

# Plan de tesis

Villanueva Portella, Jhon Gesell

18/03/2019

## 0.1. Introducción

Un paquete hidro-sedimentario ayuda a la toma de mejores decisiones en la gestión de los recursos hídricos consolidando una base de datos que permita recopilar a lo largo de los años los datos de hidráulica y de sedimentos. Esta base ayuda a poder analizar el comportamiento de los ríos, determinar las épocas de máximas crecidas y secas.

Contar con una base igualmente permite tener un mejor acceso para un mayor público científico y técnico.

Una de las ventajas de tener una herramienta con paquetes libres es el acceso, la continua actualización y evitar la piratería de software.

El paquete que se desea proponer brindará las posibilidades de acceso libre a centros de estudio y centros de investigación; la propuesta pretende beneficiar a todos ellos, con este motivo se pone bajo la licencia GPL V3 ( General Public License).

## 0.2. Objetivos

### 0.2.1. Ojetivos Generales

- Crear un software hidrosedimentario con interfaz gráfica de usuario.
- Brindar un producto que ayude a los científicos e ingenieros que trabajan con flúidos geofísicos.
- Entregar un producto open-source para la comunidad científica internacional.

### 0.2.2. Objetivos Específicos

- Crear una lectura de archivos de caudales.
- Crear un formulario para insertar los datos de las muestras se sedimentos.
- Crear una base de datos.
- Gráficar la sección del río para visualizar las cotas de fondo y los valores de sedimentos en suspensión.

## 0.3. Hipótesis

El software que se desarrolla mediante la presente investigación debe ser calibrado con el software Winriver II de la empresa Teledyne Marine para la visualización de datos brindados por el ADCP (Perfilador de Corriente Acústico Doppler por sus siglas en ingles) Rio Grande de 1200 kHz y a su vez por el VMT (Velocity Mapping Toolbox) de la USGS (Servicio Geológico de los Estados Unidos por sus siglas en ingles) para así ofrecer valores cercanos al 5 % de este.

## 0.4. Materiales y métodos

### 1. Materiales

#### a) Datos recopilados del campo por el ADCP Rio Grande de 1200 kHz:

Los datos que se encuentran almacenados en el ordenador sobre el cual se esta trabajando son llamados mediante el programa "gestor de caudales líquidos y sólidos".

- b) Datos del laboratorio ingresados por el usuario:  
Consiste en insertar en el software propuesto la tabla de muestras procesadas en laboratorio de muestras sólidas.
- c) Librería de desarrollo de aplicaciones con interfaz gráfica de usuario:  
PyQt5 es la librería de código abierto que permite trabajar en el lenguaje Python haciendo uso de multiples clases para así poder aprovechar el framework Qt5 Design.
- d) Lenguaje de programación Python:  
Es un lenguaje open-source multiplataforma y multipropósito que viene creciendo en diferentes campos, hoy muy usado en el campo científico, tiene una gran comunidad en todo el mundo, tiene una sintáxis fácil y limpia.
- e) SQLite3:  
Es un facilitador de base de datos liviano y robusto multiplataforma teniendo alcance de hasta 2 TB por cada base de datos.

## 2. Métodos

- a) Lectura de archivos brindados por el usuario.
- b) Insertar datos por el usuario.
- c) Almacenamiento en la base de datos.
- d) Generación de una matriz.
- e) Interpolación de puntos.
- f) Gráfica de la sección del río.
- g) Gráfica de las celdas de aforo.
- h) Cálculo del caudal.

## 0.5. Resultados esperados

1. Vista preliminar de la interfaz gráfica de usuario  
Para ello vea a la Figura 1
2. Base de datos Se ha elaborado unos scripts que permiten crear la tabla; llenar, actualizar y eliminar los registros que constituyen la base de datos. Todo esto se logró desde el lenguaje Python3.6, los códigos se visualizan en los Anexos 01, 02 y 03 respectivamente.  
Para ello vea a la Figura 2

## 0.6. Cronograma

Meses	Junio	Julio	Agosto	Septiembre	Octubre	Noviembre	Diciembre
R.I.	x	-	-	-	-	-	-
L.B.	x	x	-	-	-	-	-
C.B.D.	x	-	-	-	-	-	-
C.F.	x	-	-	-	-	-	-
I.F.B.D.	-	x	-	-	-	-	-
I.R.	-	-	x	x	-	-	-
A.M.	-	-	-	-	x	-	-
C.M.	-	-	-	-	-	x	-
M.U.	-	-	-	-	-	-	x

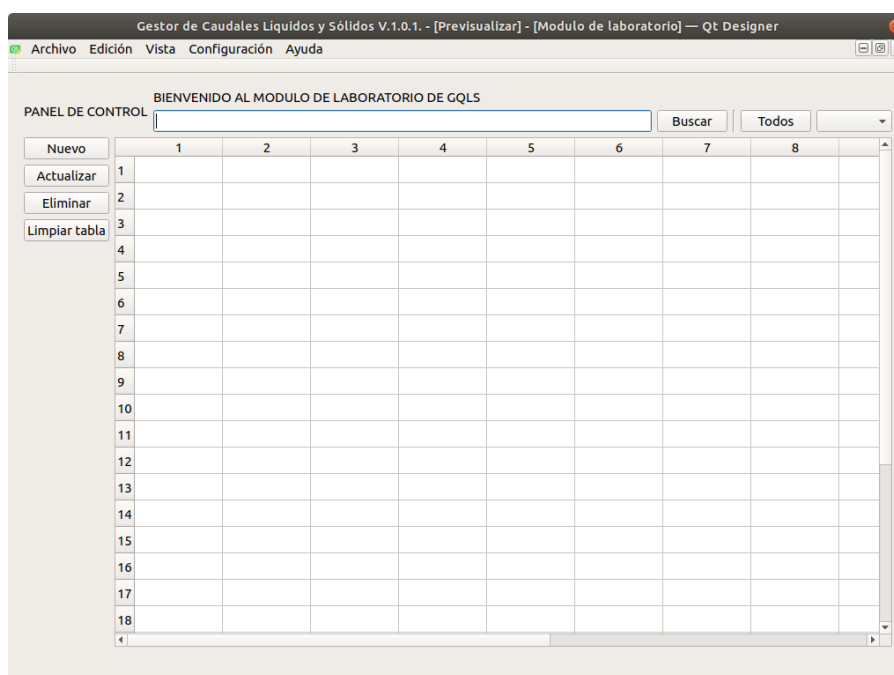


Figura 1: Vista construída con Qt Designer

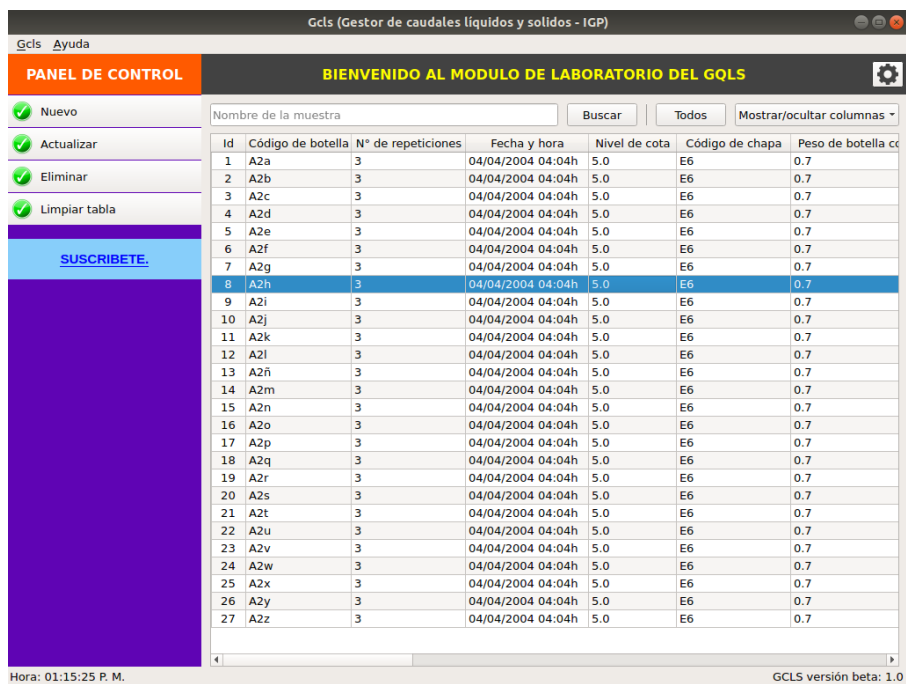


Figura 2: Tabla ingresada por el usuario mediante cuadros de dialogo

- Leyenda:
  - R.I.: Recolección de información.
  - L.B.: Lectura de la bibliografía.
  - C.B.D.: Creación de la base de datos.
  - C.F.: Creación del formulario.
  - I.F.B.D.: Implementación funcional de la base de datos.
  - I.R.: Impresión de resultados en la GUI.
  - A.M.: Ajustes del modelo.
  - C.M.: Corrección del modelo.
  - M.U.: Manual de usuario.

## 0.7. Discusiones

## 0.8. Conclusiones

## 0.9. Bibliografía

- Gonzáles, R. (s.f.). *Python para todos*. Recuperado de: <http://mundogeek.net/tutorial-python>
- Coutinho, N. (2016). *Introducción a la programación con Python: Algoritmos y lógica de programación para principiantes*, Brasil: Novatec Editora Ltda.
- Harwani, B. (2018). *Qt5 Python GUI Programming Cookbook: Building responsive and powerful cross-platform applications with PyQt*, Estados Unidos: Packt Publishing Ltd.
- Owens, M. and Allen, G. (2010). *The Definitive Guide to SQLite*, Estados Unidos: Springer.
- Johansson, R. (2015). *Numerical Python*, Estados Unidos: Springer
- Carneiro, M. (2007). *Manual de redacción superior*, Perú: Editorial San Marcos E.I.R.L.

## 0.10. Anexos

### 0.10.1. Base de datos

1. Creación de la tabla.

*CrearTablaSedimentos.py*

```
#!/usr/bin/python
# -*- coding: cp1251 -*-
```

```
# Crear base de datos y tabla con sqlite3
# 20/06/2018
```

```
__autor__ = u"Jhon Gesell"
```

```

import sqlite3

conexion = sqlite3.connect('tablasedimentos.db')
cursor = conexion.cursor()

#Crear tabla
cursor.execute('''CREATE TABLE MUESTRAS
(CODIGO TEXT NOT NULL,
REPETICIONES TEXT NOT NULL,
FECHA TEXT NOT NULL,
COTA_NIVEL REAL NOT NULL,
COD_CHAPA TEXT NOT NULL,
PESO_BOTELLA_CON_LIQUIDO REAL NOT NULL,
PESO_BOTELLA REAL NOT NULL,
VOLUMEN REAL NOT NULL,
FINOS_PESO_INICIAL_FILTRO REAL NOT NULL,
FINOS_PESO_FINAL_FILTRO REAL NOT NULL,
GRUESOS_PESO_INICIAL_FILTRO REAL NOT NULL,
GRUESOS_PESO_FINAL_FILTRO REAL NOT NULL,
ESTACION TEXT NOT NULL)''')

conexion.close()

```

## 2. Inserción de registros en los campos.

*datosdedimetos.py*

```

#!/usr/bin/python
# -*- coding: cp1252 -*-

# Insertar datos de una tabla con sqlite3
# 21/06/2018

__autor__ = u"Jhon Gesell"

import sqlite3

codigo = raw_input("Codigo: ")
repeticiones = raw_input("Repeticiones: ")
fecha = raw_input("Fecha: ")
cota_nivel = raw_input("Cota nivel: ")
cod_chapa = raw_input("Codigo de botella: ")
peso_botella_con_liquido = raw_input("Peso botella con liquido: ")
peso_botella = raw_input("Peso botella: ")
volumen = raw_input("Volumen: ")
finos_peso_inicial_filtro = raw_input("Peso inicial filtro fino: ")
finos_peso_final_filtro = raw_input("Peso final filtro fino: ")
gruesos_peso_inicial_filtro = raw_input("Peso inicial filtro grueso: ")

```

```

gruesos_peso_final_filtro = raw_input("Peso final filtro grueso: ")
estacion = raw_input("Estacion: ")

conexion = sqlite3.connect('tablasedimentos.db')
cursor = conexion.cursor()

#Insertar datos en la tabla
cursor.execute('''INSERT INTO MUESTRAS(CODIGO, REPETICIONES,
FECHA, COTA_NIVEL, COD_CHAPA, PESO_BOTELLA_CON_LIQUIDO,
PESO_BOTELLA, VOLUMEN, FINOS_PESO_INICIAL_FILTRO,
FINOS_PESO_FINAL_FILTRO, GRUESOS_PESO_INICIAL_FILTRO,
GRUESOS_PESO_FINAL_FILTRO, ESTACION)
VALUES ('%s', '%s', '%s', '%s', '%s',
'%s', '%s', '%s', '%s', '%s', '%s', '%s', '%s')
''' %(codigo, repeticiones, fecha, cota_nivel,
cod_chapa, peso_botella_con_liquido,
peso_botella,
volumen, finos_peso_inicial_filtro,
finos_peso_final_filtro,
gruesos_peso_inicial_filtro,
gruesos_peso_final_filtro, estacion))

conexion.commit()
conexion.close()

```

### 3. Actualización de registros en los campos.

*ActualizarSedimentos.py*

```

#!/usr/bin/python
#-*- coding: cp1252 -*-

```

```

# Actualizar datos en una tabla con sqlite3
# 23/06/2018

```

```

__autor__ = u"Jhon Gesell"

```

```

import sqlite3

```

```

#codigo = raw_input("Codigo: ")
#repeticiones = raw_input("Repeticiones: ")
fecha = raw_input("Fecha: ")
#cota_nivel = raw_input("Cota nivel: ")
cod_chapa = raw_input("Codigo de botella: ")
#peso_botella_con_liquido = raw_input("Peso botella con liquido: ")
#peso_botella = raw_input("Peso botella: ")
volumen = raw_input("Volumen: ")
#finos_peso_inicial_filtro = raw_input("Peso inicial filtro fino: ")
#finos_peso_final_filtro = raw_input("Peso final filtro fino: ")

```

```

")
#gruesos_peso_inicial_filtro = raw_input("Peso inicial filtro
grueso: ")
#gruesos_peso_final_filtro = raw_input("Peso final filtro grueso:
")
#estacion = raw_input("Estacion: ")

conexion = sqlite3.connect('tablasedimentos.db')
cursor = conexion.cursor()

```

```

#Actualizar datos en la tabla
cursor.execute("UPDATE MUESTRAS SET VOLUMEN=:volumen
WHERE FECHA=:fecha and COD_CHAPA=:cod_chapa",
              {"volumen": volumen, "fecha": fecha,
               "cod_chapa": cod_chapa})

```

```

conexion.commit()

```

#### 4. Eliminación de registros por campos.

*vaciartablasedimentos.py*

```

#!/usr/bin/python
#-*- coding: cp1252 -*-

# Borrar datos en una tabla con sqlite3

__autor__ = u"Jhon Gesell"

import sqlite3

codigo = raw_input("Codigo: ")
#repeticiones = raw_input("Repeticiones: ")
#fecha = raw_input("Fecha: ")
#cota_nivel = raw_input("Cota nivel: ")
#cod_chapa = raw_input("Codigo de botella: ")
#peso_botella_con_liquido = raw_input("Peso botella con liquido:
")
#peso_botella = raw_input("Peso botella: ")
#volumen = raw_input("Volumen: ")
#finos_peso_inicial_filtro = raw_input("Peso inicial filtro fino:
")
#finos_peso_final_filtro = raw_input("Peso final filtro fino:
")
#gruesos_peso_inicial_filtro = raw_input("Peso inicial filtro
grueso: ")
#gruesos_peso_final_filtro = raw_input("Peso final filtro
grueso: ")
#estacion = raw_input("Estacion: ")

conexion = sqlite3.connect('tablasedimentos.db')
cursor = conexion.cursor()

```



```
#Eliminar datos en la tabla
```

```
cursor.execute("DELETE FROM MUESTRAS WHERE CODIGO = %s" %codigo)  
conexion.commit()
```

```
conexion.close()
```