**Capítulo 5**

**Ilustración DE LA PROPUESTA**

**5.1 INTRODUCCIÓN**

En este capítulo se describirá el proceso llevado a cabo para realizar una validación preliminar de *MoWebA* con extensiones *RIA*. La validación consiste en comparar la capa de presentación que se puede generar con *MoWebA* con extensiones *RIA* con respecto a la que se puede generar con el mismo enfoque pero sin extensiones. La comparativa entre los enfoques tomados se enmarca contextualmente en el dominio de las aplicaciones Web, precisamente con la obtención de datos analíticos, en base a la implementación de un sistema de administración de personas o *Person Manager*.

Existen diversos métodos empíricos para llevar a cabo validaciones formales sobre algún fenómeno en particular, entre los que se pueden citar a los experimentos, las encuestas y los casos de estudio. Es común en el campo de la ingeniería de software emplear a los casos de estudio como método de validación, debido a su flexibilidad y a la posibilidad de tener un mejor control sobre las variables de medición, a costas de un mayor esfuerzo en la interpretación de los resultados obtenidos [<c2012>].

Según Runeson [<p2012>], un caso de estudio es llevado a cabo para investigar una sola entidad o un fenómeno en su contexto de la vida real, en un espacio de tiempo específico. Típicamente el fenómeno puede ser difícil de distinguir de su ambiente y el investigador debe colectar información detallada sobre un proyecto durante un periodo sostenido de tiempo. Durante la realización del caso de estudio, una variedad de procedimientos de colección de datos y perspectivas de análisis deben aplicarse.

Atendiendo a la anterior definición, no siempre es posible realizar un caso de estudio. Una alternativa es la ilustración, que a pesar de no ser un método de validación formal, sirve para presentar a una audiencia cómo funciona un artefacto y cómo este puede resolver un *toy problem* en particular, permitiendo sugerir una conclusión intuitiva [<r2014>].

En este capítulo se utiliza una ilustración para validar preliminarmente las extensiones *RIA* de *MoWebA* por medio de la resolución de un *toy problem* denominado *Person Manager*. Para brindar mayor formalidad a la ilustración, la misma se realizó siguiendo las guías propuestas por Runeson para la definición y análisis de casos de estudio.

**5.2 LA ILUSTRACIÓN**

**5.2.1 Objetivos**

Esta ilustración se presenta con la intención de ofrecer un análisis crítico de las extensiones *RIA* llevadas a cabo con el enfoque *MoWebA* desde el punto de vista de las interfaces enriquecidas. La propuesta de extensión se basa principalmente en proveer a *MoWebA* de características enriquecidas a nivel de la interfaz de usuario, que le permitirán mantenerse vigente con respecto a las nuevas tendencias de las aplicaciones Web de hoy en día, que demandan una mayor interactividad y riqueza en las interfaces de usuario.

También se busca obtener datos lo suficientemente reveladores que permitan intuir que la propuesta de extensión a nivel de la capa de presentación para el lado del cliente llevada a cabo al enfoque *MoWebA*, ofrece cobertura a algunas de las diversas características que contemplan las *RIA* analizadas en el capítulo 2. Puntualmente, estas características abarcan a la lógica de negocios en el lado del cliente, específicamente a las validaciones locales de campos en un formulario, y a las presentaciones enriquecidas que contemplan a ciertos eventos en el lado del cliente, *widgets* interactivos y el paradigma de una sola página o *single page paradigm*. El objetivo es analizar estas características por medio de la resolución de un *toy problem* denominado *Person Manager*. El *Person Manager* es una aplicación Web que contiene en sus especificaciones funcionales características de las *RIA* y resulta lo suficientemente expresiva para ilustrar la propuesta de extensión.

Analizar la productividad en el proceso de modelado de los *PIM* en *MoWebA* con *RIA*, también es uno de los objetivos de esta ilustración, con lo que se intenta comparar los tiempos de modelado para ambos enfoques, como así también las veces que es necesario generar el código fuente de la aplicación a partir de la corrección de los *PIM*, hasta obtener la interfaz final. Finalmente, se busca comparar la cantidad de líneas de código generadas de manera automática en ambos enfoques.

**5.2.1.1 Objetivos específicos**

Comparar a *MoWebA* sin *RIA* y a *MoWebA* con extensiones *RIA*, con respecto al tiempo de modelado y a la cantidad de generaciones y refinamientos a nivel de modelado que se deben hacer hasta obtener una interfaz final satisfactoria.

Verificar si *MoWebA* con extensiones *RIA* ofrece ventajas sobre *MoWebA* sin *RIA* con respecto a las presentaciones enriquecidas y con respecto a la lógica en el lado del cliente.

Analizar la cantidad de líneas de código generadas de manera automática a partir de los modelos *PIM*, para ambos enfoques.

**5.2.2 Preguntas de investigación**

A partir de los objetivos anteriormente presentados, surgen las siguientes preguntas de investigación para esta ilustración:

PI1: ¿Consume una mayor cantidad de tiempo modelar la aplicación aplicando *MoWebA* con *RIA* que *MoWebA* sin *RIA*?

PI2: ¿Para cuál de los enfoques es necesaria una mayor cantidad de generaciones de código para obtener la interfaz de usuario final?

PI3: Desde el punto de vista de las presentaciones enriquecidas, ¿qué ventajas aportan las características *RIA* presentes en la aplicación implementada con *MoWebA* con *RIA* con respecto a *MoWebA* sin *RIA*?

PI4: Desde el punto de vista de la lógica de negocios en el lado del cliente, ¿qué ventajas aportan las características *RIA* presentes en la aplicación implementada con *MoWebA* con *RIA* con respecto a *MoWebA* sin *RIA*?

PI5: Para cada una de las vistas del *Person Manager,* ¿qué cantidad de líneas de código para la interfaz de usuario se pudieron generar de manera automática a partir de los modelos, en cada uno de los enfoques implementados?

**5.2.3 El caso y las unidades de análisis**

La ilustración de la propuesta está basada en un sistema de administración de personas (*Person Manager*) en el dominio de las aplicaciones Web, que fue elegido entre varias otras opciones debido a que sus requerimientos funcionales ofrecen la posibilidad de representar a todas las características *RIA* que han sido agregadas a la metodología *MoWebA*, de una manera clara y sencilla.

En el Anexo 1 del libro de tesis se presenta la descripción del sistema *Person Manager* de una manera general, y se agregan algunos requerimientos adicionales, que son precisamente, requerimientos *RIA*. El *Person Manager* está basado en el trabajo de Gharavi [].



Figura 1 Ilustración del sistema *Person Manager* implementado con *MoWebA* desde dos enfoques distintos

El caso fue analizado desde dos unidades de análisis. La primera unidad de análisis se refiere a la implementación de la capa de presentación del *Person Manager* con *MoWebA* sin *RIA*. La segunda unidad de análisis se refiere a la implementación de la misma capa de presentación del caso estudiado, con la nueva propuesta de extensión *RIA* a *MoWebA*. Por lo tanto, se trata de una ilustración embebida, ya que se cuenta con más de una unidad de análisis para un mismo caso, como se puede apreciar en la Figura 1.

Las unidades de análisis representan a los dos enfoques implementados con *MoWebA* (con o sin *RIA*) que servirán para obtener respuestas a las preguntas de investigación.

**5.2.4 Variables**

Las variables de medición necesarias para responder las preguntas de investigación PI1, PI2 y PI5 se definen a continuación:

i = {1, 2, 3, 4}, en donde **1** corresponde a la vista *Agregar Persona*, **2** corresponde a la vista *Listar Personas*, **3** corresponde a la vista *Remover Persona y* **4** corresponde a la vista general que es común a *Agregar Persona*, *Listar Personas* y *Remover Persona*.

j = {a, b}, en donde **a** es el enfoque *MoWebA* sin *RIA* y **b** es el enfoque *MoWebA* con *RIA*.

ti,j = tiempo total en minutos, empleado para el modelado de la vista i empleando el enfoque j.

ni,j = número total de generaciones de código para obtener la vista i utilizando el enfoque j.

Ta = ti=1,a + ti=2,a + ti=3,a , es el tiempo total en minutos, empleado para modelar todas las vistas i para j = a o *MoWebA* sin *RIA*.

Tb = ti=1,b + ti=2,b + ti=3,b , es el tiempo total empleado para modelar todas las vistas i para j = b o *MoWebA* con *RIA*.

Na = ni=1,a + ni=2,a + ni=3,a , es el número total de generaciones de código necesarias para obtener la interfaz de usuario final de todas las vistas i para j = a o *MoWebA* sin *RIA*.

Nb = ni=1,b + ni=2,b + ni=3,b , es el número total de generaciones de código necesarias para obtener la interfaz de usuario final de todas las vistas i para j = b o *MoWebA* con *RIA*.

lcai,j = total de líneas de código generadas de manera automática a partir de los modelos *PIM* de entrada para la vista i empleando el enfoque j.

lcmi,j = total de líneas implementadas manualmente para la vista i empleando el enfoque j.

LCAa = lcai=1,a + lcai=2,a + lcai=3,a + lcai=34,a , total de líneas de código generadas de manera automática a partir de los modelos *PIM* de entrada empleando el enfoque a.

LCMa = lcmi=1,a + lcmi=2,a + lcmi=3,a + lcmi=4,a , total de líneas de código implementadas manualmente empleando el enfoque a.

LCAb = lcai=1,b + lcai=2,b + lcai=3,b + lcai=4,b , total de líneas de código generadas de manera automática a partir de los modelos *PIM* de entrada empleando el enfoque b.

LCMb = lcmi=1,b + lcmi=2,b + lcmi=3,b+ lcmi=4,b , total de líneas de código implementadas manualmente empleando el enfoque b.

Las preguntas de investigación PI3 y PI4 no necesitan de la definición de variables para obtener sus respuestas, sino más bien, por medio de capturas de pantalla comparativas y descripciones detalladas del *Person Manager* para ambos enfoques se podrá dar respuesta a las mismas.

Una generación de código (equivalente a compilar la aplicación) es llevada a cabo para obtener el código fuente de la interfaz final a partir de los modelos de entrada. En cada una de las generaciones, el código fuente se va refinando a partir de la modificación de los modelos *PIM* de entrada.

**5.2.5 Métodos de colección de datos**

La ilustración de esta propuesta se realizó utilizando a un individuo como población (el autor de la tesis), tanto para la implementación de las diferentes unidades de análisis, como para la recolección de los datos necesarios para responder PI1, PI2, PI3, PI4 y PI5, por lo tanto no se llevaron a cabo entrevistas ni encuestas que son empleadas de manera común en casos de estudio y experimentos.

Teniendo en cuenta este hecho particular, los datos necesarios para calcular las variables de respuesta fueron recolectados a medida que las unidades funcionales de la aplicación (las vistas del *Person Manager)* eran implementadas. Primeramente se recolectó toda la información correspondiente a *MoWebA* sin *RIA* y luego se procedió a la colección de los datos de *MoWebA* con *RIA*. Para cada uno de los enfoques, primeramente se recolectaron todos los datos correspondientes a una vista en particular hasta la conclusión de ésta. Seguidamente se pasaba a la siguiente vista y se recababan los datos correspondientes y así sucesivamente.

Para la recolección de los tiempos de modelado para cada una de las vistas del *Person Manager* con los distintos enfoques, se utilizó la , en donde se presentan las variables que deben ser colectadas.

|  |  |  |
| --- | --- | --- |
| **Tiempos en minutos de modelado y generación de código para cada i,j** | ***MoWebA* sin *RIA* (j = a)** | ***MoWebA* con *RIA* (j = b)** |
| **Agregar persona (i = 1)** | **ti=1,a** | **ti=1,b** |
| **Listar personas (i = 2)** | **ti=2,a** | **ti=2,b** |
| **Remover persona (i = 3)** | **ti=3,a** | **ti=3,b** |
| **Totales** | **Tew}r+ = ti=1,a+ ti=2,a + ti=3,a** | **Tb = ti=1,b+ ti=2,b + ti=3,b** |

Tabla 1 Tiempo de modelado para cada uno de las vistas del *Person Manager* en cado uno de los enfoques.

Para recolectar el número de generaciones de código para cada una de las vistas del *Person Manager* en los distintos enfoques, se utilizó la , en donde se presentan las variables que deben ser colectadas.

|  |  |  |
| --- | --- | --- |
| **Número de generaciones de código para cada i,j** | ***MoWebA* sin *RIA* (j = a)** | ***MoWebA* con *RIA* (j = b)** |
| **Agregar persona (i = 1)** | ni=1,a | ni=1,b |
| **Listar personas (i = 2)** | ni=2,a | ni=2,b |
| **Remover persona (i = 3)** | ni=3,a | ni=3,b |
| **Totales** | Na = ni=1,a+ ni=2,a + ni=3,a | Nb = ni=1,b+ ni=2,b + ni=3,b |

Tabla 2 Numero de generaciones de código para cada una de las vistas del *Person Manager* en cada uno de los enfoques

Finalmente, para llevar a cabo un análisis de las líneas de código generadas automáticamente a partir de los modelos de entrada se utilizó la Tabla 3.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Líneas de código / Vistas de la aplicación** | ***MoWebA* sin *RIA* (j = a)** | | | ***MoWebA* con *RIA* (j = b)** | | |
| **Líneas de código automáticas** | **Líneas de código manuales** | **Totales** | **Líneas de código automáticas** | **Líneas de código manuales** | **Totales** |
| **Agregar persona (i = 1)** | lcai=1,a | lcmi=1,a | lcai=1,a + lcmi=1,a | lcai=1,b | lcmi=1,b | lcai=1,b + lcmi=1,b |
| **Listar personas (i = 2)** | lcai=2,a | lcmi=2,a | lcai=2,a + lcmi=2,a | lcai=2,b | lcai=2,b | lcai=2,b+ lcai=2,b |
| **Remover persona (i = 3)** | lcai=3,a | lcmi=3,a | lcai=3,a + lcmi=3,a | lcai=3,b | lcmi=3,b | lcai=3,b + lcmi=3,b |
| **Estructura y código común para todas las vistas (cabecera, estructura y pie de página (i = 4)** | lcai=4,a | lcmi=4,a | lcai=4,a + lcmi=4,a | lcai=4,b | lcmi=4,b | lcai=4,b + lcmi=4,b |
| **Totales** | LCAa = lcai=1,a+ lcai=2,a + lcai=3,a | LCMa = lcmi=1,a+ lcmi=2,a + lcmi=3,a | LCAa + LCMa = | LCAb = lcai=1,b+ lcai=2,b + lcai=3,b | LCMb = lcmi=1,b+ lcmi=2,b + lcmi=3,b | LCAb + LCMb |

Tabla 3 Tabla utilizada para la comparación de líneas de código

**5.2.6 Métodos de análisis de los datos**

Para cada una de las variables definidas en la sección 5.2.4 sólo se va a tener una medición por cada variable y para cada uno de los enfoques, y por lo tanto no se va a poder realizar un análisis estadístico formal, sino más bien, se van a comparar los valores obtenidos en cada una de las implementaciones del *Person Manager*. Estos valores recolectados servirán para reportar las respuestas a cada una de las preguntas de investigación.

**5.2.7 Amenazas a la validez de los datos**

Como se mencionó anteriormente, se llevó a cabo una ilustración comparativa, en la cual se optó por un *toy problem* en el contexto de las aplicaciones Web, el cual fue implementado por el autor del trabajo. Se optó por el *Person Manager,* ya que es posible representar en él las nuevas características extendidas a *MoWebA* de manera clara y concisa. Para que los datos de las mediciones obtenidas sean lo más fidedignos posibles, se intentó mitigar las amenazas a la validez de los datos llevando a cabo ciertas acciones preventivas.

Con respecto a las medidas tomadas en esta ilustración para minimizar los factores de riesgo en la validez de los datos obtenidos, se ha tenido en consideración los siguientes puntos:

1. Primeramente el autor del trabajo recibió entrenamiento sobre la metodología *MoWebA* en su versión original (sin *RIA*) y trabajó con diversos ejemplos distintos al de esta ilustración para modelar y generar aplicaciones. Seguidamente extendió el metamodelo original junto a los perfiles e implementó el nuevo generador de código. Entonces no hizo falta entrenamiento previo para el modelado de los *PIM* en ambos enfoques del *Person Manager.* Previamente se llevó a cabo un *testing* exhaustivo de la herramienta de generación de código para evitar problemas.
2. Se empleó completamente el enfoque *MoWebA* sin *RIA* hasta obtener la interfaz final del *Person Manager* y luego el enfoque *MoWebA* con *RIA* de igual manera con una semana de diferencia entre la implementación de cada enfoque. Se tomó esta decisión, ya que si se implementa una vista con el enfoque *MoWebA* sin *RIA* y luego la misma vista con el enfoque *MoWebA* con *RIA*, se puede obtener demasiada familiaridad con el modelado de la última de las vistas implementadas, y por ende no sería tan realista la medición de los tiempos de modelado en el enfoque *MoWebA* con *RIA*, ya que estos podrían reducirse.Sin embargo, no pudo atenuarse el hecho de tener bastante familiaridad con *MoWebA* con *RIA* al implementar una vista luego de haber implementado con una semana de anterioridad la misma vista con *MoWebA* sin *RIA*.
3. A medida que se iban implementando cada una de las vistas en un enfoque particular, se iban recabando los datos correspondientes para las variables de medición de esa vista. Se pasaba a implementar la siguiente vista de la aplicación, una vez culminada en su totalidad la vista actual con todos los datos analíticos recabados. Se tomó esta decisión para hacer una medición más precisa de los tiempos y evitar estimaciones arbitrarias.
4. Para la medición de los tiempos de modelado de los *PIM,* se designó a una persona ajena al proyecto que tomaba los valores correspondientes desde el inicio del *PIM* hasta su fin, para una mayor objetividad en las mediciones.
5. Para una mayor calidad en el análisis de líneas de código se optó por el uso de la herramienta de uso libre CLOC[[1]](#footnote-1). Con esta herramienta se evitó el tener que llevar a cabo el conteo de líneas de código manualmente.

En cierto sentido las medidas tomadas sirvieron para mitigar algunas amenazas posibles a la validez de los datos recabados, pero no se han podido eliminar todas las amenazas. Teniendo en cuenta que el autor del trabajo implementó las unidades de análisis y a la vez recabó los datos analíticos, se trató de llevar adelante cada paso con la mayor transparencia y objetividad posibles, para que los resultados obtenidos sean fidedignos y de valor. Sin embargo, se reconoce que esto no basta para otorgar la suficiente formalidad a los datos como es esperado en casos de estudio o experimentos. Es bajo esta circunstancia que se decidió llevar a cabo una ilustración de la propuesta de extensión y no un caso de estudio específicamente, ya que los resultados y conclusiones obtenidos deben ser considerados en el contexto en el que fueron recabados.

**5.2.8 Colección de los datos**

En esta sección se presentan los datos recabados para responder a las 5 preguntas de investigación. La información correspondiente a las PI1, PI2 y PI5 se encuentra en la Tabla 4, Tabla 5 y Tabla 6, respectivamente. Estas tablas fueron presentadas y descriptas previamente, en la sección 5.2.5.

|  |  |  |
| --- | --- | --- |
| **Tiempos en minutos de modelado y generación de código para cada i,j** | ***MoWebA* sin *RIA* (j = a)** | ***MoWebA* con *RIA* (j = b)** |
| **Agregar persona (i = 1)** | 50 minutos | 56 minutos |
| **Listar persona (i = 2)** | 27 minutos | 28 minutos |
| **Remover persona (i = 3)** | 29 minutos | 30 minutos |
| **Totales** | 106 minutos | 114 minutos |

Tabla 4 Tiempo de modelado para cada uno de los enfoques

|  |  |  |
| --- | --- | --- |
| **Número de generaciones de código para cada i,j** | ***MoWebA* sin *RIA* (j = a)** | ***MoWebA* con *RIA* (j = b)** |
| **Agregar persona (i = 1)** | 3 | 4 |
| **Listar persona (i = 2)** | 1 | 1 |
| **Remover persona (i = 3)** | 2 | 3 |
| **Totales** | 6 | 8 |

Tabla 5 Cantidad de generaciones de código para cada uno de los enfoques para la obtención de la interfaz final

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Líneas de código/ Vistas de la aplicación** | ***MoWebA* sin *RIA* (j = a)** | | | ***MoWebA* con *RIA* (j = b)** | | |
| **Líneas de código automáticas** | **Líneas de código manuales** | **Totales** | **Líneas de código automáticas** | **Líneas de código manuales** | **Totales** |
| **Agregar persona (i = 1)** | 51 | 56 | 107 | 135 | 56 | 191 |
| **Mostrar persona (i = 2)** | 1 | 45 | 46 | 3 | 45 | 48 |
| **Remover persona (i = 3)** | 7 | 27 | 34 | 31 | 27 | 58 |
| **Estructura y código común para todas las vistas (cabecera, estructura y pie de página)** | 67 | 10 | 77 | 52 | 38 | 90 |
| **Totales** | 126 | 138 | 264 | 221 | 166 | 387 |

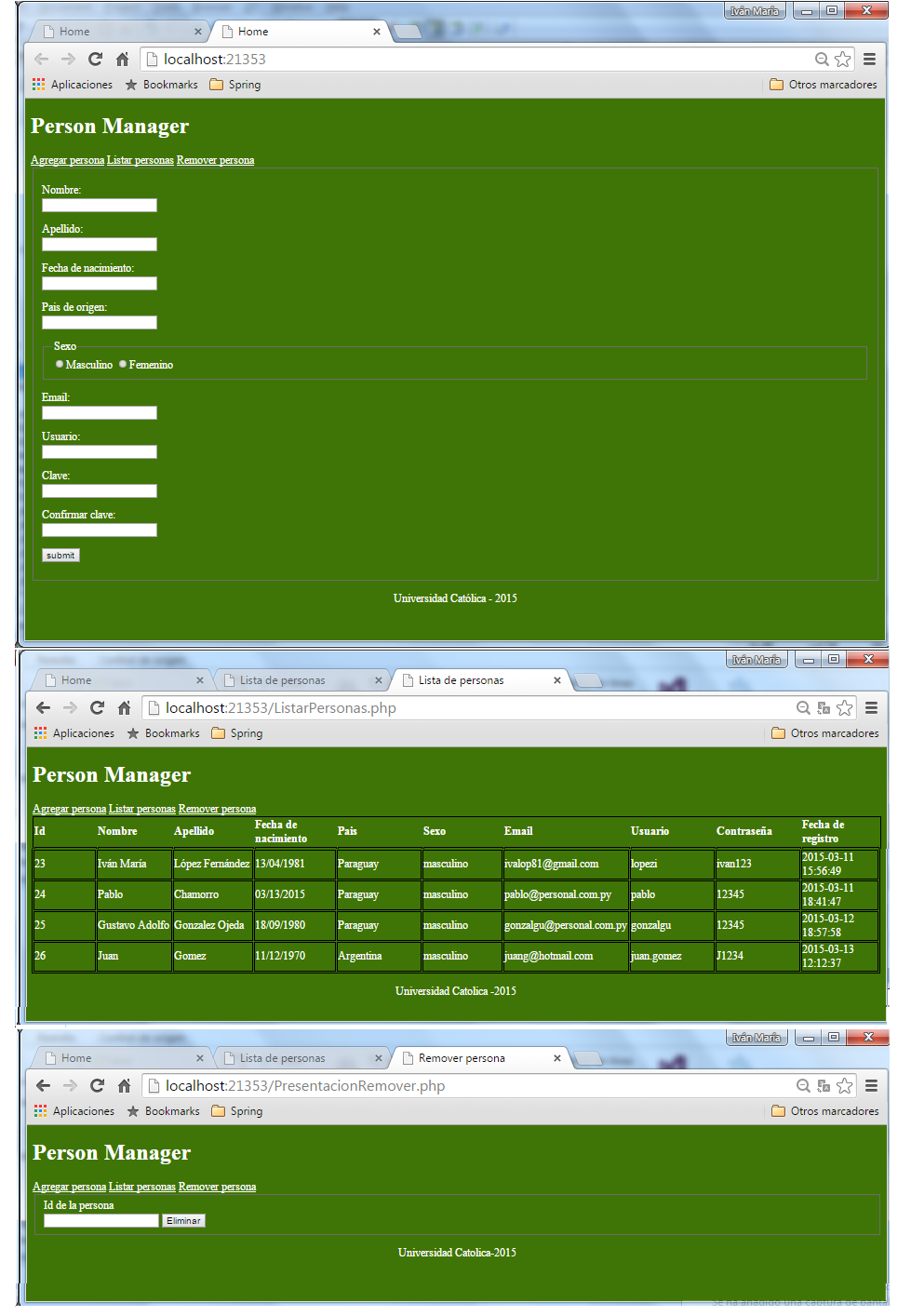
Tabla 6 Líneas de código para ambos enfoques del *Person Manager*

Figura 2 Vistas del Person Manager sin RIA

Figura 3 Vistas del *Person Manager* sin *RIA*

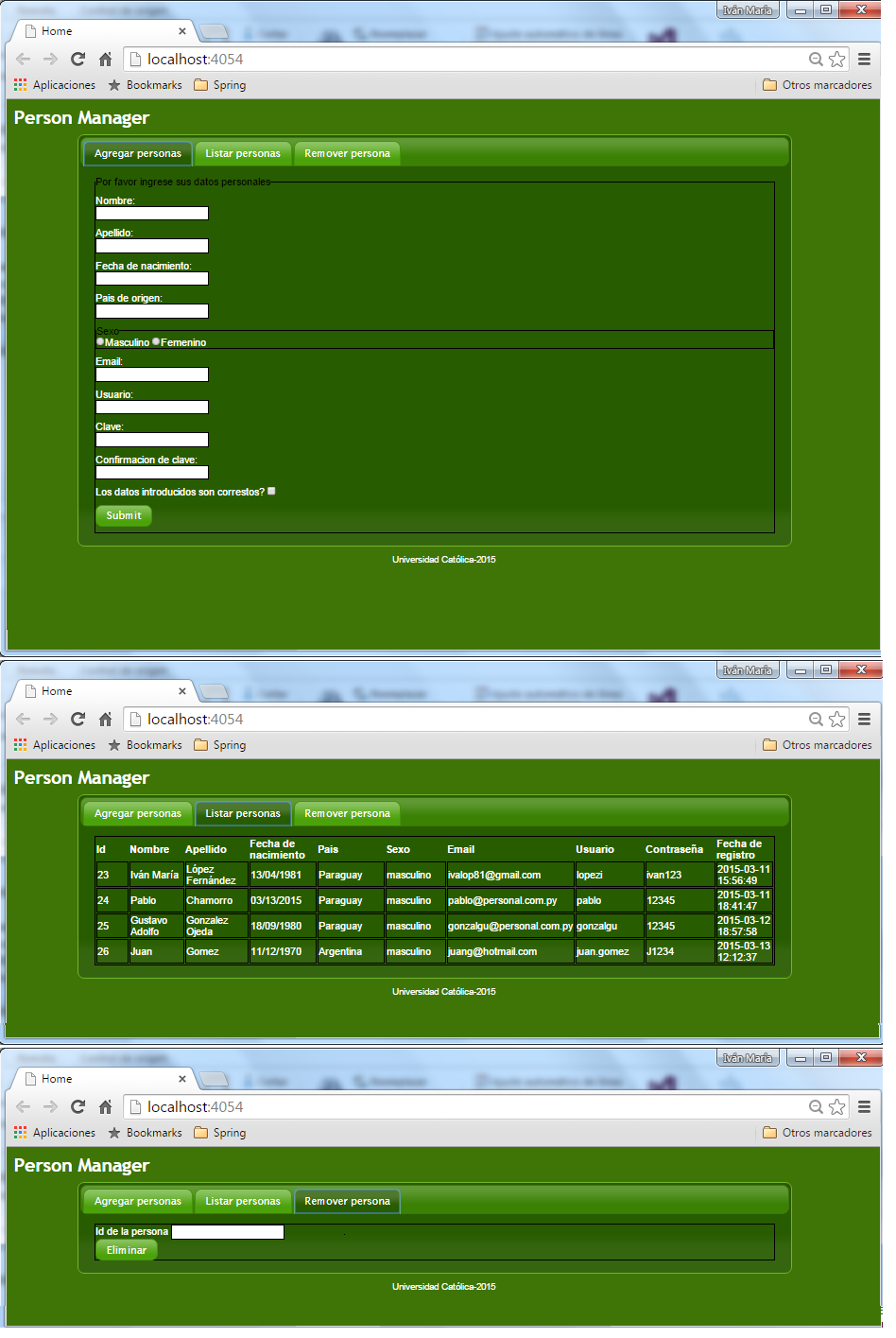
****

Figura 3 Vistas del *Person Manager* con *RIA*

**5.2.9 Análisis e interpretación de los resultados**

Aquí se analizan los resultados de las mediciones realizadas en la sección anterior. En base a los datos obtenidos, se responden las preguntas de investigación.

**PI1: ¿Consume una mayor cantidad de tiempo modelar la aplicación aplicando *MoWebA* con *RIA* que *MoWebA* sin *RIA*?**

Según puede apreciarse en la , a nivel general existe una diferencia de 8 minutos entre los enfoques aplicados al *Person Manager*. El enfoque *MoWebA* con *RIA* conlleva definir, en los modelos *PIM*, diversas características particulares, expresadas por medio de valores etiquetados, para cada uno de los elementos *RIA* presentes. Para el enfoque *MoWebA* sin *RIA* no ocurre de igual forma, ya que son muchas menos las propiedades que pueden definirse sobre cada uno de los elementos que forman parte de los perfiles de Contenido y Estructura. Por lo tanto, al tener que definir más valores etiquetados para el caso *MoWebA* con *RIA,* se incurre en una mayor cantidad de tiempo en el modelado.

Particularmente, para la vista *Agregar Persona,* que corresponde a la vista con mayor número de requerimientos *RIA*, existe la mayor diferencia de tiempo en el modelado (6 minutos más en el enfoque con *RIA* que su contraparte sin *RIA*) y resulta natural este hecho debido a la cantidad de valores etiquetados que hay que establecer en casi todos los elementos del formulario. En esta vista se utilizan *richDatePicker*, *richAutoSuggest*, *richToolTips* y validaciones de diversos campos.

En el caso de la vista *Listar Personas,* no hubo mucha diferencia en el tiempo de modelado (tan solo 1 minuto más en el enfoque con *RIA*), lo que no resulta muy relevante, ya que el *PIM* de esta vista es el mismo para ambos enfoques al no tener características *RIA*.

Finalmente, la vista *Remover Persona* en el enfoque *MoWebA* con *RIA* llevó 1 minuto más de tiempo de modelado que *MoWebA* sin *RIA*. Aquí la diferencia es mínima y esto se debe a que el formulario en cuestión tan solo tiene un campo de entrada y un botón *submit,* pero para el enfoque con *RIA*, hay que definir más valores etiquetados que en su contraparte sin *RIA*, ya que es necesario efectuar validaciones de campo. Es importante que la implementación MoWebA sin *RIA* también podría llevar a cabo validaciones sobre los campos de entrada, pero estas validaciones deberían implementarse de manera manual.

Desde el punto de vista práctico, que *MoWebA* con *RIA* tarde 8 minutos más que su par sin *RIA* no es una limitante demasiado condicionante, teniendo en cuenta que este tiempo extra de modelado permite a la interfaz de la aplicación *Person Manager* enriquecerse notablemente.

**PI2: ¿Para cuál de los enfoques es necesaria una mayor cantidad de generaciones de código para obtener la interfaz de usuario final?**

Se pudo apreciar que *MoWebA* con *RIA* deparó en una mayor cantidad de generaciones de código para obtener la interfaz *RIA* final, precisamente un 20% más con respecto a la implementación llevada a cabo con *MoWebA* sin *RIA*. Analizando las generaciones de cada una de las vistas del *Person Manager* de la , se puede notar que la vista que tuvo que generarse una mayor cantidad de veces fue la vista Agregar Persona, y este dato resulta concordante con lo que puede intuirse preliminarmente, ya que esta vista es la que contiene la mayor cantidad de requerimientos de interfazy por ende existe una mayor probabilidad de cometer fallos en el modelado, lo que incurre en una mayor cantidad de veces que la aplicación debe generarse hasta su depuración final. Para la vista Remover Persona se incurrió en un número mínimamente superior de generaciones de código implementando el *MoWebA* con *RIA*, precisamente un 10% más que *MoWebA* sin *RIA*, pudiendo deberse también, a que la vista con *MoWebA* con *RIA* contiene un número mayor de detalles en sus *PIM,* precisamente para las validaciones en el campo de eliminación de personas, a diferencia de*MoWebA* sin *RIA*. También el número de generaciones disminuyó en ambos métodos con respecto a la vista Agregar Persona. En la vista Listar Personas, se tuvo la mínima cantidad de generaciones de código en ambos métodos aplicados, debido a que gran parte de ella fue implementada de manera manual, ya que la propuesta de extensión no contempla el acceso al modelo de datos.

De los resultados presentados puede intuirse que a mayor requerimientos de interfaz, se requiere una mayor cantidad de generaciones de código para ambas metodologías aplicadas para ir depurando la aplicación, con un leve incremento en el caso de *MoWebA* con *RIA* y esto podría deberse a que dada una mayor cantidad de detalles a especificar en los modelos de entrada de la aplicación, existe una mayor posibilidad de cometer fallos.

**PI3: Desde el punto de vista de las presentaciones enriquecidas, ¿qué ventajas aportan las características *RIA* presentes en la aplicación implementada con *MoWebA* con *RIA* con respecto a *MoWebA* sin *RIA*?**

Son diversos los aportes de *MoWebA* con *RIA* con respecto a *MoWebA* sin *RIA* para el *Person Manager*. Las comparativas de las vistas generales de cada uno de los enfoques pueden apreciarse en la y Figura 3. A continuación se describen algunas ventajas.

***Apariencia de una aplicación single page***

Cada una de las páginas que forman parte de la aplicación *Person Manager* implementada con MoWebA con *RIA*, son equivalentes a las pestañas pertenecientes a un *widget richTab*. Por lo tanto, cuando se navega en la aplicación se tiene la sensación de que se trata de una aplicación de escritorio, ya que se puede recorrer cada una de las pestañas sin necesidad de un refrescado de página, teniendo toda la información de manera local en una sola página. Esta característica mejora la interactividad con el usuario de la aplicación y el *look and feel* de la misma. En la implementación llevada a cabo con MoWebA sin *RIA*, cada una de las páginas de la aplicación está representada por un enlace, y por ende, cada vez que se visita una página de la aplicación, un refrescado total de página se lleva a cabo, perdiéndose de esta forma el concepto de *single page*.

Con la extensión *RichTab* presente en el *Person Manager* con el enfoque *RIA,* es posible encapsular distintos elementos de interfaz presentes en *MoWebA*, tales como entradas de texto, enlaces, botones, textos, hipervínculos, formularios y tablas, como así también elementos enriquecidos que son parte de la extensión *RIA* propuesta a *MoWebA*, como los *RichDatePicker*, *RichToolTip*, *RichAutoSuggest*. La extensión de validación de campostambién puede ser utilizada dentro de un *RichTab*, para la validación de los campos en un formulario. La posibilidad de encapsular muchos elementos de interfaz dentro de cada una de las pestañas que forman parte de un *RichTab* resulta ventajosa con respecto a la optimización espacial de los elementos dentro de las páginas. Estas mismas ventajas relacionadas al *RichTab*, también pueden ser aprovechadas al utilizar la extensión *RichAccordion.*

***Widgets interactivos en la interfaz de usuario***

Todos estos elementos enriquecidos que forman parte del *Person Manager* se han presentado en el capítulo 4 del libro en la Figura 12.

***RichDatepicker***

El *richDatePicker* permite al *Person Manager* implementado con el enfoque *MoWebA* con *RIA* desplegar de una manera ágil e interactiva un calendario debajo de la entrada textual que corresponde al campo fecha de nacimiento. Este calendario interactivo (ver Figura 12-c del capítulo 4) ofrece la posibilidad de navegar por los distintos meses del año actual, con las flechas indicadoras izquierda y derecha, como así también, permite seleccionar un mes en particular desplegando una lista de meses. Con respecto a los años del calendario, es posible definir un rango de años que podrán seleccionarse de igual manera de una lista desplegable.

Este *widget* resulta de gran ayuda a los usuarios finales, ya que gráfica e intuitivamente permite seleccionar una fecha, evitando a estos cometer errores innecesarios al digitar una fecha en un formato dado y optimizando su tiempo de interacción con las páginas Web. Para el enfoque *MoWebA* sin *RIA*, el campo fecha de nacimiento es solamente un campo de entrada textual, sin validaciones de formato, en el que el usuario debe digitar completamente la fecha de nacimiento. Esto da pie a cometer errores por parte del usuario.

***RichAutoSuggest***

El rich*AutoSuggest* (ver Figura 12-d del capítulo 4) ofrece la posibilidad de desplegar un listado de opciones que facilitan al usuario la escritura de texto. Para el enfoque *MoWebA* con *RIA*, en el campo país de origen, a medida que el usuario va introduciendo caracteres correspondientes al país deseado, interactivamente se despliegan todos los países que coinciden con el patrón introducido, permitiendo navegar de arriba a abajo por medio de un cursor sobre los distintos países. El cursor se resalta con un color diferente a medida que se va recorriendo por los países sugeridos. Una vez que el usuario encuentra el país de origen deseado, al presionar la tecla *enter* o al hacer clic sobre el país, este se escribe en el cuadro de texto de entrada. En el enfoque *MoWebA* sin *RIA* aplicado al *Person Manager*, el campo país de origen es un campo de entrada textual que permite el ingreso de cualquier cadena sin ninguna validación ni sugerencia. En este enfoque el usuario debe escribir el país completamente exponiéndolo a cometer errores.

***ToolTip***

A menudo es útil complementar con información adicional los campos de entrada de los formularios Web. Con el *richToolTip* (ver Figuras 12-b y 12-h del capítulo 4) en el *Person Manager* con el enfoque *MoWebA* con *RIA*, un mensaje informativo útil al usuario es desplegado al posar el puntero del mouse sobre un cuadro de texto de entrada en particular. Para el *Person Manager*, en los campos nombre y apellido se muestra un mensaje en el que se indica al usuario que se ingrese el nombre y el apellido completo. Para el caso del campo contraseña, se despliega al usuario, a modo de sugerencia, el mensaje de seguridad que solicita el ingreso de caracteres alfanuméricos con mayúsculas y minúsculas combinados con caracteres especiales y que contenga por lo menos una longitud de ocho caracteres. En contrapartida, para el enfoque *MoWebA* sin *RIA*, no se despliegan mensajes interactivos que podrán complementar a un campo en particular.

**PI4: Desde el punto de vista de la lógica de negocios en el lado del cliente, ¿qué ventajas aportan las características *RIA* presentes en la aplicación implementada con *MoWebA* con *RIA* con respecto a *MoWebA* sin *RIA*?**

Cuando se habla de lógica de negocios en el lado del cliente, hablamos de operaciones complejas y específicas para un dominio en particular, como así también de validaciones sobre los datos de entrada. Las extensiones *RIA* propuestas a *MoWebA* abarcan específicamente a las validaciones sobre los campos de entrada en los formularios.

***Validaciones locales de los diversos campos de un formulario***

En el capítulo 4 en la figura 12 se presentan todas las validaciones posibles de campos en el *Person Manager*. La ventaja principal de llevar a cabo validaciones en los formularios de manera local, es que no es necesaria ninguna interacción con el lado servidor, lo cual mejora el rendimiento de la aplicación, evitando retardos al recargar la página tras la solicitud de envío de los datos. Con el *Live validation* es posible llevar a cabo validaciones a los diversos campos de los formularios de la aplicación *Person Manager* aplicando el enfoque *MoWebA* con *RIA*. Dentro de las validaciones que se han efectuado se muestran (ver Figuras 12-b, 12-e, 12-h, 12-i y 12-j del capítulo 4) primeramente la validación en los campos que son obligatorios y que no pueden quedar vacios.

Seguidamente se efectuaron controles locales sobre la cantidad de caracteres que deben tener como mínimo algunos campos, tales como los de usuario (mínimo 3) y clave (mínimo 8) como se aprecia en las Figuras 12-g y 12-h del capítulo 4. Para los campos clave y confirmación de clave, también se verificó que ambos coincidan en los valores introducidos (ver Figura 12-h del capítulo 4).

Para los campos numéricos, se valida que solamente sea posible el ingreso de dígitos (valores del 0 al 9), por ejemplo, en el campo id de la vista Remover persona, utilizado para borrar un registro del sistema. En este campo, no es posible ingresar más de 10 dígitos para evitar algún desbordamiento numérico (ver Figura 12-i del capítulo 4). También, el campo email verifica que la cadena ingresada por el usuario corresponda a un email válido (ver Figura 12-f del capítulo 4).

Por último, en el campo de selección de género es obligatorio seleccionar uno de los radio controles (masculino, femenino), como así también, es mandatorio seleccionar la caja de selección del campo de conformidad (ver Figuras 12-i y 12-j del capítulo 4). Los datos introducidos en el formulario solo serán enviados al servidor cuando todos los campos pasen la validación correspondiente a cada uno de ellos. Para el enfoque aplicado a *MoWebA* sin *RIA*, si se desea validar los campos de entrada, es necesario implementar estas validaciones manualmente. El no contar con ninguna validación tiene implicancias negativas a nivel de seguridad, ya que la aplicación queda expuesta a usos indebidos y errores involuntarios en el ingreso de los datos.

**PI5: Para cada una de las vistas del *Person Manager,* ¿qué cantidad de líneas de código para la interfaz de usuario se pudieron generar de manera automática a partir de los modelos, en cada uno de los enfoques implementados?**

Analizando primeramente el tamaño total del *Person Manager* para ambos enfoques, se puede apreciar que el enfoque sin extensiones *RIA* posee 123 líneas de código menos (equivalente a un 32 %) que el enfoque con extensiones *RIA*. Esto se debe a que en el enfoque sin *RIA* no se genera código *Javascript* en la interfaz de usuario ya que su interfaz no posee elementos enriquecidos interactivos.

También puede apreciarse que en el enfoque de *MoWebA* sin *RIA* el 47% del código de la aplicación completa fue generado de manera automática a partir de los modelos y el 57% para el caso de *MoWebA* con *RIA*. Teniendo en cuenta que el objetivo de este trabajo de fin de carrera está enmarcado en los *front-ends* de las interfaces de usuario web, el porcentaje restante de la aplicación, que fue generado de manera manual (53% y 43%) respectivamente, corresponde a código para refinar la aplicación final y código para el acceso a la capa lógica y de dominio de la aplicación.

Según la definición del *Person Manager* la vista con más requerimientos funcionales enriquecidos corresponde a la vista Agregar Persona en la cual se debe definir *richToolTips*, *richDatepicker*, *richAutoSuggest* y diversas validaciones de campos en el formulario de entrada. Esto conlleva a tener código *Javascript* y *HTML* generado para cada uno de los elementos enriquecidos que han sido definidos en la vista Agregar Persona (135 líneas de código automático y 56 líneas de código manual, lo que indica que el 70% de la interfaz de usuario fue generada de manera automática a partir de los modelos *PIM* de entrada) para el enfoque de *MoWebA* con *RIA*. Sin embargo para la contraparte (*MoWebA* sin *RIA*) para la vista *Agregar Persona* se tiene un 56% menos de código con respecto a *MoWebA* con *RIA*, en donde el 47% del código fue generado de manera automática y el 53% agregado de manera manual. Vale la pena acotar que la línea de código manual agregado a cada una de las 3 vistas del *Person Manager* es el mismo. En vista que el enfoque *MoWebA* sin *RIA* no posee elementos interactivos, solamente código *HTML* para cada uno de los elementos definidos es generado, por lo tanto, es natural que existan menos líneas de código.

La vista Mostrar Personas no es muy relevante para el análisis debido a que la extensión propuesta a MoWebA no contempla el acceso al modelo de dominio, por lo tanto solo el 2% y el 6% del código es generado de manera automática en cada uno de los enfoques y el resto fue implementado de manera manual. Para la vista Remover Persona, en el enfoque sin *RIA* puede notarse que el 21% del código se genera de manera automática. Esto tiene sentido debido a que el formulario que forma parte de esta vista, contiene solamente un campo de entrada sin ningún tipo de validación por lo que es mínimo el código HTML correspondiente al elemento. Sin embargo para el enfoque *MoWebA* con *RIA*, esta vista es generada de manera automática en un 58%. Esto tiene su justificativo en que el campo de entrada para esta vista, contiene diversas validaciones, por ende código *Javascript* y *HTML* acompaña al elemento.

En el *Person Manager* se genera código que es común para cada una de las vistas, lo que corresponde a código *CSS* para representar la parte estructural de la aplicación y código correspondiente al *Header* *HTML* con las distintas inclusiones a las librerías *jQuery* y el archivo CSS. Puede notarse en la tabla que para ambos enfoques bastante de ese código es generado de manera automática (87% para el enfoque MoWebaA sin *RIA* y 58% en su contraparte *RIA*).

Finalmente se puede concluir de la que es posible generar más del 50% por ciento de la aplicación final *Person Manager* de manera automática para ambos enfoques.

**5.3 RESUMEN DEL CAPÍTULO**

Para ilustrar el uso de la propuesta de extensión a *MoWebA* con características *RIA* y para compararla con la propuesta original, un mismo proyecto de ejemplo se implementó con los dos enfoques. Se definieron unas preguntas de investigación para establecer específicamente cuáles son los aspectos a comparar de ambos enfoques. Seguidamente se identificaron las variables a medir para poder contestar las preguntas de investigación. Para las preguntas de investigación que no implicaban una medición (PI3 y PI4), se utilizaron las muestras de pantalla de las aplicaciones de cada uno de los enfoquespara obtener las respuestas correspondientes.

Luego, durante las implementaciones, se realizaron las mediciones necesarias para las variables de medición asociadas a PI1, PI2 y PI5 y se recolectaron y almacenaron otros datos necesarios para contestar posteriormente las preguntas de investigación.

Finalmente, luego de medir y de recolectar toda la información necesaria para responder las preguntas de investigación, se procedió a analizar estas mediciones y datos recolectados, y se reportaron los resultados obtenidos, respondiendo a cada una de las preguntas de investigación. Con respecto a PI1 el enfoque *MoWebA* con *RIA* tardó 8 minutos más en el proceso de modelado que *MoWebA* sin *RIA*. Esto se debe principalmente a que para establecer características *RIA* con *MoWebA*, es necesario definir un mayor número de propiedades (valores etiquetados) intrínsecas en cada uno de los elementos que forman parte de la extensión, a diferencia de *MoWebA* sin *RIA*. En otras palabras, esto concuerda con la intuición en el sentido de que cuanto mayor es el nivel de detalle en el modelo, mayor tiempo de modelado se requiere. Sin embargo, este hecho no constituye una limitante demasiado grave, teniendo en cuenta que ese tiempo extra de modelado permite a la interfaz de la aplicación *Person Manager* enriquecerse notablemente. Con respecto a PI2, una vez que los modelos de la aplicación se encontraban listos, el siguiente paso era generar código fuente a partir de ellos. Cuando a alguno de los modelos le faltaba definir alguna propiedad para alguno de sus elementos, entonces el código fuente generado en primera instancia, no reflejaba el resultado esperado. En estos casos, iterativamente se volvían a hacer cambios al modelo y luego se generaba de vuelta la aplicación. Un leve incremento en el número de generaciones para el enfoque *MoWebA* con *RIA* se pudo apreciar a partir de los datos recabados, con respecto a *MoWebA* sin *RIA*. Las vistas de agregar persona y de borrar persona son las que incurrieron en la mayor cantidad de generaciones de código. Debido a que los requerimientos *RIA* requieren un mayor nivel de detalle en los modelos para el caso de *MoWebA* con *RIA* con respecto a *MoWebA* sin *RIA*, existe una mayor posibilidad de cometer errores en los modelos y por ende será necesaria una mayor cantidad de generaciones de código para ir depurando la aplicación. Con respecto a PI3, el enfoque *MoWebA* con *RIA* ofreció numerosas ventajas con respecto a las presentaciones enriquecidas, evitando recargas innecesarias de las páginas y presentando *widgets* interactivos como los *richDatePicker*, *richAutoSuggest* y *richToolTip*. El enfoque *MoWebA* sin *RIA* no contempló tales elementos enriquecidos y navegar por cada una de sus páginas implicó recargar completamente cada una de ellas. Con respecto a PI4, el enfoque *MoWebA* con *RIA* permitió llevar a cabo diversas validaciones en los campos de entrada de la aplicación, como campos que deben ser obligatorios, longitudes mínima y máxima de caracteres en un campo, validaciones de claves y formato de email. En contraparte el enfoque *MoWebA* sin *RIA*, no se generó ningún tipo de validación de manera automática.Finalmente, con respecto a PI5, se analizó primeramente el tamaño total del *Person Manager* para ambos enfoques, y se pudo apreciar que el enfoque sin extensiones *RIA* posee 123 líneas de código menos que el enfoque con extensiones *RIA* (equivalente a un 32% menos). Esto se debió a que en el enfoque sin *RIA* no se genera código Javascript en la interfaz de usuario, ya que su interfaz no posee elementos enriquecidos interactivos. Teniendo en cuenta que el objetivo de este trabajo de fin de carrera está enmarcado en los *front-ends* de las interfaces de usuario Web, el código que fue implementado manualmente en la aplicación, 53% (para el enfoque *MoWebA* sin RIA y 43% (para *MoWebA* con *RIA*) respectivamente, corresponde a código para refinar la aplicación final, y código para el acceso a la capa lógica y de dominio de la aplicación. De ahí se puede concluir que el 47% del código del *Person Manager* fue generado de manera automática para el enfoque *MoWebA* sin *RIA* y el 57% para el enfoque *MoWebA* con *RIA*.

1. **CLOC:** http://cloc.sourceforge.net/ 2015 [↑](#footnote-ref-1)