|  |  |
| --- | --- |
|  | **UNIVERSIDAD PERUANA DE CIENCIAS APLICADAS**  **FACULTAD DE INGENIERÍA**  DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN |

Ingeniería de Software: Labeling Correcto de los Objetos de TinyUML

**Trabajo 4**

Integrantes:

Alejandro Carranza

Sebastian Silva

Índice

[Solicitud de Cambio](#_k17ab1yd8h9g) **2**

[Ubicación de Concepto](#_46c3ai94w4t) **2**

[Análisis de Impacto](#_lgwjlucasji) **3**

[Prefactoring](#_1f52u8smj8zl) **4**

[Proceso de aprendizaje](#_pi6j8wbz3ejh) **4**

[Actualización](#_70hjtq875s3y) **5**

[Post Factoring](#_pvh7m5aou7zv) **8**

[Conclusión](#_gygvlv4tk1f) **8**

# Solicitud de Cambio

Se pide implementar una nueva funcionalidad que permita, al agregar un objeto que contenga texto, se autocomplete el nombre con el orden siguiente del objeto, por ejemplo, si se agrega una clase, esta ingresa como class 1, si se agrega otra, esta nueva clase deberá tener el nombre class 2. Para ello se deberá:

* Agregar variables que almacenen el número de los objetos actuales.
* Configurar el comportamiento de creación de objetos para insertarles texto personalizado.

# Ubicación de Concepto

Para localizar los archivos relacionados al concepto se exploró manualmente entre las clases, debido a que la solicitud de cambio es puntual. Se busca primero en las generadoras de el programa para encontrar el paquete draw, model y diagram, las cuales se especuló que tienen importancia en la creación de los objetos.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # | Nombre del archivo | Herramienta Usada | ¿Ubicado? | Comentarios |
| 1 | Main | Exploración Manual | No relacionado | Arranca el programa |
| 2 | AppFrame | Exploración Manual | No relacionado | Genera e inicializa los componentes de la GUI |
| 3 | DrawingContextImpl | Exploración Manual | Relacionado | Este tiene una función encargada de generar el texto de los objetos |
| 4 | CreationHandler | Exploración Manual | Relacionado | Es la clase encargada de generar nuevos objetos |
| 5 | DiagramEditor | Exploración Manual | No relacionado | Clase que se encarga de editar los diagramas existentes |
| 6 | EditorMode | Exploración Manual | No relacionado | Es la clase encargada de la selección y creación de objetos |
| 7 | LineHandler | Exploración Manual | No relacionado | Clase que se encarga de los objetos en forma de línea |
| 8 | SelectionHandler | Exploración Manual | No relacionado | Clase que se encarga de los objetos seleccionados |
| 9 | DiagramElementFactoryImpl | Exploración Manual | Relacionado | Clase que se encarga de generar prototipos de los diagramas |
| 10 | UmlPackage | Exploración Manual | Relacionado | Clase que crea un paquete UML prototipo |
| 11 | UmlComponent | Exploración Manual | Relacionado | Clase que crea un componente UML prototipo |
| 12 | UmlClass | Exploración Manual | Relacionado | Clase que crea una clase UML prototipo |

Tabla 1: Lista de resumen de todas las clases visitadas durante la ubicación del concepto.

# Análisis de Impacto

Se pronostica que el impacto que cause el cambio sea mínimo, debido a que la solicitud requiere que se agregue una variable extra y una funcionalidad más a una clase ya existente.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # | Nombre de la Clase | Herramienta Usada | ¿Impactado? | Comentarios |
| 1 | Main | Exploración Manual | No | Main |
| 2 | AppFrame | Exploración Manual | No | Genera e inicializa los componentes de la GUI |
| 3 | DrawingContextImpl | Exploración Manual | Si | Este tiene una función encargada de generar el texto de los objetos |
| 4 | CreationHandle | Exploración Manual | Si | Es la clase encargada de generar nuevos objetos |
| 5 | DiagramEditor | Exploración Manual | No | Clase que se encarga de editar los diagramas existentes |
| 6 | EditorMode | Exploración Manual | No | Es la clase encargada de la selección y creación de objetos |
| 7 | LineHandler | Exploración Manual | No | Clase que se encarga de los objetos en forma de línea |
| 8 | SelectionHandler | Exploración Manual | No | Clase que se encarga de los objetos seleccionados |
| 9 | DiagramElementFactoryImpl | Exploración Manual | Si | Clase que se encarga de generar prototipos de los diagramas |
| 10 | UmlPackage | Exploración Manual | Si | Clase que crea un paquete UML prototipo |
| 11 | UmlComponent | Exploración Manual | Si | Clase que crea un componente UML prototipo |
| 12 | UmlClass | Exploración Manual | Si- | Clase que crea una clase UML prototipo |

Tabla 2: Lista de todas las clases visitadas durante el análisis de impacto.

# Prefactoring

Para la implementación del requerimiento solicitado se debe primero, localizar los archivos que controlen conceptos claves relacionados. En este caso, se indaga en el programa por el código que:

* Controlen el nombramiento o labeling de las clases, complementos y paquetes.
* Controlen la creación de las clases, complementos y paquetes.

Las características antes mencionadas están identificadas tras el proceso de búsqueda previamente realizado.

Sin embargo, para implementar este requisito será necesario crear una clase encargada del control de la cantidad de objetos que hay en el diagrama, la cual será ubicada de manera global por el momento para poder ubicar donde será llamada.

Esta modificación es puntual y no generará ninguna modificación en el código fuente además de la creación.

El siguiente paso sería la clase DiagramElementFactoryImpl para cambiar el nombre de las clases, ya que esta es la encargada de efectuar esto. En esta clase se conoce que hay una función que agrega los objetos al diagrama y es ahí donde se sabe de qué tipo es al momento de crearlo, esta función nos ayudará a contar la cantidad de objetos creados para cada uno.

Se estima que será precisa la creación de esta nueva clase, además de los cambios que serán efectuados en DiagramElemtFactoryImp.

# Proceso de aprendizaje

El proceso sigue la siguiente leyenda: naranja para clases cambiadas, verde para clases que se agregaran, azul para clases estudiadas, y blanco para clases no estudiadas pero importantes.

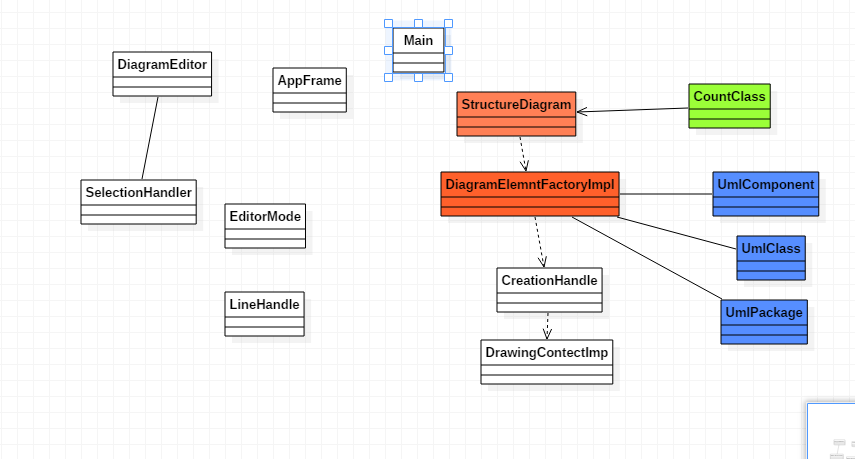
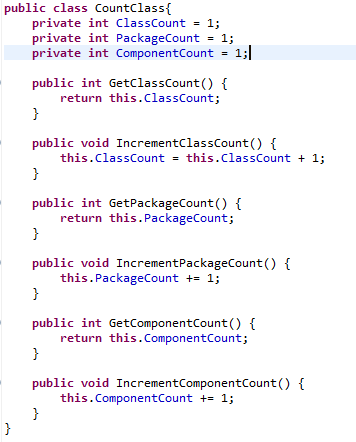


Figura 1: Diagrama de clases importantes.

# Actualización

El primer paso para esta actualización será la creación de la clase CountClass, la cual será la siguiente.



En esta se tiene 6 funcionalidades básicas que incrementan cada objeto y retorna el número de ese objeto.

Luego se necesita encontrar en donde se inicializará esta clase para que mantenga las cantidades; el lugar encontrado es la clase Structure Diagram. En esta, se le agrega la siguiente línea al inicio de la clase como una variable.

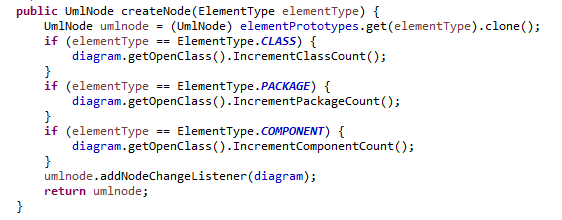


Lo siguiente será crear el get y el set de la misma para poder manejar las cantidades en la siguiente clase.

El siguiente cambio sería el más largo, el cual es la modificación de la clase DiagramElementFactoryImpl, ya que esta contiene esta función:



La cual agrega prototipos y les otorga un nombre, aquí es donde se agrega a cada set name, haciendo referencia al diagrama el valor que le corresponde del siguiente número. Sin embargo, aquí no se aumenta el número ya que estos son prototipos, solo los crea en memoria pero no se agregan al diagrama aun, en el momento que se agregan es el siguiente:



En el que podemos extraer el tipo del elemento que será agregado, en cada caso se hace el aumento del valor si es que elemento agregado es de el tipo que corresponde, de lo contrario solo se omite ese aumento, lo cual no genera ningún problema con los otros.

# Post Factoring

No se encontro ningun problema referente al cambio de las clases, sin embargo, la primera creación de la clase Count Class, esta se llamaba OpenClass, y se encontraba en el mismo paquete que el main. Por cuestiones de orden y buenas prácticas, este pasó al paquete structure, ya que ahí es donde se ejecutan el resto de clases que hacen posible el nombramiento correcto de las clases.

# Conclusión

La mejora realizada en el sistema tenía como meta implementar un contador para cada tipo de objeto creado y aumentarlo cada vez que se cree cada tipo de objeto. Para este fin se agregó una clase que cree los contadores y fue inicializada como variable en la clase Structure Diagram. Luego se realizó el cambio en la clase DiagramElementFactoryImpl para que se incrementen los contadores.

El cambio solicitado fue implementado correctamente y los nombres de los objetos creados aumentan. Se sugiere que a futuro se implemente una funcionalidad para que cada vez que se destruya un objeto, el contador se disminuya para reflejar la cantidad de objetos que existen.