



UNIVERSIDAD
Blas Pascal

Programación **Genérica y Eventos:** **Parcial I**

- **Asignatura:** Programación Genérica y Eventos.
- **Profesora:** Mónica Nano.
- **Alumnos:** Tomas Molina y Edgar Karpowicz.
- **Tema:** Parcial I – Análisis y Mejora de la Interfaz de Usuario en Software utilizando C++.
- **Fecha:** 13/09/24.

Materia: Programación Genérica y Eventos.
Institución: Universidad Blas Pascal.
Profesora: Mónica Nano.
Alumnos: Tomas Molina y Edgar Karpowicz.

Parcial I: Análisis y Mejora de la Interfaz de Usuario en Software utilizando C++

- **Introducción:**

En el Presente Informe, para la Materia Programación Genérica y Eventos de la Carrera Ingeniería Informática en la Universidad Blas Pascal, con el propósito de reforzar e integrar los conceptos de Callbacks, Bucles, Eventos Paint, y Funciones de Conversión, se describirá y señalarán los problemas de usabilidad del Software conocido como GIT, más específicamente, la versión desktop terminal también conocida como “Command-Line Interface” (CLI) de este último. Esto, con el objetivo de, posteriormente, proponer una serie de soluciones que se implementaran en un Prototipo escrito en C++, acompañante de este Informe, que resuelvan las dificultades identificadas en su versión original. Todo esto último haciendo uso de Callbacks, Bucles, Eventos Paint, y Funciones de Conversión.



Ilustración 1 - Logo de GIT

Materia: Programación Genérica y Eventos.
Institución: Universidad Blas Pascal.
Profesora: Mónica Nano.
Alumnos: Tomas Molina y Edgar Karpowicz.

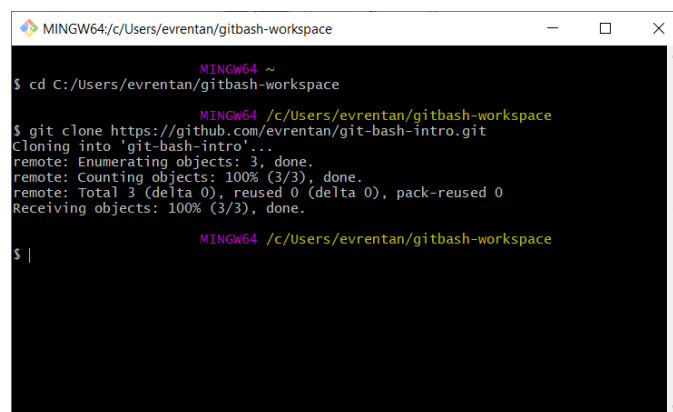
- **GIT/CLI:**

Antes de poder señalar los principales problemas de interfaz de la versión original de GIT, conocida como “Command-Line Interface” (CLI), y sus posibles soluciones, es necesario definir que es GIT y cuál es su principal propósito.

GIT o git, es un software de control de versiones diseñado por Linus Trovalds. Su principal propósito es llevar registro de los cambios en archivos de computadora incluyendo coordinar el trabajo que varias personas realizan sobre archivos compartidos en un repositorio de código. Esto último quiere decir que, GIT es principalmente utilizado para rastrear cambios en código fuente durante el desarrollo de Software. Así, permitiendo a múltiples desarrolladores colaborar en un Proyecto al mantener rastro de todas las modificaciones hechas en el Código.

Existen diferentes métodos para acceder a GIT hoy en día, uno de los más conocidos siendo GITHUB, pero el principal, más básico, y poderoso, es la Version Desktop Terminal de GIT, también conocida como “Command-Line Interface”. Esta última, actúa como la forma más directa de acceder a GIT, proveyendo acceso completo a todas las características, funciones y comandos de GIT, algo que no ocurre en los otros métodos o variaciones de Software para acceder a GIT. Por lo tanto, permite acceder a Repositorios, Rastrear Cambios, colaborar con otros directamente desde la Terminal, entre otro sin fin de funciones de GIT.

- <https://git-scm.com/> (Sitio WEB de GIT)

A screenshot of a terminal window titled "MINGW64/c/Users/evrentan/gitbash-workspace". The terminal shows the following commands and output:

```
$ cd C:/Users/evrentan/gitbash-workspace
MINGW64 /c/Users/evrentan/gitbash-workspace
$ git clone https://github.com/evrentan/git-bash-intro.git
Cloning into 'git-bash-intro'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (3/3), done.
MINGW64 /c/Users/evrentan/gitbash-workspace
$ |
```

Ilustración 2 - Command-Line Interface de GIT

Materia: Programación Genérica y Eventos.

Institución: Universidad Blas Pascal.

Profesora: Mónica Nano.

Alumnos: Tomas Molina y Edgar Karpowicz.

● **Problemas de Usabilidad de GIT/CLI:**

A pesar de ser una poderosísima herramienta, la versión “Command-Line Interface” (CLI) de GIT tiene problemas de usabilidad para los Usuarios monumentales. Entre ellos, el principal dilema que se puede mencionar es la Interfaz de Texto Exclusiva, ya que esta misma no provee casi ningún dato, y el hecho que toda acción debe ser hecha mediante oraciones redactadas por el Usuario lo exacerba, todo esto dándose ya que la CLI, no posee ningún elemento gráfico. Así mismo, aunque esta sea la principal dificultad de la CLI, también se puede mencionar la existencia de los siguientes inconvenientes para el Usuario:

- **Falta de Información Contextual** (La CLI de GIT no provee Información Contextual o Pistas sobre que debería hacer el Usuario. Por ejemplo, si ocurre un Conflicto de Operación Merge, la CLI no sugerirá resolver el Conflicto a menos que el Usuario sepa los Comandos Necesarios).
- **Poca Guianza del Usuario** (En la CLI no hay guianza y/o ayuda para el Usuario para realizar Operaciones, sean Complejas o Básicas, por lo que los Usuarios pueden cometer múltiples errores en ya sea redacción o acción al momento de utilizar el Software).
- **No hay Respuesta Visual o Indicadores de Progreso** (Cuando se ejecutan Comandos que toman mucho tiempo, la CLI no da ninguna Respuesta, Salida Visual y/o Indicador de Progreso, solo dándose una Salida / Respuesta, una vez el comando ha finalizado. Por lo que, debido a esto, puede haber situaciones en que el Usuario llegue a la conclusión errónea de que el Programa ha quedado atascado y no responde, o el Comando se ejecutó de forma errónea).
- **Salidas Superadoras** (Algunos Comandos de GIT producen Salidas con grandes cantidades de Texto, las cuales pueden superar al Usuario y ser difíciles de leer. Por ejemplo, los Comandos “git log” o “git diff” pueden inundar la terminal con Información).
- **Personalización Inexistente** (Si bien algunos Aspectos de la CLI pueden ser personalizados / customizados, como los alias o el color, el nivel de customización de la Interfaz es bastante limitado).

Materia: Programación Genérica y Eventos.

Institución: Universidad Blas Pascal.

Profesora: Mónica Nano.

Alumnos: Tomas Molina y Edgar Karpowicz.

- **Baja Velocidad / Multitareas** (El CLI al ser simplemente una Interfaz de Texto, solo opera con un Comando a la vez, sin ningún soporte para realizar Múltiples Tareas y conllevando a que los Comandos tengan que ser escritos uno a la vez en orden. Por lo tanto, la Velocidad y Tolerancia del Trabajo se ve afectada).
- **Y otros Problemas de Usabilidad Menores.**



Ilustración 3 - Dilema de la CLI, si mantenerse como tal o introducir Elementos Gráficos

● **Posibles Soluciones y Prototipo:**

Ante la Situación que la mayoría de los Problemas que sufre CLI derivan de la falta de Elementos Gráficos que clarifiquen el Software y su multitud de diferentes funciones, las soluciones que se podrían implementar serían:

- **Introducción de Elementos Gráficos** (Introducir una Interfaz con Elementos Gráficos en reemplazo de la Interfaz de Texto Exclusiva Original para facilitar el manejo de la APP por parte del Usuario).
- **Sugerencias / Explicaciones de Funciones** (Sugerir o Explicar las Funciones / Características del Programa durante el Run-Time, mediante elementos como Pop-Ups on Hover, o una Sección que explique los Comandos y sus utilidades).
- **Indicadores de Progreso Mejorados** (Implementar Barras de Progreso o Indicadores de Porcentaje. Principalmente útil para las Acciones / Funciones que lleven mucho tiempo en realizarse).

Materia: Programación Genérica y Eventos.

Institución: Universidad Blas Pascal.

Profesora: Mónica Nano.

Alumnos: Tomas Molina y Edgar Karpowicz.

- **Información Relevante Customizable** (Permitirle al Usuario modificar y/o decidir la Información que recibirá como Salida al momento de finalizarse una Operación realizada. Esto principalmente para simplificar y afrontar las Salidas de Texto Superadoras llenas de Información que puede llegar a no ser relevante para el Usuario).
- **Simplificación de Funciones / Comandos** (Simplificar la elección y/o realización de Comandos durante el Run-Time del Programa al hacer la Selección o Sintaxis de estos más simplificada, accesible, y fácil).
- **Visualización Mejorada** (Integrar Opciones de Visualización Básicas, como un Historial o Diagramas).

Para implementar todas estas Soluciones que intentan resolver las cuestiones ya mencionadas anteriormente, se desarrolló un Prototipo en C++, más específicamente con C++/CLI, una versión del lenguaje que permite la integración entre C++ y el .NET Framework.

En este Prototipo Inicial, se agregaron Elementos Gráficos e Incluyeron algunas de las Principales Funciones de GIT en primera instancia, las cuales son, poder designar una Carpeta en la Computadora como Repositorio Local y poder subir Datos que se encuentren en esta a un Repositorio de Git en la Nube.

Apenas uno ejecuta el Software, se encuentra que el Programa le solicita los Datos de Usuario de GIT, desde el Nombre, Email, entre otros Datos. Posteriormente, uno puede interactuar para subir los Datos almacenados Localmente al Repositorio en la Nube. Todo esto realizándose con Callbacks, Bucles, Eventos Paint, y Funciones de Conversión.

Además, se explico / realizo un breve tutorial de los Comandos incluidos, al incluir un Botón / Sección en la cual se encuentra una sección que explica las Utilidades y Funcionamiento de los diferentes Comandos en el Programa.

Materia: Programación Genérica y Eventos.

Institución: Universidad Blas Pascal.

Profesora: Mónica Nano.

Alumnos: Tomas Molina y Edgar Karpowicz.

- **Imágenes del Prototipo:**

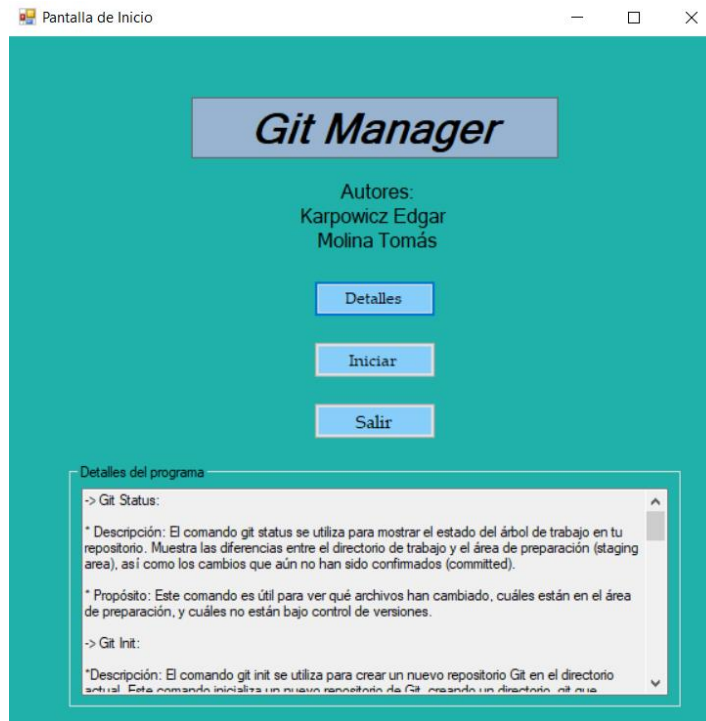


Ilustración 4 - Pantalla de Inicio del Prototipo

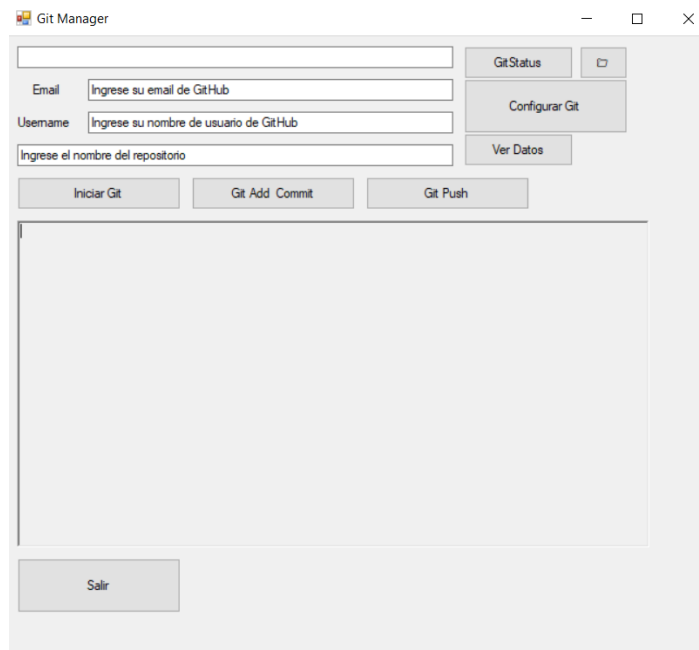


Ilustración 5 - Pantalla de Operaciones del Prototipo

Materia: Programación Genérica y Eventos.

Institución: Universidad Blas Pascal.

Profesora: Mónica Nano.

Alumnos: Tomas Molina y Edgar Karpowicz.

● **Bibliografía:**

- <https://git-scm.com/>
- <https://git-scm.com/book/en/v2/Getting-Started-What-is-Git%3F>
- <https://www.atlassian.com/git/tutorials/what-is-git>
- https://www.atlassian.com/git/tutorials/atlassian-git-cheatsheet?utm_source=gd&utm_medium=paid-display&campaign=21252494340&adgroup=&device=c&devicemode=l=&placement=&aceid=&creative=&adtype=&gclid=Cj0KCQjw28W2BhC7ARIsAPerrcJsb7k0iX3TZLFn9FSmZ0nCyWIEDgeTjnmalmKE46NFnFSEMAC4BVQaAhFUEALw_wcB&country=&network=x&targetid=&gad_source=1
- <https://learn.microsoft.com/es-es/cpp/dotnet/dotnet-programming-with-cpp-cli-visual-cpp?view=msvc-170>