el código generado sea compatible por cualquier navegador, y para ello PHP4DB genera código estándar según la W3C (World Wide Web Consortium) [16].

El desarrollo de esta herramienta estuvo ideado, desde un principio, para llevarse a cabo en productos de licencia libre. Por este motivo, el DBMS utilizado fue MySQL. En versiones posteriores, se observó que las limitaciones implantadas a partir del uso de un DBMS particular no eran adecuadas, y por este motivo evolucionó para abstraerse del motor de BD particular. Para lograr la abstracción requerida se utilizó la librería DB de PEAR (PHP Extension and Application Repository) [17] y la nueva MDB2 de PEAR (combinación de Metabase php Database abstraction layer y DB) [18].

La Figura 1 presenta la ejecución de un *repositorio* como parte de un sistema. Inicialmente, como acceso principal a dicho *repositorio*, se muestra una grilla paginada de datos con un formulario de filtro asociado. A partir de aquí, es posible acceder a todas las demás funcionalidades del

Framework. En la grilla se describen los datos listados, los cuales pueden ser derivados, como se dijo anteriormente, a un reporte en formato PDF.

A los datos mostrados se le pueden aplicar acciones, algunas básicas, como la modificación o baja, y otras específicas al *repositorio* vinculadas con el comportamiento definido por los requerimientos del sistema. Toda la funcionalidad específica se presenta asociada a cada fila de la grilla y se aplica sobre ella. La presentación puede ser mediante una lista desplegable o una barra de herramientas.

Además, como funcionalidad básica, es posible insertar nuevos elementos. La figura 2 presenta un ejemplo de este formulario, mientras que la figura 3 presenta la vista de un registro particular ante una consulta.

La capa de seguridad de PHP4DB, por su parte, provee las siguientes funcionalidades:

- Login/Logout al sistema web
- Recuperación y cambio de contraseña
- Bloqueo temporal de un usuario ante reiterados fallos de login
- Trazabilidad de las acciones de los usuarios
- Generación dinámica del menú del sistema, según las funciones a las que el usuario tiene acceso.
- Administración de perfiles
- Asociación de funciones a perfiles
- Administración de usuarios
- Control de acceso en los repositorios

Figura 1: Ejemplo de repositorio

3.2 Estructura

PHP4DB esta diseñado como un núcleo centralizado encargado de crear toda la funcionalidad mencionada en la sección previa. Para ello, cada *repositorio* definido utiliza la funcionalidad del núcleo para la presentación de la información.

El núcleo necesita ser configurado para cada proyecto específicamente, a fin de responder a cada aplicación desarrollada en el Instituto. Esto da lugar al ProyectDataScript (PDS), archivo de

configuración de la aplicación, el cual cuenta con la descripción de la BD del proyecto (servidor, DBMS, usuario, clave), el estilo que tendrán las interfaces (CSS, Iconos), así como información adicional necesaria.

Mediante el PDS el núcleo tiene a su disposición la configuración del sistema y la información en común de todos los *repositorio*s.

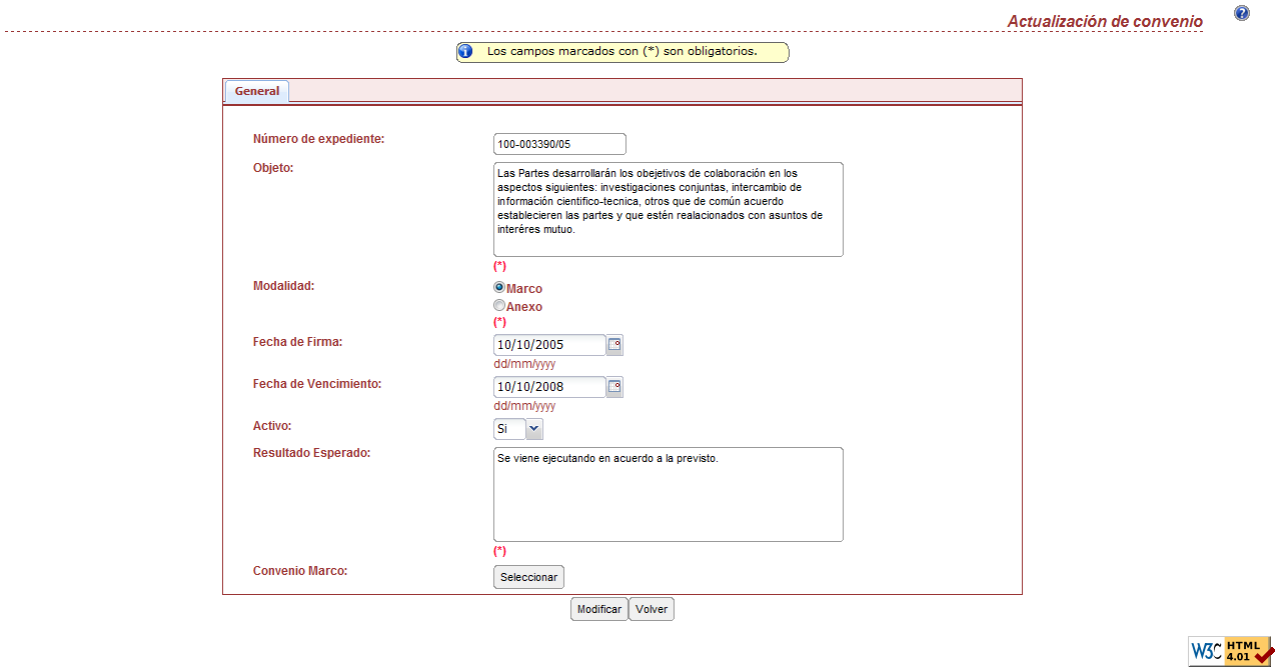

Figura 2: Formulario Alta/Modificación

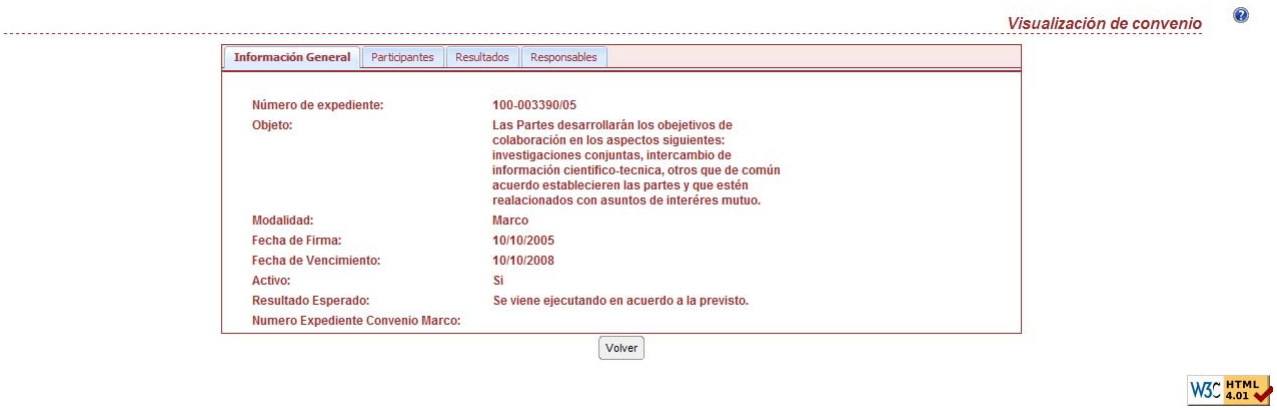

Figura 3: Vista rápida de un registro

Además se definió un archivo denominado FDS (FormDataScript), encargado de brindar toda la información específica de cada *repositorio* y de la tabla que hace referencia. Con este descriptor de datos, PHP4DB puede brindar toda la funcionalidad para el *repositorio* asociado. Los FDS describen, entre otras cosas, la siguiente información:

- Títulos para cada operación del *repositorio*
- Nombre de la tabla de la BD, que hace referencia el *repositorio*
- Nombre de la tabla de la BD a la que hace referencia el *repositorio.*

- Campos de la tabla de la BD, donde para cada uno de ellos se tiene:
    - Nombre del campo.
    - Etiqueta (Label) significativa a mostrar.
    - Visibilidad en grilla/reportes.
    - Visibilidad en el filtro.
    - Tipo del campo.
- Permisos para cada función del *repositorio*, con el objetivo de habilitar/deshabilitar funciones de acuerdo al perfil del usuario

En cuanto a la capa de seguridad, el Framework presenta un esquema de "Acceso según perfil". Es decir, los usuarios del sistema pueden ser miembros de distintos perfiles o grupos, los cuales permiten acceder a funciones asociadas a éstos. Por lo tanto el usuario solo puede acceder al conjunto de funciones asociadas a sus perfiles. Estas funciones son accedidas desde el menú del sistema creado dinámicamente, luego de determinar los perfiles del usuario al momento del "loggeo".

Una característica importante de la capa de seguridad es la trazabilidad de las acciones del usuario. Todo evento ocasionado por el usuario (desde login/logout, hasta los accesos a las funciones del sistema) es registrado, para que luego, un usuario administrador pueda hacer seguimientos y estadísticas de los usuarios.

La Figura 4 presenta la estructura del Framework PHP4DB. Cada FDS, contiene la información de un *repositorio* en particular. Cuando se invoca a uno de estos *repositorio*s, el FDS envía toda su información al núcleo PHP4DB, quien lleva a cabo alguna de sus funciones, recuperando la información de la BD.

Cabe destacar que cada nueva funcionalidad que se le incorpore al núcleo del Framework, como puede ser por ejemplo la exportación de datos a un formato particular, es obtenida por cada *repositorio* sin realizar modificación alguna. Lo mismo ocurre con el mantenimiento de cada funcionalidad o corrección de errores, todo *repositorio* recibirá estos beneficios sin necesidad de alterar su contenido.

Es interesante destacar que cuando se necesita crear un nuevo *repositorio*, no es necesario agregar programación alguna (ya sea código PHP, HTML o SQL). Solo se debe crear el FDS asociado al *repositorio* y el Framework será el responsable de crear las interfaces, mostrar los datos, ejecutar las funciones, etc.

3.2 Asistente para la creación de los FDS

Si bien los desarrolladores pueden generar/modificar los FDS manualmente, se desarrolló una herramienta que automatiza esta tarea, ahorrando tiempo y evitando errores que pueden surgir con la opción manual.

Esta herramienta es una aplicación de escritorio denominada PHP4DB Assistant, que en pocos pasos permite crear los FDS para cada *repositorio*. Teniendo en cuenta que PHP4DB es un Framework pensado para que coexista como núcleo de varios proyectos simultáneamente, la primera tarea consiste en utilizar el PDS asociado al proyecto donde se desea agregar el nuevo *repositorio*. PHP4DB Assistant permite visualizar las tablas de la BD del proyecto, para queel usuario elija la tabla específica que será referenciada por el nuevo *repositorio*.

Seleccionada la tabla, automáticamente se despliega la información de sus campos , restando configurar ciertos detalles tales como: (1) los *labels* que tendrán los campos, (2) el tipo de dato base de cada campo (texto, número, clave foránea, etc), (3) los títulos de las diferentes acciones, (4)

las funcionalidades que estarán activas para el *repositorio*, etc. La figura 5 presenta un resumen de estos detalles.

Una vez configurado, el archivo FDS se almacena en un servidor web, y se está en condiciones de presentar el *repositorio* desde cualquier navegador.

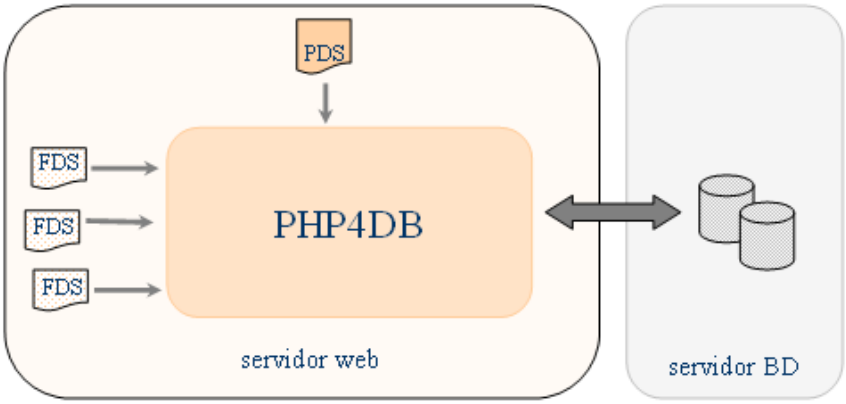

Figura 4: Estructura del Framework PHP4DB

3.3 Características de los repositorios

Tipos de campos

En los archivos FDS se describe la información sobre los campos a presentar en los formularios, grillas, filtros y reportes. Estos campos varían unos a otros. Por ejemplo, la manera de presentación de una fecha en un formulario, no debería ser semejante a mostrar un texto. Es por esto que PHP4DB identifica a cada campo con un tipo en particular.
Actualmente el Framework puede trabajar con los siguientes tipos de campos:

- Textos cortos y extensos
- Números enteros y flotantes
- Fechas
- Imágenes
- Archivos
- Conjunto definidos por el usuario (pudiéndose representar mediante RadioGroups, ComboBoxs o ListBoxs)
- Campos foráneos
- Campos Booleanos
- Custom. Clase definida por el usuario que hereda de algún tipo primitivo.

Al disponer de un diseño orientado a objetos, agregar un nuevo tipo de dato al Framework no es una tarea costosa. De esta forma es posible extender el Framework fácilmente ampliando el alcance de los *repositorios*.

Eventos

Si bien todas las funcionalidades básicas de un *repositorio* se pueden resolver automáticamente, existen casos en que algunas de ellas necesitan comportarse de otra forma.

PHP4DB brinda la posibilidad de que los repositorios tengan orientación a eventos, permitiendo ejecutar acciones en determinados momentos de la ejecución. Para llevarlo a cabo, PHP4DB verifica si el FDS que está siendo ejecutado tiene definido el evento correspondiente al punto de

ejecución. De estar definido, PHP4DB invoca al evento; caso contrario sigue trabajando normalmente. Eventos como *before_execute_insert()* o *after_execute_insert()* por citar ejemplos, aumentan el nivel de dinamismo del Framework

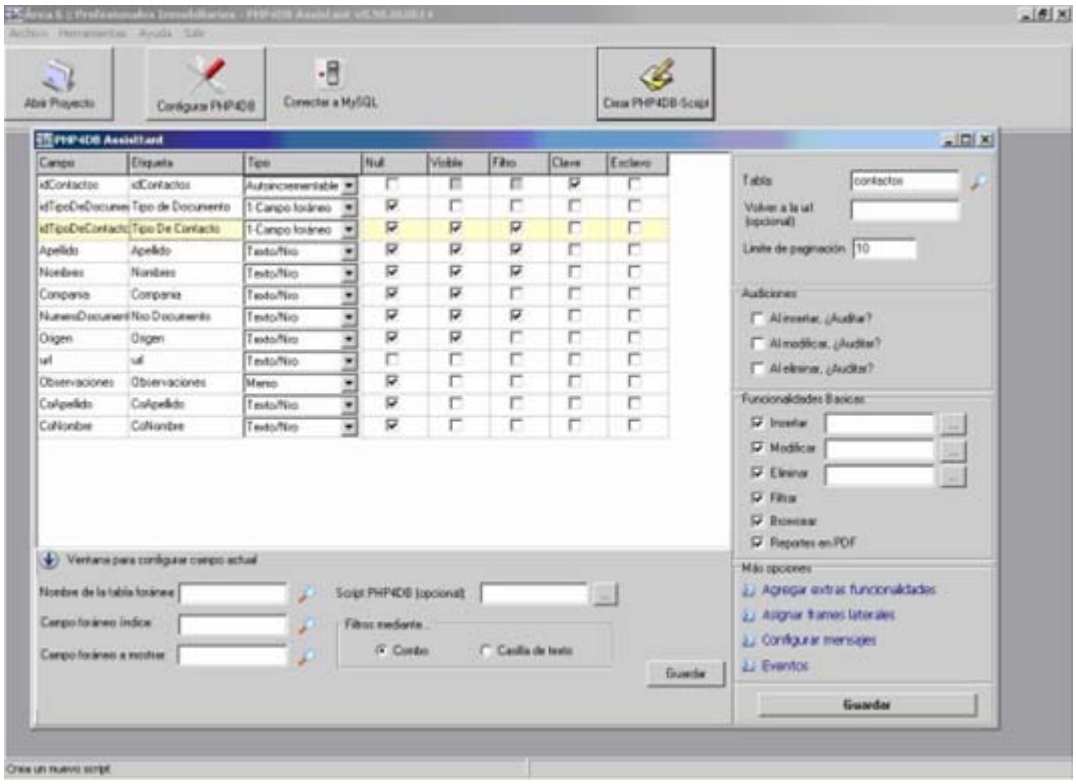

Figura 5: Creación de un repositorio con PHP4DB Assistant

Relación con otras funcionalidades

Se mencionó que el marco ideal de trabajo seria que el equipo de desarrollo dedique el tiempo en los módulos que requieren una programación específica, ad-hoc al problema a resolver, sin perderlo con el desarrollo de los *repositorios*.

En caso de contar con módulos específicos, existe la necesidad de poder relacionarlo con los *repositorios*. La Figura 6 muestra como se puede relacionar un registro en particular con otras funciones del sistema. Mediante una lista desplegable, o simples iconos en cada fila de la grilla, se le puede aplicar una acción a un registro seleccionado. Estas acciones implican llamadas a otros módulos que han sido desarrollados específicamente, o bien, otros repositorios creados con PHP4DB Assistant.

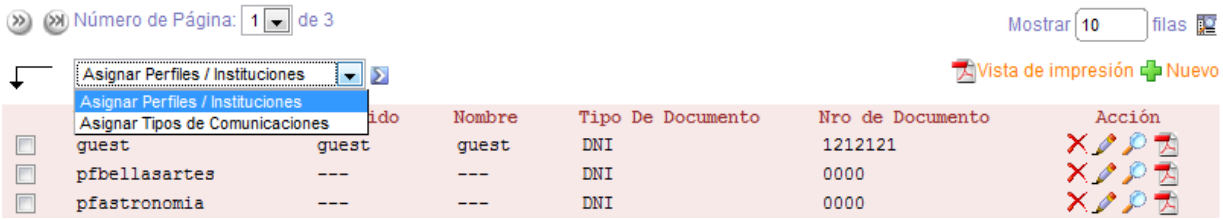

Figura 6: Relación con otras funciones

4. Evolución de PHP4DB

A continuación se enuncian los avances más significativos incorporados a PHP4DB en relación a su versión inicial [10]:

- Especificación de hojas de estilos especificas a un repositorio en particular
- Posibilidad de añadir código Javascript a un repositorio en particular
- Validaciones para los formularios tanto del lado del cliente como del lado del servidor
- Completa y amigable visualización de errores de carga en los formularios de los repositorios
- Posibilidad de permitir formularios de asignación múltiples
- Cumplimiento del standard de HTML según la w3c
- Generación automática de páginas de ayuda para los repositorios
- Posibilidad de exportar datos a formato CSV
- Nuevos tipos de dato para los repositorios:
    - Archivos
    - Conjuntos definido por el usuario, representándose mediante RadioGroup, ComboBox o ListBox
    - Custom, donde el usuario puede definir el comportamiento deseado
- Capa de seguridad y auditoria
- Posibilidad de definir pestañas y grupos de campos, para presentar la información de forma más clara y ordenada

5. Resultados obtenidos

Desde la definición del Framework ha sido utilizado en diversos proyectos, lo que ha permitido capturar experiencias, traducidas en la continua evolución de este producto. En líneas generales, los resultados obtenidos pueden catalogarse en varios grupos:

- **Mejora del tiempo de respuesta para implementar los requerimientos más elementales de un sistema de software.** En este punto se puede concluir que luego de la implantación del producto, se redujeron en un mínimo de 50% y hasta el 80% los tiempos requeridos para el desarrollo de la algorítmica clásica y básica de un Sistema de Software(altas, bajas, modificaciones, filtrados y listados generales).

- **Rápido aprendizaje en el uso de la herramienta.** En general, los programadores con experiencia en lenguajes web tienen una curva de aprendizaje muy rápida para lograr utilizar los aspectos generales y aún particulares de PHP4DB. Además, profesionales informáticos con poca experiencia en el desarrollo de sistemas web, particularmente utilizando PHP, lograron obtener resultados interesantes dentro de primera semana de utilización.

- **Alto nivel de confianza y reducción de los tiempos de prueba.** Al disponer de una herramienta estable, los desarrollos tradicionales obtenidos automáticamente no necesitaron pruebas exhaustivas del software, lo que produce mejoras en términos de calidad de desarrollo de software. Estas pruebas se limitan a evaluar con el usuario/cliente, la funcionalidad generada y compararla con la requerida.

- **Auditoría y control de cambios.** Dentro de las mejoras presentadas en la última versión de la herramienta, está el proceso de auditoría y control de cambios en la información de la BD. Así, es posible que el administrador del sistema defina de manera dinámica la información que será auditada por PHP4DB, como elemento adicional dentro del sistema desarrollado. Además, dentro de las atribuciones del administrador estará la definición de aquellos datos

considerados indispensables para el sistema, sobre los cuales un usuario estándar con bajos permisos no podrá realizar determinadas operaciones.

- **Versatilidad para la incorporación de código particular por el programador.** De esta forma, es posible incorporar fácilmente la funcionalidad propia a cada sistema. Si bien las posibilidades de la herramienta está en continua evolución, siempre hay características puntuales que se necesita del software. PHP4DB está pensado para incorporar de una manera sencilla código específico.

Entre los proyectos que utilizaron PHP4DB podemos mencionar

- **Hospital Felipe Pelaez. Municipalidad de Florentino Ameghino**: Es indiscutible la importancia social del sistema sanitario, en todos sus niveles. En particular en pequeños Municipios "el Hospital" constituye uno de los ejes de la actividad del estado, que debe combinar un servicio adecuado a la población con un manejo eficaz y la mejor utilización posible de los recursos humanos y tecnológicos disponibles. La complejidad de los sistemas involucrados y la cantidad de información en juego hacen muy necesario el empleo de herramientas informáticas, para la gestión y también para la toma de decisiones dentro del Hospital y a nivel del Municipio. El proyecto apunta a resolver un "caso problema" centrado en Hospitales de pequeño porte (100 camas máximo) que sea utilizable en un número importante de Municipios, requiriendo el menor costo posible de equipamiento informático y mantenimiento mensual. [19][20].

- **Auditoría de la Red Única Provincial de Comunicación de Datos del Gobierno de la Provincia de Buenos Aires a través de la RedPIBA de la CICPBA**: Definición del manual de los procesos y productos necesarios para la aceptación de los nodos de acuerdo a la licitación del Gobierno de la Provincia de Buenos Aires para proveer "Servicio de Transmisión de Datos y Canales de Ordenes". Implementación del software de seguimiento del proyecto Red Única Provincial de Comunicación de Datos, que vincula 135 Municipios (con alrededor de 1300 nodos) planificando la verificación de las instalaciones y la puesta en servicio (incluyendo el control de ancho de banda en tiempo real) de toda la red [21] [22].

- **Relaciones Internacionales de la UNLP:** Desarrollo un sistema cuyo objetivo es mantener información relacionada a la participación de los miembros de la UNLP en actividades con el exterior, y así fomentar el conocimiento, la colaboración y la responsabilidad de informar y registrar dichas participaciones. El sistema se divide en dos subsistemas: *Gestión de convenios;* administra la información relacionada con los convenios entre la Universidad (o alguno de sus miembros) e instituciones del exterior. *Gestión de características de las facultades;* administrará información relacionada con las fortalezas y debilidades de las diferentes unidades académicas de la UNLP, además de las líneas de investigación/trabajo y la participación en diferentes redes internacionales [22].

- **Area6 Team**: Desarrollo un Sistema orientado a la Web para un Consorcio de Inmobiliarias de España. Esta aplicación, actualmente en producción, incluye las operaciones transaccionales clásicas de altas, bajas y modificaciones, y la posibilidad de realizar un análisis comparativo de mercado, esta funcionalidad, compleja para cualquier aplicación de escritorio, lo es aún más para un sistema web, dado el nivel de interactividad requerido [23].

6. Conclusiones

Inicialmente, la herramienta fue probada de manera experimental y los resultados obtenidos fueron satisfactorios. Se debe tener en cuenta que, en dichas situaciones, los analistas que utilizaron

PHP4DB tenían un alto nivel de conocimiento del producto. Luego, el Framework maduró para dar soporte a una serie de proyectos (ver Resultados Obtenidos)

Como se presentó oportunamente, la utilización del Framework en los proyectos fue esencial, ya que logró automatizar un gran porcentaje de casos de uso (CU). Se desarrollaron proyectos donde solo un 5% de los Casos de Uso que se presentaron, necesitaron de programación específica. Además, ningún proyecto necesitó, hasta el momento, codificar en particular más de un 30% de los casos de uso definidos.

Es claramente visible el beneficio obtenido con PHP4DB. El tiempo de desarrollo se redujo drásticamente con la utilización de la herramienta, con la consecuente satisfacción del usuario dada la temprana disponibilidad de los productos requeridos. Además, la centralización provista por PHP4DB, ha permitido lograr interfaces homogéneas, facilitando el mantenimiento posterior.

La prototipación también es posible con PHP4DB. Se han realizado experiencias que involucran proyectos de gran envergadura, utilizando PHP4DB como herramienta de prototipado en reuniones tipo Joint Application Development (JAD). Estas reuniones requieren de la generación muy ágil de prototipos que describan las funcionalidades descriptas hasta el momento.

7. Trabajos futuros

La línea de trabajo actual y futura tienen varias aristas:

- **Evolución continúa de la herramienta.** A modo se ejemplo, se está desarrollando actualmente, por ejemplo, una grilla que permita editar/insertar elementos de manera interactiva.
- **Evolución hacia soporte multi-idioma.** Agregar una capa sobre PHP4DB que permita al sistema presentar sus interfases en el idioma deseado. Esta opción está actualmente en prototipación.
- **Evolución hacia un PHP4DB con interfaz personalizable por el usuario final.** Un objetivo planteado, principalmente orientado a darle mayor versatilidad al usuario final del producto, es permitir que la interfaz de interacción sea personalizable. El equipo de desarrollo está discutiendo las mejores opciones para llevar a cabo una solución integral a estas expectativas.
- **Evolución web del asistente.** PHP4DB Assistant, como se mencionó anteriormente, permite la generación automática de repositorios que luego PHP4DB se encargará de procesar. Permitir disponer esta herramienta web marcaría un avance en cuanto a la disponibilidad y productividad del equipo de desarrollo, pudiendo generar nuevos modulos para los sistemas desde cualquier lugar con acceso a internet.

Referencias

[1] Beck K., *Una explicación de la Programación Extrema*, Addison Wesley, 2002.

[2] http://agilemanifesto.org/sign/display.cgi?ms=all

[3] Roger Pressman , Ingeniería de Software. Un enfoque práctico. Mc Graw Hill. 1998.

[4] Capers Jones, Software Assessments, Benchmarks, and Best Practices. Editorial Addison Wesley. ISBN 0201485427, 2000.

[5] Bertone, Pasini, Ramón. Programación Extrema y Calidad. Estudio de Compatibilidad XP – CMM. Cacic 2005 (Congreso Argentino de Ciencias de la Computación) Concordia, Entre Rios. Argentina. Octubre 2005

[6] Bertone, Pasini, Ramón. Programación Extrema y Moprosoft: Son compatibles? Informe técnico III- LIDI. Octubre 2006.

[7] Bertone, Pasini, Ramón, Esponda, Pesado, Mon, Gigante, De María, Estayno, Gestión de Calidad en la Construcción del Software. Unenfoque para PyME's. Cacic 2006 (Congreso Argentino de Ciencias de la Computación) San Luis, Argentina. Octubre 2006

[8] M. Walsamakis, M. Mansutti, R. Bertone, R. Champredonde, Generación Automática de Código a partir del modelo de datos. CACIC2004. La Matanza. Octubre 2004

[9] Ian Sommerville, Ingeniería de Software. 6ta Edición. Addison Wesley 2002.

[10] L. Delía, M. Iglesias, G. Cáseres , H. Ramón, P. Thomas, R. Bertone, Framework for Web Application Agile Development, JCS&T (Journal on Computer Science & Technology) supported by ISTEC, Special Issue on Selected Papers from CACIC 2006, Vol. 7 - No. 1 - March 2007 - ISSN 1666-6038, pág. 86-90, http://journal.info.unlp.edu.ar.

[11] Watts S Humphrey, Introduction to the Team Software Process, Addison-Wesley Professional 1999

[12] http://www.w3.org/

[13] M. Torchiano, et. al.,Overlooked Aspects of COTS-Bases Development, IEEE Software, 2004.

[14] Johannes Sametinger, Software Engineering With Reusable Components, Springer, 2001.

[15] Rick Leander, Building Application Servers, Cambridge University Press, 2000.

[16] http://validator.w3.org/

[17] http://pear.php.net/package/DB/

[18] http://pear.php.net/package/MDB2/

[19] SAIH – Documento de Requerimientos. Hospital Felipe Pelaez. Municipalidad de Florentino Ameghino. IEEE 830. Instituto de Investigación en Informática. 2005, saih-lidi.info.unlp.edu.ar

[20] Framework for Agile Development: Its use in Web Applications for Hospitals, L. Delía, M. Iglesias, G. Cáseres , H. Ramón, P. Thomas, R. Bertone. EJIS (Eletronic Journal of Information Systems), Edition 09, Nº III 2006 special number on Information Systems on Health Care (human) ISSN 1677-3071, 2006, http://www.inf.ufsc.br/resi/

[21] www.dpic.sg.gba.gov.ar dpic-lidi.info.unlp.edu.ar

[22] Sistemas de Software Distribuidos y Bases de Datos Distribuida, WICC 2006, Morón, 1 y 2 de Junio 2006.

[23] Caseres, Delia, Thomas, Ramón, Bertone, Use of asynchronous JavaScript and XML for Comparative Market Analysis, CACIC 2006, San Luis, Octubre de 2006. www.area6team.com.ar.