3. Análisis del problema.

A partir de las reflexiones anteriores, el problema central de esta investigación reside en hacer una herramienta que automatice el proceso de romper dependencias entre ensamblados del .NET Framework. Una de las partes importantes es investigar las asunciones para llevar esto a cabo, así como crear escenarios en los que estos puedan ser útiles. Todo esto es con el objetivo de tener una herramienta que permita mejorar la reusabilidad y extensibilidad de componentes .NET cuando solo se tiene acceso a una versión binaria de este, dando como resultado, una aplicación que procesa DLLs creando nuevos DLLs con puntos de extensión de distintas partes que eran concretas e imposibles de cambiar.

Para responder a estas interrogantes, no se desarrolló la aplicación tal cual directamente, más bien se utilizó una metodología que permitió definir los puntos importantes, así como las herramientas y técnicas a utilizar.

El primer paso fue definir los tipos de dependencias que se pensaban atacar durante la investigación, ya que hay un gran número distintos de ellas. Dado el tiempo disponible, se decidió atacar los tipos de dependencia contenida y directa los cuales ya fueron definidos tanto en la introducción y se retoman en el siguiente capítulo. Cada una de estos escenarios a su vez se divide en distintos casos. Como por ejemplo, la dependencia contenida se puede dar cuando se usa una clase más abstracta como variable, ó usando la misma clase concreta, por lo que los problemas se tienen que atacar de distintas formas. Así mismo, las dependencias directas contienen un gran número de sub-casos, como por ejemplo parámetros de retorno, subclases, genéricos, etc. En este caso se optó también por atacar los problemas más comunes que son parámetros y parámetros de retorno.

Una vez definido los alcances de nuestra investigación, se tuvo que decidir los distintos enfoques y tecnologías que se iban a usar para atacar el problema que teníamos. Para esto se usó una metodología ágil. Lo que significa que hacíamos iteraciones pequeñas de la investigación y del software, con objetivos específicos, y una vez que los lográbamos, nos poníamos nuevos objetivos, y cambiábamos la aplicación para que cumpliera con los nuevos requisitos.

Estas iteraciones implicaron que se tuvieran que estudiar los pros y contras del uso de distintos enfoques, como por ejemplo la librería para la manipulación de ensamblados ó la implementación de nuestro propio contenedor de dependencias.

Finalmente, hubo una concentración mayor en el caso de las dependencias contenidas, ya que resultaron ser las más posibles de eliminar de forma completa y

automatizada. Sin embargo se lograron las bases para el estudio de las dependencias directas en futuras investigaciones e incluso de otro tipo de dependencias.

Uno de los objetivos del desarrollo de la aplicación fue el de desarrollar una aplicación con los más altos estándares dictados por la comunidad de desarrollo de software. Es por eso que se siguieron estrictamente las guías dictadas por el libro Framework Design Guidelines, el cual contiene las mejores prácticas para el desarrollo de librerías reusables orientada a objetos.

En el siguiente capítulo detallaremos la construcción de la aplicación, el cual fue dividido en tres distintas capas. También se detallaran los algoritmos usados.