Informe individual Ariel Andia R. Ingeniero Calidad.

Esta semana se realizaron las siguientes actividades:

Definición de estándares para la programación PHP: De manera más específica se definieron estándares que el grupo de trabajo debe seguir para la programación en PHP (adjunto 1). Estos estándares están basados en los estándares definidos por Zend pero omitiendo convenciones que no se hallaron necesarias para nuestro grupo.

Definición de pauta de calidad para casos de uso: Para llevar un mejor control de calidad, se han estado definiendo pautas de calidad para las distintas actividades del grupo (individual o grupal), esta semana correspondió la de casos de uso (adjunto 2). La pauta para los casos de uso contempla en una parte los elementos a considerar para hacer casos de uso de calidad y en la otra parte de la pauta se definen algunos tópicos para la revisión de los casos de uso.

Actividades para la entrega grupal: Para el avance de esta semana se trabajó en grupo en la mayoría de las actividades, pero también se definieron algunas actividades individuales, en las cuales se asignó la redacción de la introducción del informe escrito y la definición de las interfaces externas de salida al Ingeniero de Calidad.

Adjunto 1: Estándares de programación PHP.

Indentación: Serán compuestas por 4 espacios y no por tabulación.

Tamaño máximo de líneas: La longitud recomendable para una línea de código es de 80 caracteres. Esto quiere decir que el desarrollador al programar debe intentar mantener su línea de código por debajo de los 80 caracteres, aunque se puede sobrepasar siempre y cuando sea necesaria. El tamaño máximo de cualquier línea de código PHP es de 120 caracteres.

Clases: Los nombres de clases pueden contener sólo caracteres alfanuméricos. Los números están permitidos en los nombres de clase, pero desaconsejados en la mayoría de casos. Para la notación se utilizará camelCase.

Nombre de archivos: Los nombres de los archivos no deben llevar espacios.

Funciones y Métodos: Los nombres de funciones pueden contener únicamente caracteres alfanuméricos. Los guiones bajos (_) no están permitidos. Los números están permitidos en los nombres de función pero no se aconseja en la mayoría de los casos.

Los nombres de funciones deben empezar siempre con una letra minúscula. Cuando un nombre de función consiste en más de una palabra, la primera letra de cada nueva palabra debe estar en mayúsculas.

Los nombres de función deben ser lo suficientemente elocuentes como para describir su propósito y comportamiento.

Para la programación orientada a objetos, los métodos de acceso para las instancias o variables estáticas deben ir antepuestos con un "get" o un "set".

Variable: Los nombres de variables pueden contener caracteres alfanuméricos. Las barras bajas (_) no están permitidas. Los números están permitidos en los nombres de variable pero no se aconseja en la mayoría de los casos.

Demarcación de código PHP: No se permite el uso de etiquetas cortar.

Array asociativos: Al declarar arrays asociativos con la construcción array, se recomienda partir la declaración en múltiples líneas. En este caso, cada línea sucesiva debe ser tabuladas con cuatro espacios de forma que tanto las llaves como los valores están alineados:

```
$sampleArray = array('firstKey'  => 'firstValue',
                     'secondKey' => 'secondValue');
```

También se permite hacer un salto de línea para el primer elemento del array:

```
$sampleArray = array(
    'firstKey'  => 'firstValue',
    'secondKey' => 'secondValue'
```

```
);
```

Declaración de clases:

Las Clases deben ser nombradas de acuerdo a las convenciones definidas anteriormente. La llave "{" deberá escribirse siempre en la línea debajo del nombre de la clase ("one true brace"). Todo el código contenido en una clase debe ser separado con cuatro espacios. Únicamente una clase está permitida por archivo PHP.

Las clases que extiendan otras clases o interfaces deberían declarar sus dependencias en la misma línea siempre que sea posible.

```
class SampleClass extends FooAbstract implements BarInterface
{
}
```

Si como resultado de esas declaraciones, la longitud de la línea excede la longitud del Tamaño máximo de línea , se debe romper la línea antes de la palabra clave "extends" y / o "implements" e indentarlo con un nivel de indentación (4 espacios).

```
class SampleClass
    extends FooAbstract
    implements BarInterface
{
……
}
```

Adjunto 2: Pauta de Calidad para Casos de Uso.

La siguiente pauta actúa a modo de guía y no limita a los encargados del análisis de casos de uso a considerar otros puntos de evaluación además de los que en el presente se mencionan

Elementos a considerar:

“Unicidad”: Que los casos de uso procesen una acción específica.

Modularidad: Que el caso de uso desarrolle acciones relacionadas entre sí, esto pude ser visto como un caso de uso con <include> y/o <extends>.

Actor: Que espere una respuesta de los casos de uso con el cual interactúa.

Valor del Caso de Uso: Que la acción que ejecute el caso de uso tenga un valor de resultado importante para el Actor.

Requisito principal: Requisito principal por el cual el caso de uso es definido.

Guía para la revisión.

Análisis de Actores: Analizar los Actores desde la perspectiva de las siguientes interrogantes:

-¿Qué espera el Actor del sistema? (Basándose en la toma de requerimientos, que información y/o respuesta debe entregar el sistema sin considerar conceptos de diseño, velocidad, modelo de datos, etc.).

-¿Qué respuesta espera el Actor del caso de uso con el que interactúa?

Tomar el rol de Actor: Sin pensar en que interfaz o diseño, tomar el lugar del Actor e interactuar con el caso de uso y analizar ¿Cómo es la interacción con él sistema?, ¿Qué información le entrego al caso de uso?, ¿Qué resultado espero del caso de uso?, ¿La acción del caso de uso tiene suficiente valor como para ser definido como tal?, ¿Qué casos de uso pueden o faltan en él diagrama?, y otros puntos que el grupo de análisis considere importantes dependiendo el problema. Se recomienda hacer esta revisión de forma individual, tomando apuntes y luego presentarlo al grupo para revisar dichas anotaciones, después de esto hacer una revisión grupal.

Analizar casos de uso: Analizar la “unicidad”, modularidad, valor del caso de uso, el cumplimiento del requisito principal, y otros puntos de análisis que estime necesario el grupo, después de esto evaluar el caso de uso estimando si debe ser eliminado del diagrama o modificado.

Revisar de manera global el diagrama de casos de uso: Después de todo lo anterior revisar de forma general para detectar falencias y realizar convenciones si fuese necesario.