FACULTAD
DE INGENIERIA
Universidad de Buenos Aires

75.42 – Taller de Programación I

Grupo 2

2do Cuatrimestre 2016

Self Interpreter

Manual de Proyecto

Índice

Integrantes

Alumnos

Apellido y nombres	N°Padrón	E-mail
Castro Pippo, Juan Manuel	93.760	jmc.pippo@gmail.com
Moriello, Khalil Alejandro	96.525	khalil.moriello@gmail.com

Tutores del proyecto

Jefe de Cátedra:	Veiga, Andrés
Ayudante 1:	Di Paola, Martín
Ayudante 2 :	Sanguinetti, Axel

Enunciado

Enunciado Simplificado

El tutor del grupo simplificó el enunciado del trabajo práctico para darle una dificultad acorde al trabajo que debería ser realizado por dos personas. Considerando que el enunciado general fue contemplado para grupos de tres integrantes.

Las consideraciones a tener en cuenta fueron las siguientes:

Parser:
No se realizaron modificaciones respecto al enunciado original.

Servidor/Cliente:
Para mantener la concurrencia pero simplificar la dificultad se nos indicó que el servidor tenga un único workspace sobre el que deberían poder trabajar varios clientes en forma simultánea. Luego, si el tiempo lo permitía extender el concepto para que se soporten varios workspaces.

Interfaz gráfica:
Este fue el apartado con mayor simplificación.
Se nos indicó que la interfaz cuente con una línea de comandos desde la que se pueda enviar código self al servidor para ser procesado.

Por otro lado debemos mostrar un único objeto de por vez en pantalla (el cual no debería contar con *drag and drop*), y el usuario debería poder interactuar con el mismo, ya sea agregando slots, permitiendo la posibilidad de acceder a objetos de sus slots y todas *features* que podamos implementar manteniendo la restricción de un único objeto por pantalla.

Enunciado Original

Ver Documento Anexo:

Enunciado Original

División de tareas

La división de tareas se realizó de la siguiente manera.

En la primera clase hicimos esta división:
Castro Pippo, Juan Manuel se encargaría del Parser.
Quino, Julián se encargaría de la Interfaz Gráfica.
Moriello, Alejandro Khalil se encargaría del Cliente/Servidor.

Como a la semana siguiente Julián tuvo que abandonar la materia, el ayudante nos dio un nuevo enunciado (Ver Enunciado Simplificado) en el que se trataba de simplificar principalmente el área que acontecía al tercer integrante que ya no estaba.

La división de tareas luego de esa instancia se definió de la siguiente manera:
Castro Pippo, Juan Manuel se encargaba del Parser y de la Interfaz Gráfica.
Moriello, Khalil Alejandro se encargaba del Cliente/Servidor.

Si bien esa fue la división inicial propuesta, finalmente se realizaron muchas reuniones en donde muchas de las tareas se realizaron en conjunto. De modo tal que la división de tareas en algún punto es más bien orientativa.
Por problemas que tuvo Juan Manuel con GTK, finalmente Khalil se encargó de la implementación de la GUI. Se lista la división real de tareas.

Castro Pippo, Juan Manuel
Parser de código self y Parsers de protocolos.
Diseño de la interfaz gráfica utilizando la herramienta glade.
Diseño e implementación de gran parte de los protocolos del cliente/servidor a "alto nivel". Incluye los métodos de la que se provee la interfaz gráfica al realizar los distintos eventos que disparan acciones hacia el servidor y viceversa.
Informe, Manuales y diagramas.
Selector de modo en el Servidor.

Moriello, Alejandro Khalil
Implementación a bajo nivel de cliente/servidor (aceptador, sockets, proxys).
Esqueleto del cliente/servidor/workspace y la mayor parte de la implementación.
Implementación de la interfaz gráfica.
Documentación Doxygen.
Implementación de object (incluye clone, garbage collector).
Implementación de la mayor parte de la máquina virtual.
Generación de documentos anexos con las herramientas Doxygen y Latex.

En conjunto

Muchas de las tareas antes listadas se realizaron en conjunto, aunque en general se utilizaban para *debug*, definición de diseños o terminar implementaciones que había comenzado alguno de los dos integrantes.

Inconvenientes encontrados

A lo largo del desarrollo del proyecto nos topamos con tres problemas que nos hicieron retrasar bastante.

Parser de código self

Por un lado tuvimos varios inconvenientes con el parser de código self. Si bien inicialmente tuvimos un parser aceptable, *debuggearlo* para que quedara sólido y estable nos insumió varias reuniones. Se sumó a ello que varias veces se dieron ciertas inconsistencias de enunciado sobre la sintaxis BNF.
Por ejemplo (entre varias cosas similares que nos sucedieron), no se pueden sobreescribir operadores ya que *slot_name_extended* sólo soporta names.

Librerías GTK3

Por otro lado tuvimos varios problemas con las librerías de GTK3 obligando a que un miembro del grupo que inicialmente no iba a estar implicado con la implementación de la GUI tuviera que hacerse cargo del tema. Asimismo tuvimos varios problemas de compatibilidad por trabajar con Debian sid que utiliza librerías nuevas que no son consideradas estables en la mayoría de las distribuciones.

Persistencia

Por último tuvimos que pensar varias veces el tema de la persistencia y nunca llegamos a una solución sencilla y satisfactoria. Se desarrolla el tema en el apartádo "**Análisis de puntos pendientes**".

Análisis de puntos pendientes

Keyword Message

Se nos aconsejó dejar de lado el Keyword Message dado que el Parser es sólido y estaba muy avanzado. En cambio, se nos recomendó profundizar en puntos que teníamos más flojos relacionados a la GUI y al Cliente/Servidor.

Workspaces públicos y privados

En la semana anterior a la entrega final teníamos un solo workspace con múltiples clientes (como fue la propuesta inicial en la simplificación del enunciado). Como el ayudante nos vio relativamente avanzados nos propuso apuntar a múltiples workspaces y como plus si llegábamos bien a implementarlos, incorporar el concepto de workspaces públicos y privados. Finalmente este plus no se llegó a implementar.

Persistencia

Se fue charlando con los ayudantes a lo largo del proyecto distintas formas de implementar la persistencia. En todas había que pensar varias veces el tema. No era tan sencillo como parecía inicialmente.

La semana anterior a la entrega final fuimos con una propuesta muy sencilla de implementar, pero al ayudante no le terminó de convencer la idea por lo que la descartamos.

La idea consistía en guardar en un archivo (uno por workspace) todos los comandos que le iban llegando al server relacionados al mismo. Si bien es muy sencillo guardar el mundo e inicializarlo. El problema principal de hacerlo de esta manera, es que en realidad se está guardando un historial completo de cambios, y no sólo una "foto" del workspace. Por lo que eventualmente podría crecer en demasía.

Actualización en tiempo real de cambios

Actualmente el cliente no recibe novedades del servidor a menos que el mismo realice la solicitud explícitamente.

Inicialmente no nos pareció cómodo desde el punto de vista del usuario que si por ejemplo estaba trabajando sobre el bloque de código de un objeto y todavía no había guardado los cambios, pueda aparecer otro cliente y le pise lo que había armado.

Sobre la preentrega lo pensamos bastante y no parecía tan grave la situación antes mencionada. Haciendo un balance llegamos a la conclusión de que iba a resultar más

intuitivo para el usuario tener la desventaja del ítem antes mencionado pero ver una actualización constante de novedades (aunque eso implicara pisarle cambios).

Aplicar estos cambios nos implicaba un rediseño importante y estando a una semana de la entrega final no parecía viable.

Para tratar de suplir esta deficiencia se agregó botones de actualización para que cuando un usuario estuviese por modificar un objeto, antes tuviese la posibilidad de ver las últimas novedades.

Herramientas

A continuación se enumeran las herramientas que se utilizaron durante el desarrollo del proyecto.

gdb
Es una herramienta de debug.
Se utilizó para depurar y corregir bugs que de otra manera hubieran sido difíciles de identificar y resolver.
https://www.gnu.org/software/gdb/

valgrind
Es una herramienta que permite depurar y buscar leaks de memoria y analizar el manejo de hilos.
http://valgrind.org/

git
Es una herramienta que permite realizar un control de versiones.
https://git-scm.com/

github
Plataforma web para manejo del repositorio de git
https://github.com/
Repositorio del proyecto:
https://github.com/kmoriell/self-interpreter

Eclipse (IDE)
Se utilizó para simplificar el desarrollo.
Tiene muchas utilidades de asistencia de código y verificación de sintaxis entre otras virtudes.
https://eclipse.org/

Graphviz
Es una herramienta que permite representar grafos.
Se utilizó para realizar el diagrama del funcionamiento del Parser.
http://www.graphviz.org/

Lucidchart
Es una herramienta web que permite realizar diagramas de distintos tipos de forma colaborativa entre varias personas en simultáneo.
Se utilizó para realizar los diagramas UML del proyecto.
https://www.lucidchart.com

MyBalsamiq
Es una herramienta web que permite realizar mockups (bosquejos) de forma colaborativa entre varios personas en simultáneo.
Se utilizó para realizar mockups de la GUI del cliente.
https://www.mybalsamiq.com/

Glade
Es una herramienta que permite simplificar la implementación de la interfaz gráfica en GTK.
https://glade.gnome.org/

Doxygen
Es una herramienta que permite generar documentación a partir del código fuente.
http://www.stack.nl/~dimitri/doxygen/

Latex
Es una herramienta de composición de textos, orientado a la creación de documentos técnicos.
Se utilizó en combinación con Doxygen para generar el pdf con el código fuente y el pdf con la documentación de las clases y los métodos.
https://www.latex-project.org/

Conclusiones

Llegamos a la conclusión de que es clave armar un diseño sólido y consistente en las primeras etapas del proyecto. Si bien se sabe que los proyectos son iterativos e incrementales, hay ciertos cambios de diseño que no detectados a tiempo pueden ser catastróficos.
En particular a nuestro grupo le pasó que al tener un enunciado simplificado, se dieron varios items del enunciado que quedaron un poco ambiguos al no ser definidos por escrito. Lo que a semanas posteriores en charlas con el ayudante iban cambiando ciertas interpretaciones lo que nos obligó a rediseñar varios módulos.

Por otro lado nos pareció muy interesante el tópico del proyecto.
Estábamos al tanto de que en la cátedra se venían proponiendo proyectos de juegos, y de hecho nos anotamos en la misma pensando que así iba a ser.

Si bien en un primer momento el enunciado del trabajo práctico resultó incluso hasta confuso, luego resultó bastante gratificante tener una máquina virtual de un lenguaje de programación. Es algo distinto y original.