HERRAMIENTA SOFTWARE PARA EL ANÁLISIS DE PRUEBAS DE PRESIÓN

DIEGO ANDRÉS OJEDA VARGAS

JEFERSON MARÍN RAMÍREZ

UNIVERSIDAD INDUSTRIAL DE SANTANDER

FACULTAD DE INGENIERÍAS FISICOQUÍMICAS

ESCUELA DE INGENIERÍA DE PETRÓLEOS

2012

HERRAMIENTA SOFTWARE PARA EL ANÁLISIS DE PRUEBAS DE PRESIÓN.

JEFESON MARIN RAMIREZ

DIEGO ANDRES OJEDA VARGAS

Trabajo de Grado para optar por el título de Ingeniero de Petróleos

Director:

MSC. OLGA PARTRICIA ORTIZ CANCINO

Docente UIS

Codirector:

MSC. ELKIN SANTAFE RANGEL

Ingeniero de Petróleos

UNIVERSIDAD INDUSTRIAL DE SANTANDER

FACULTAD DE INGENIERÍAS FISICOQUÍMICAS

ESCUELA DE INGENIERÍA DE PETRÓLEOS

BUCARAMANGA

2012

DEDICATORIA

A Dios que hizo posible este día

A mis padres por su amor incondicional siempre ofrecieron amor y apoyo incondicional y me demuestro que no existe amor más fuerte que el amor de un padre por sus hijos.

A mis abuelos que siempre sonaron con un ver aunque solo fuera uno de sus nietos convertido en un profesional antes de partir.

A mis hermanos quienes verán el fruto de este arduo trabajo

A mis amigos a quienes les debo mucho y parte de lo que hoy soy se debe a todas esas experiencias malas y buenas.

Jeferson Marín Ramirez

DEDICATORIA

A Dios por permitirme estudiar esta carrera tan grandiosa.

A toda mi familia por su apoyo incondicional, especialmente a Nancy mi madre, Clara Inés mi abuela y Marina mi tía.

A la Universidad Industrial de Santander y a la Escuela de Ingeniería de Petróleos por la formación integral que recibí.

A mis amigos y compañeros por ser un gran apoyo académico, profesional y personal.

A Andrea por su apoyo y su paciencia durante todo este proceso.

Diego Andrés Ojeda Vargas

AGRADECIMIENTOS

Expresamos un especial agradecimiento a Juan Correa por su acompañamiento en este proyecto, pues nos apoyó con capacitaciones, datos de pruebas y su vasto conocimiento técnico.

A Gerson Rivera por facilitarnos la construcción de las pruebas sintéticas.

A Olga Patricia Ortiz por su entrega a este proyecto.

A Juan David Morales y Otros por facilitarnos su modelo de fallas conductivas.

A Elkin Santafé y Nelson Quintero por su asesoría y dirección en este trabajo.

CONTENIDO

Pág

LISTA DE FIGURAS

LISTA DE TABLAS

LISTA DE ANEXOS

RESUMEN

TITULO: HERRAMIENTA SOFTWARE PARA EL ANALISIS DE PRUEBAS DE PRESIÓN

AUTORES: JEFERSON MARIN RAMIREZ; DIEGO ANDRES OJEDA VARGAS*

Palabras clave: software libre, pruebas de presión, C#, programación orientada a Objetos, PDD, PBU, YNF´s, pozos hidráulicamente fracturados, cañoneo parcial, algoritmo de suavizado**

DESCRIPCIÓN

El aprendizaje de Pruebas Transientes de Presión (PTA por sus siglas en Ingles) sin una herramienta de software puede ser llegar a ser un trabajo tedioso y que demanda tiempo, esto impide alcanzar un conocimiento más profundo en la materia. Las herramientas de software de Pruebas Transientes de Presión se han convertido en poderosas herramientas, pero la mayoría de los usuarios ignoran su funcionamiento interno. Este trabajo presenta una alternativa de software libre desarrollado para plataforma Windows en el lengua de programación C# orientado a objetos (POO) para el Aprendizaje asistido de Pruebas Transientes de Presión, implementando una interfaz de usuario gráfica amigable. El software está en la capacidad de analizar pruebas de declinación de Presión (PDD por sus siglas en Inglés) a tasa constante y tasa variable, Pruebas de Restauración de Presión (PBU por sus siglas en Ingles) a tasa constante y tasa variable, además cuenta con análisis especiales para Yacimientos Naturalmente Fracturados (YNF's), Pozos Fracturados Hidráulicamente y Parcialmente Cañoneados Utilizando Análisis Convencionales (Métodos de Línea Recta), Modernos por medio de la Síntesis Directa de Tiab y un algoritmo de Suavizado para pruebas ruidosas en la derivada.

1

*Trabajo de Grado.

** Facultad de Ingenierías Fisicoquímicas. Escuela de ingeniería de petróleos, Director Olga Patricia Ortiz, Codirector Elkin Santafé Rangel

ABSTRACT

TITLE: PRESSURE TRANSIENT ANALYSIS SOFTWARE TOOL

AUTHORS: JEFERSON MARIN RAMIREZ; DIEGO ANDRES OJEDA VARGAS*

Key Words: open software, pressure transient analysis, C#, OOP, PDD, PBU, NFR´s, hidraulic Fracture, partial penetration, smoothing**

DESCRIPTION

Learning of Pressure Transient Analysis (PTA) without software tools could be a tedious work and time consuming, that limits a deeper Knowledge in the topic. Current PTA Software has become in powerful tools but most users ignore its internal functions. This work present new open source software developed based windows platform in C# with OOP (Object- Oriented Programming) for aided learning of Pressure Transient Analysis employing a friendly Graphic User Interface (GUI). The Software is capable of analyzing Pressure Drowdawn (PDD) constant rate and variable rate, Pressure Buildup (PBU) constant rate and variable rate, plus special analysis including Naturally Fracture Reservoirs (NFR´s), Hydraulic Fracture Wells and Partial Penetration using Conventional Analysis (Straight Line Methods) and Modern Analysis Tiab Direct Synthesis (TDS) and Smoothing Algorithm for Noisy Derivatives.

*Undergraduate Thesis.

**Physical-chemistry Engineering Faculty. School of Petroleum Engineering. Director Olga Patricia Ortiz, Codirector Elkin Santafe Rangel

INTRODUCCION

El aprendizaje de Pruebas Transientes de Presión sin la asistencia de una herramienta de software se puede transformar en una tarea tediosa y repetitiva que demanda tiempo, esto impide alcanzar un conocimiento más profundo en la materia. Las herramientas de software comercial de hoy en día se han convertido en herramientas muy poderosas, pero los usuarios desconocen su funcionamiento interno y algunas poseen un material de ayuda muy pobre que ocasiona que el usuario abandone la herramienta.

En este trabajo se presenta una alternativa de software libre con fines académicos que ha sido desarrollado en un ambiente Windows empleando el lenguaje de programación C# y haciendo uso de estilo de Programación Orientado a Objetos (POO), la herramienta se enfoca a usuarios con un conocimiento básico en análisis de pruebas de presión o que inician su aprendizaje en la asignatura.

Haciendo uso de una interfaz gráfica amigable para el usuario, la herramienta analiza pruebas de declinación de Presión a tasa constante y tasa variable, Pruebas de Restauración de Presión a tasa constante y tasa variable, además incluye análisis especiales para Yacimientos Naturalmente Fracturados, Pozos Fracturados Hidráulicamente y Parcialmente Cañoneados Utilizando Análisis Convencionales (Métodos de Línea Recta) y Modernos (Síntesis Directa de Tiab) e incorpora un algoritmo de Suavizado para pruebas con ruidos en la derivada.

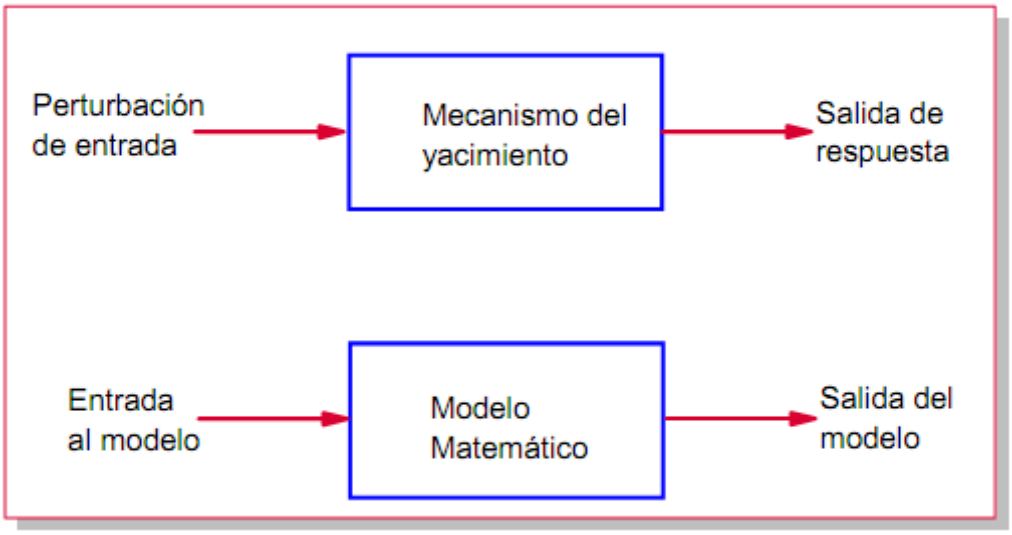
Finalmente se realizan pruebas de comparación con datos de la academia, pruebas sintéticas de presión generadas a través de modelos de simulación y pruebas reales tomadas en algunos campos Colombianos. Para facilitar su uso el software viene acompañado de video tutoriales en cada módulo, manual técnico de referencia, un manual de usuario y un amplio compendio de ejemplos tomados de la literatura.

1. FUNDAMENTACIÓN TEÓRICA USADA EN LA HERRAMIENTA

1.1. CONCEPTOS DE ANÁLISIS DE PRESIONES Y SU APLICACIÓN

Definición Análisis de Pruebas de Presiones: [3]Durante una prueba de pozo, la respuesta de un yacimiento al cambio en las condiciones de producción (o inyección) es monitoreada, dado que la respuesta es en mayor o menor grado, un efecto de las propiedades características del yacimiento y del pozo, es posible en muchos casos inferir las propiedades del yacimiento de esta respuesta obtenida. Las pruebas de análisis de presiones es por lo tanto un problema inverso en la que los parámetros del modelo son inferidos analizando la respuesta de un modelo dado una salida (respuesta o comportamiento).

Figura 1: Esquema de la representación matemática de una prueba de presión



Fuente. Tomado de Análisis Moderno de Presiones. Escobar F. 2003

En términos generales, el objetivo de las pruebas de presiones es obtener información acerca del pozo y el yacimiento, para obtener esta información, la tasa de flujo del pozo es cambiada y esta variación afecta la presión existente en el yacimiento. La medición de las variaciones de presión contra el tiempo y su

[1] Roland N. Horne, Modern Well Test Analysis, 1990, p 1.

interpretación proporciona datos sobre el yacimiento y el pozo. Los cambios de presión son interpretados usando ciertas leyes de la mecánica de fluidos.

1.1.1. Objetivos de Una Prueba de Presión

a). Evaluación del Yacimiento: Para tomar la decisión de poner en producción un yacimiento se debe conocer:

- Su capacidad de entrega
- Propiedades
- Tamaño

A través de las pruebas de presión, se puede determinar: conductividad (kh), la presión inicial (p_i) y los límites del yacimiento.

- La conductividad (kh) gobierna qué tan rápido los fluidos pueden fluir al pozo. Por lo cual es un parámetro a tener en cuenta para diseñar el espaciamiento y el número de pozos.
- La presión indica qué tanta energía tiene el yacimiento y permite pronosticar por cuanto tiempo el yacimiento podrá producir.
- Las presiones en la vecindad del pozo son afectadas por la perforación y por la producción, y puede ser muy diferente del valor de la presión del yacimiento. La interpretación de las pruebas de pozo permite inferir las presiones a distancias considerables del pozo a partir de las presiones locales que se miden en los pozos.
- El análisis de los límites permite determinar cuánto fluido está presente en el yacimiento y si los límites son cerrados o abiertos.

b). Manejo del Yacimiento: Durante la vida del yacimiento se debe monitorear el desempeño y las condiciones de los pozos.

Es útil monitorear los cambios en la presión promedio del yacimiento de tal manera que se puedan refinar los pronósticos de desempeño del yacimiento.

Al monitorear los pozos es posible determinar los candidatos para trabajos de workover o de estimulación.

c.) Descripción del Yacimiento: Las pruebas de presión pueden ser interpretadas para estimar las propiedades globales del yacimiento, ya que dichas pruebas no son sensitivas a las heterogeneidades de escala local.

1.1.2. Aplicación

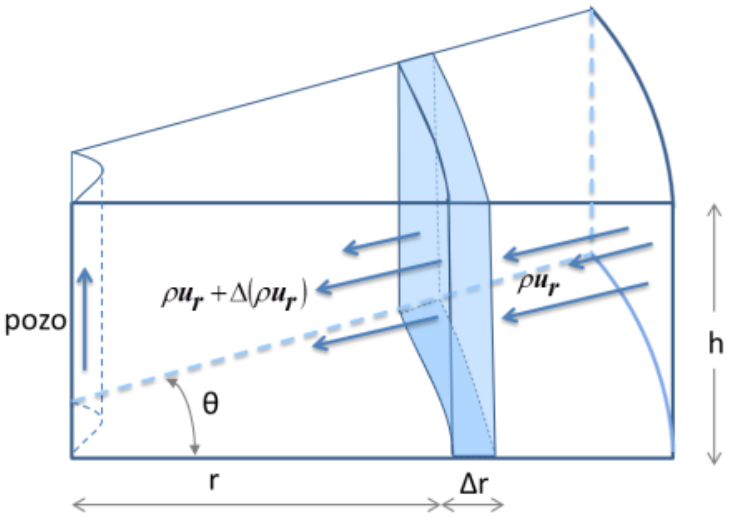
Las pruebas de presión pueden ser usadas para obtener:

- Presión promedio del yacimiento o del área de drenaje
- Permeabilidad de la formación
- Daño
- Efectividad de una estimulación o tratamiento
- Conectividad entre pozos
- Estructuras geológicas
- Longitud de Fracturas
- Conductividad de las Fracturas

1.1.3. Ley de Conservación de la Masa:

Cantidad de masa que entra al sistema menos cantidad masa que sale del sistema es igual a Acumulación de masa en el sistema. Se obtiene la ecuación de continuidad:

Figura 2: Elemento de volumen sobre el cual se aplica balance de masa



Fuente: Imagen Editada de Presentaciones Clases Análisis de Presiones UIS. Ortiz O. 2011

$$\frac{1}{r}\frac{\partial}{\partial r}(r\rho u_r) = -\frac{\partial(\emptyset\rho)}{\partial t}$$

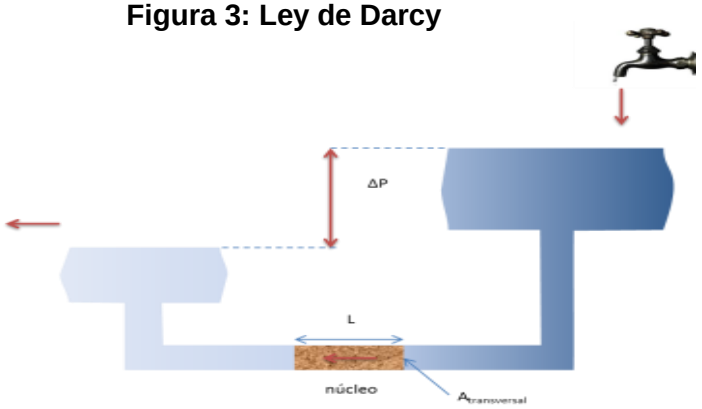
Ecuación 1

1.1.4. La Ley de Darcy es:

$$u_r = -\frac{k}{\mu}\frac{\partial p}{\partial r}$$

Ecuación 2

Figura 3: Ley de Darcy



Fuente: Imagen editada de Introduction to Well Testing.2006

Considera un fluido de comprensibilidad constante: un líquido, por ejemplo: petróleo o agua.

1.1.5. Ecuación de estado:

$$C = -\frac{1}{v}\left(\frac{\partial V}{\partial p}\right)_T = \frac{1}{\rho}\left(\frac{\partial \rho}{\partial p}\right)_T$$

Ecuación 3

Haciendo los respectivos procedimientos matemáticos a partir de las ecuaciones 1-3 se llega a la ecuación de difusividad.

$$\frac{\partial^2 \rho}{\partial r^2} + \frac{1}{r}\left(\frac{\partial \rho}{\partial r}\right) = \frac{\emptyset \mu c_t}{k}\left(\frac{\partial \rho}{\partial t}\right)$$

Ecuación 4

A esta ecuación linealizada se conoce como ecuación de difusividad, la cual puede resolverse para determinadas condiciones iniciales y de contorno mediante métodos analíticos.

Las suposiciones inherentes a la ecuación son:

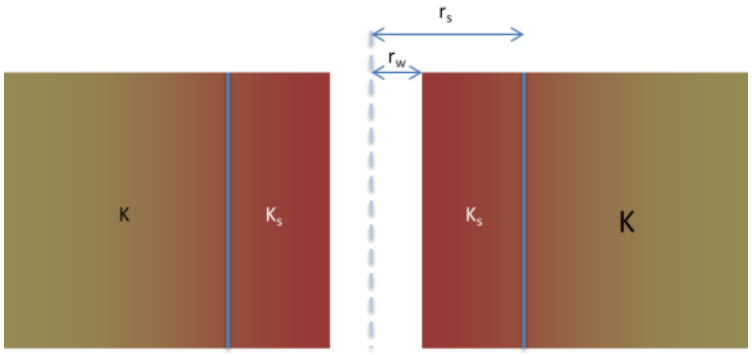
- Flujo radial en el pozo, el cual está produciendo a través de toda la formación
- Medio poroso homogéneo e isotrópico
- Espesor uniforme
- ϕ y K constantes (independientes de la presión)
- Fluido de compresibilidad pequeña y constante
- Gradientes de presión pequeños
- Efectos gravitacionales y térmicos despreciables

- Flujo monofásico
- Viscosidad constante

1.1.6. Daño (Factor Skin)

Van Everdingen y Hurst mostraron que, en muchos casos, la permeabilidad de la formación cerca al Wellbore esta reducida o "dañada", causando una caída de presión adicional. La severidad del daño, se cuantifica por una cantidad adimensional, conocida como el factor skin (S). En la mayoría de los casos S es independiente de la tasa de flujo, pero la caída correspondiente de presión (ΔP_s) depende de la tasa de flujo, un daño positivo representa un daño cerca del pozo, mientras que un valor de daño negativo denota una estimulación, y físicamente significa que hay una caída de presión más pequeña cerca de la cara del pozo de la que se debería esperar en el caso ideal.

Figura 4: Daño de un Pozo



Fuente: editada de Introduction to Well Testing.2006

1.1.7. Principio de Superposición

Este enfoque hace posible obtener respuestas del yacimiento en situaciones complejas, usando modelos básicos.

Se usa para representar:

- La respuesta debida a varios pozos (sumando respuestas individuales)
- Límites del yacimiento (escogiendo de manera apropiada la tasa de flujo y la localización de los pozos)

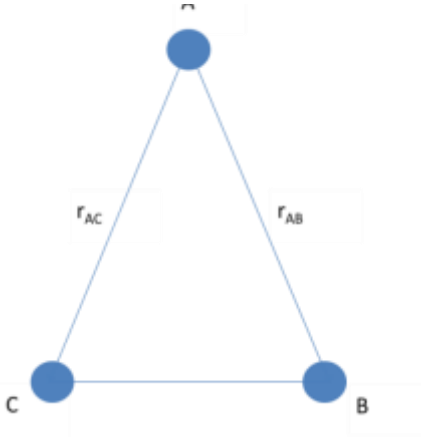
También se puede usar el principio de superposición en el tiempo para determinar la respuesta del yacimiento a un pozo que fluye a tasa variable, usando solamente soluciones a tasa constante.

El principio se resume en “la respuesta de un sistema a un número de perturbaciones es exactamente igual a la suma de las respuestas de cada perturbación”

“la caída de presión total en un punto del yacimiento es la suma de las caídas de presión en ese punto causada por el flujo en cada pozo en el yacimiento”

Ejemplo: Los pozos A, B, C empiezan a producir al mismo tiempo, en un yacimiento infinito. La configuración de los pozos es

Figura 5: Representación esquemática de los pozos



Fuente: Imagen Editada de Presentaciones Clases Análisis de Presiones UIS. Ortiz O. 2011

La caída de presión total en el pozo A, es:

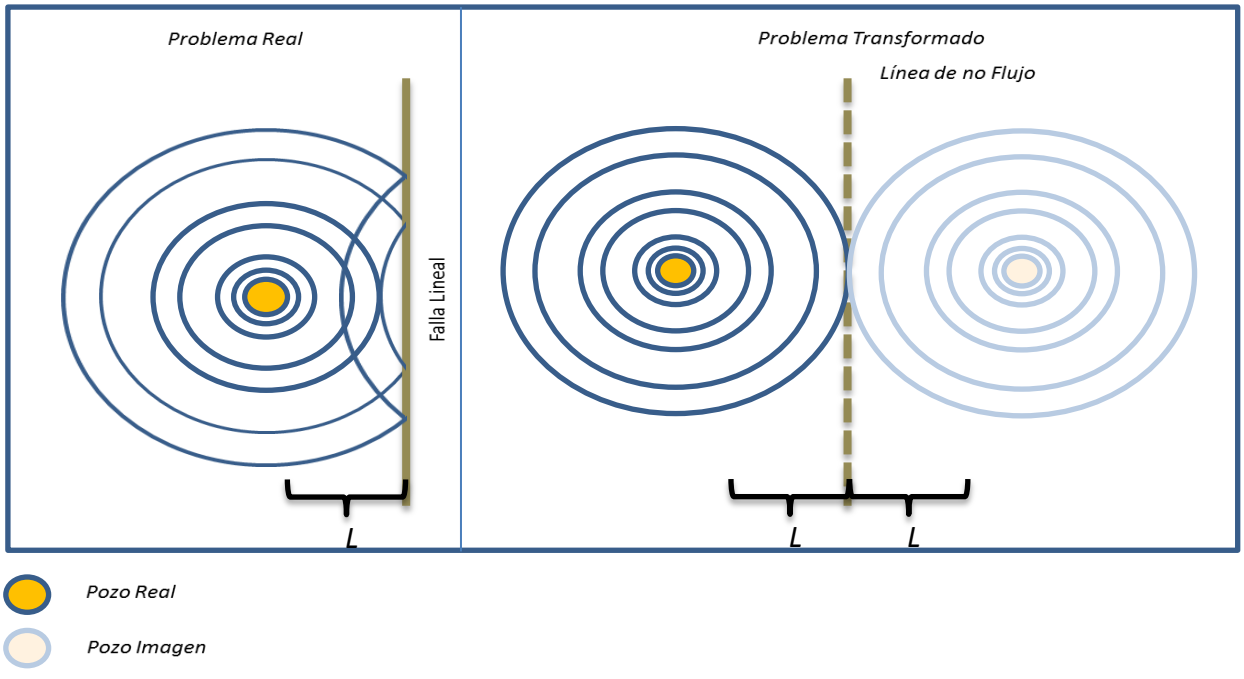
$$(p_i-p_{wf})_{\text{total en el pozo A}} = (p_i-p_{wf})_{\text{total en el pozo A}} + (p_i-p_{wf})_{\text{total en el pozo B}} + (p_i-p_{wf})_{\text{total en el pozo C}}$$

$$(p_i - p_{wf})_{total\ en\ el\ pozo\ A}$$
$$= -70.6\frac{q_A B\mu}{kh}\left(\ln\left(-\frac{1688\emptyset\mu C_t r^2{}_{wA}}{kt}\right) - 2S_A\right)$$
$$-\frac{70.6 q_B B\mu}{kh}\left(E_i\left(-\frac{948\emptyset\mu C_t r^2{}_{AB}}{kt}\right)\right)$$
$$-\frac{70.6 q_c B\mu}{kh}\left(E_i\left(-\frac{948\emptyset\mu C_t r^2{}_{AC}}{kt}\right)\right)$$

Ecuación 5

Otra aplicación del principio de superposición es simular el comportamiento de la presión en yacimientos con límites.

Figura 6: Pozo Único Cerca de una falla sellante

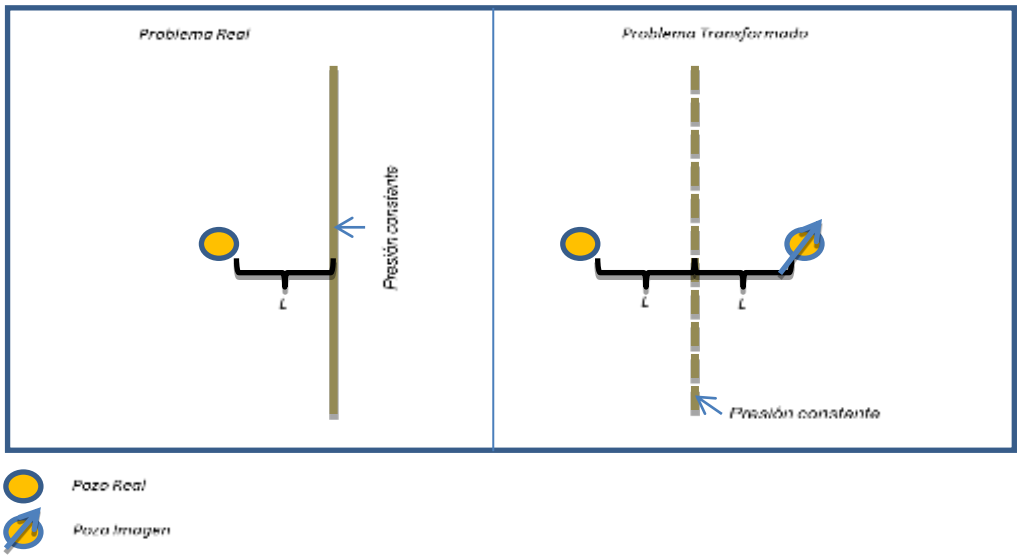


Fuente: Imagen Editada de Presentaciones Clases Análisis de Presiones UIS. Ortiz O. 2011

$$(p_i - p_{wf}) = -70.6\frac{qB\mu}{kh}\left(ln\left(\frac{1688\emptyset\mu C_t {r^2}_w}{kt}\right) - 2S\right) - \frac{70.6qB\mu}{kh}\left(E_i\left(-\frac{948\emptyset\mu C_t(2L)^2}{kt}\right)\right)$$

Ecuación 6

Figura 7: Pozo Cerca de una Barrera de Flujo o Línea de Presión Constante

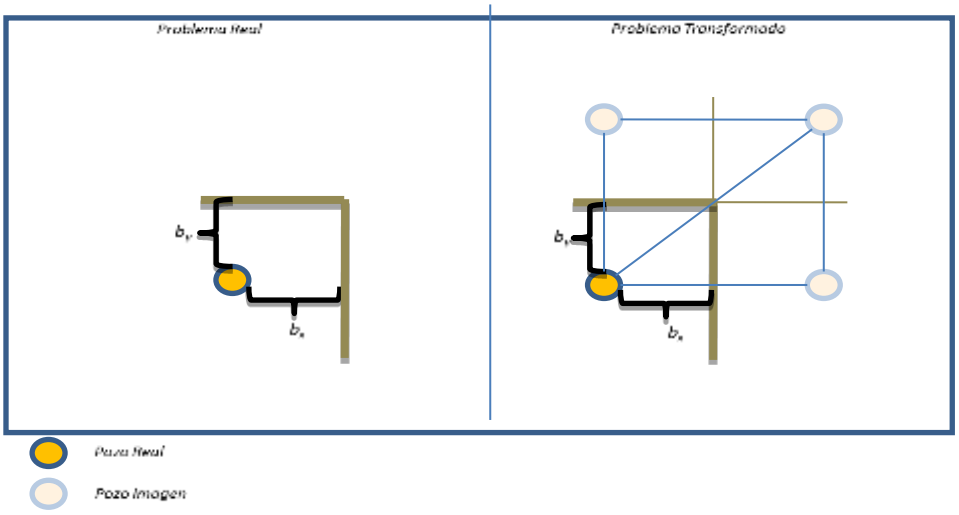


Fuente: Imagen Editada de Presentaciones Clases Análisis de Presiones UIS. Ortiz O. 2011

$$(p_i - p_{wf}) = -70.6\frac{qB\mu}{kh}\left(ln\left(\frac{1688\emptyset\mu C_t {r^2}_w}{kt}\right) - 2S\right) + \frac{70.6qB\mu}{kh}\left(E_i\left(-\frac{948\emptyset\mu C_t(2L)^2}{kt}\right)\right)$$

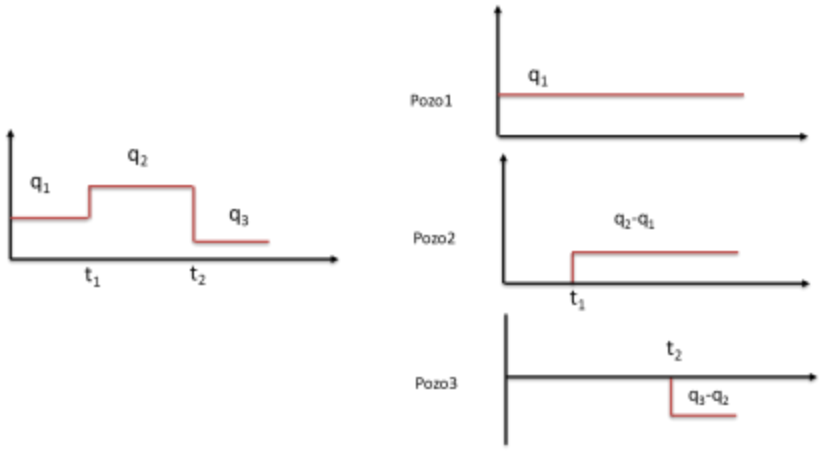
Ecuación 7

Figura 8: Pozo en Medio de dos Fallas que se Interceptan



Fuente: Imagen Editada de Presentaciones Clases Análisis de Presiones UIS. Ortiz O. 2011

Figura 9: Superposición en el Tiempo



Fuente: Imagen Editada de Presentaciones Clases Análisis de Presiones UIS. Ortiz O. 2011

¿Cuál es la presión en la cara de la formación para un $t>t_{2?}$

La primera contribución a la caída de presión es por un pozo que produce a una tasa q_1 y que empieza a un tiempo cero

$$(p_i - p_{wf}) = -70.6\frac{q_1 B\mu}{kh}\left(ln\left(\frac{1688\emptyset\mu C_t r^2{}_w}{kt}\right) - 2S\right)$$

Ecuación 8

Por lo tanto la caída de presión total es:

$$\begin{aligned}(\mathrm{p_i} - \mathrm{p_{wf}}) = -70.6\frac{\mathrm{q_1 B\mu}}{\mathrm{kh}}\left(\ln\left(\frac{1688\emptyset\mu \mathrm{C_t r^2{}_w}}{\mathrm{kt}}\right) - 2\mathrm{S}\right) \\ - 70.6\frac{(\mathrm{q_2} - \mathrm{q_1})\,\mathrm{B\mu}}{\mathrm{kh}}\left(\ln\left(\frac{1688\emptyset\mu \mathrm{C_t r^2{}_w}}{\mathrm{k(t - t_1)}}\right) - 2\mathrm{S}\right) \\ - 70.6\frac{(\mathrm{q_3} - \mathrm{q_2})\,\mathrm{B\mu}}{\mathrm{kh}}\left(\ln\left(\frac{1688\emptyset\mu \mathrm{C_t r^2{}_w}}{\mathrm{k(t - t_2)}}\right) - 2\mathrm{S}\right)\end{aligned}$$

Ecuación 9

2. SOFTWARE

Se conoce como software al equipamiento lógico o soporte lógico de un sistema informático; comprende el conjunto de los componentes lógicos necesarios que hacen posible la realización de tareas específicas, en contraposición a los componentes físicos, que son llamados hardware.

Los componentes lógicos incluyen, entre muchos otros, las aplicaciones informáticas; tales como el procesador de texto, que permite al usuario realizar todas las tareas concernientes a la edición de textos; el software de sistema, tal como el sistema operativo, que, básicamente, permite al resto de los programas funcionar adecuadamente, facilitando también la interacción entre los componentes físicos y el resto de las aplicaciones, y proporcionando una interfaz con el usuario.

Software (pronunciación AFI:[soft'ɣware]) es una palabra proveniente del inglés (literalmente: partes blandas o suaves), que en español no posee una traducción adecuada al contexto, por lo cual se la utiliza asiduamente sin traducir y así fue admitida por la Real Academia Española (RAE). Aunque no es estrictamente lo mismo, suele sustituirse por expresiones tales como programas (informáticos) o aplicaciones (informáticas).

Software es lo que se denomina producto en Ingeniería de Software. Existen varias definiciones similares aceptadas para software, pero probablemente la más formal sea la siguiente:

Es el conjunto de los programas de cómputo, procedimientos, reglas, documentación y datos asociados que forman parte de las operaciones de un sistema de computación. Extraído del estándar 729 del IEEE5

Considerando esta definición, el concepto de software va más allá de los programas de computación en sus distintos estados: código

fuente, binario o ejecutable; también su documentación, los datos a procesar e incluso la información de usuario forman parte del software: es decir, abarca todo lo intangible, todo lo «no físico» relacionado.

El término «software» fue usado por primera vez en este sentido por John W. Tukey en 1957. En la ingeniería de software y las ciencias de la computación, el software es toda la información procesada por los sistemas informáticos: programas y datos.

El concepto de leer diferentes secuencias de instrucciones (programa) desde la memoria de un dispositivo para controlar los cálculos fue introducido por Charles Babbage como parte de su máquina diferencial. La teoría que forma la base de la mayor parte del software moderno fue propuesta por Alan Turing en su ensayo de 1936, «Los números computables», con una aplicación al problema de decisión.

2.1. CLASIFICACIÓN DEL SOFTWARE

Si bien esta distinción es, en cierto modo, arbitraria, y a veces confusa, a los fines prácticos se puede clasificar al software en tres grandes tipos:

Software de sistema: Su objetivo es desvincular adecuadamente al usuario y al programador de los detalles del sistema informático en particular que se use, aislándolo especialmente del procesamiento referido a las características internas de: memoria, discos, puertos y dispositivos de comunicaciones, impresoras, pantallas, teclados, etc. El software de sistema le procura al usuario y programador, interfaces adecuadas de alto nivel, controladores, herramientas y utilidades de apoyo que permiten el mantenimiento del sistema global. Incluye entre otros:

- Sistemas operativos

- Controladores de dispositivos
- Herramientas de diagnóstico
- Herramientas de Corrección y Optimización
- Servidores
- Utilidades

Software de programación: Es el conjunto de herramientas que permiten al programador desarrollar programas informáticos, usando diferentes alternativas y lenguajes de programación, de una manera práctica. Incluyen básicamente:

- Editores de texto
- Compiladores
- Intérpretes
- Enlazadores
- Depuradores

Entornos de Desarrollo Integrados (IDE): Agrupan las anteriores herramientas, usualmente en un entorno visual, de forma tal que el programador no necesite introducir múltiples comandos para compilar, interpretar, depurar, etc. Habitualmente cuentan con una avanzada interfaz gráfica de usuario (GUI).

Software de aplicación: Es aquel que permite a los usuarios llevar a cabo una o varias tareas específicas, en cualquier campo de actividad susceptible de ser automatizado o asistido, con especial énfasis en los negocios. Incluye entre muchos otros:

- Aplicaciones para Control de sistemas y automatización industrial
- Aplicaciones ofimáticas
- Software educativo
- Software empresarial
- Bases de datos
- Telecomunicaciones (por ejemplo Internet y toda su estructura lógica)

- Videojuegos
- Software médico
- Software de cálculo Numérico y simbólico.
- Software de diseño asistido (CAD)
- Software de control numérico (CAM)

2.2. PROCESO DE CREACIÓN DEL SOFTWARE

Se define como proceso al conjunto ordenado de pasos a seguir para llegar a la solución de un problema u obtención de un producto, en este caso particular, para lograr un producto software que resuelva un problema específico.

El proceso de creación de software puede llegar a ser muy complejo, dependiendo de su porte, características y criticidad del mismo. Por ejemplo la creación de un sistema operativo es una tarea que requiere proyecto, gestión, numerosos recursos y todo un equipo disciplinado de trabajo. En el otro extremo, si se trata de un sencillo programa (por ejemplo, la resolución de una ecuación de segundo orden), éste puede ser realizado por un solo programador (incluso aficionado) fácilmente. Es así que normalmente se dividen en tres categorías según su tamaño (líneas de código) o costo: de «pequeño», «mediano» y «gran porte». Existen varias metodologías para estimarlo, una de las más populares es el sistema COCOMO que provee métodos y un software (programa) que calcula y provee una aproximación de todos los costos de producción en un «proyecto software» (relación horas/hombre, costo monetario, cantidad de líneas fuente de acuerdo a lenguaje usado, etc.).

Considerando los de gran porte, es necesario realizar complejas tareas, tanto técnicas como de gerencia, una fuerte gestión y análisis diversos (entre otras

cosas), la complejidad de ello ha llevado a que desarrolle una ingeniería específica para tratar su estudio y realización: es conocida como Ingeniería de Software.

En tanto que en los de mediano porte, pequeños equipos de trabajo (incluso un analista-programador solitario) pueden realizar la tarea. Aunque, siempre en casos de mediano y gran porte (y a veces también en algunos de pequeño porte, según su complejidad), se deben seguir ciertas etapas que son necesarias para la construcción del software. Tales etapas, si bien deben existir, son flexibles en su forma de aplicación, de acuerdo a la metodología o proceso de desarrollo escogido y utilizado por el equipo de desarrollo o por el analista-programador solitario (si fuere el caso).

Los «procesos de desarrollo de software» poseen reglas preestablecidas, y deben ser aplicados en la creación del software de mediano y gran porte, ya que en caso contrario lo más seguro es que el proyecto o no logre concluir o termine sin cumplir los objetivos previstos, y con variedad de fallos inaceptables (fracasan, en pocas palabras). Entre tales «procesos» los hay ágiles o livianos (ejemplo XP), pesados y lentos (ejemplo RUP), y variantes intermedias. Normalmente se aplican de acuerdo al tipo y porte del software a desarrollar, a criterio del líder (si lo hay) del equipo de desarrollo. Algunos de esos procesos son Programación Extrema (en inglés extreme Programming o XP), Proceso Unificado de Rational (en inglés Rational Unified Process o RUP), Feature Driven Development (FDD), etc.

Cualquiera sea el «proceso» utilizado y aplicado al desarrollo del software (RUP, FDD, XP, etc.), y casi independientemente de él, siempre se debe aplicar un «modelo de ciclo de vida».

Se estima que, del total de proyectos software grandes emprendidos, un 28% fracasan, un 46% caen en severas modificaciones que lo retrasan y un 26% son totalmente exitosos.

Cuando un proyecto fracasa, rara vez es debido a fallas técnicas, la principal causa de fallos y fracasos es la falta de aplicación de una buena metodología o proceso de desarrollo. Entre otras, una fuerte tendencia, desde hace pocas décadas, es mejorar las metodologías o procesos de desarrollo, o crear nuevas y concientizar a los profesionales de la informática a su utilización adecuada. Normalmente los especialistas en el estudio y desarrollo de estas áreas (metodologías) y afines (tales como modelos y hasta la gestión misma de los proyectos) son los ingenieros en software, es su orientación. Los especialistas en cualquier otra área de desarrollo informático (analista, programador, Lic. en informática, ingeniero en informática, ingeniero de sistemas, etc.) normalmente aplican sus conocimientos especializados pero utilizando modelos, paradigmas y procesos ya elaborados.

Es común para el desarrollo de software de mediano porte que los equipos humanos involucrados apliquen «metodologías propias», normalmente un híbrido de los procesos anteriores y a veces con criterios propios.

El proceso de desarrollo puede involucrar numerosas y variadas tareas, desde lo administrativo, pasando por lo técnico y hasta la gestión y la gerencia. Pero, casi rigurosamente, siempre se cumplen ciertas etapas mínimas; las que se pueden resumir como sigue:

- Captura, elicitación , especificación y análisis de requisitos (ERS)
- Diseño
- Codificación
- Pruebas (unitarias y de integración)
- Instalación y paso a producción
- Mantenimiento

En las anteriores etapas pueden variar ligeramente sus nombres, o ser más globales, o contrariamente, ser más refinadas; por ejemplo indicar como una única fase (a los fines documentales e interpretativos) de «análisis y diseño»; o indicar

como «implementación» lo que está dicho como «codificación»; pero en rigor, todas existen e incluyen, básicamente, las mismas tareas específicas.

2.2.1. MODELOS DE PROCESO O CICLO DE VIDA

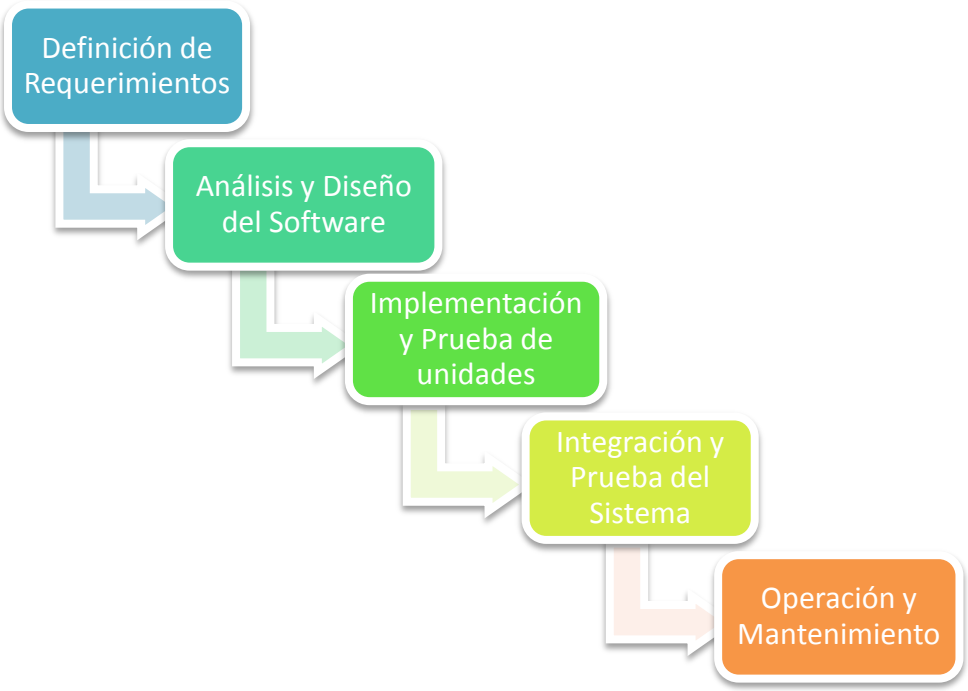
Para cada una de las fases o etapas listadas en el ítem anterior, existen sub-etapas (o tareas). El modelo de proceso o modelo de ciclo de vida utilizado para el desarrollo define el orden para las tareas o actividades involucradas también definen la coordinación entre ellas, y su enlace y realimentación. Entre los más modelos conocidos se puede mencionar: modelo en cascada o secuencial, modelo espiral, modelo iterativo incremental. De los antedichos hay a su vez algunas variantes o alternativas, más o menos atractivas según sea la aplicación requerida y sus requisitos.

2.2.1.1. MODELO CASCADA

Este, aunque es más comúnmente conocido como modelo en cascada es también llamado «modelo clásico», «modelo tradicional» o «modelo lineal secuencial».

El modelo en cascada puro difícilmente se utiliza tal cual, pues esto implicaría un previo y absoluto conocimiento de los requisitos, la no volatilidad de los mismos (o rigidez) y etapas subsiguientes libres de errores; ello sólo podría ser aplicable a escasos y pequeños sistemas a desarrollar. En estas circunstancias, el paso de una etapa a otra de las mencionadas sería sin retorno, por ejemplo pasar del diseño a la codificación implicaría un diseño exacto y sin errores ni probable modificación o evolución: «codifique lo diseñado sin errores, no habrá en absoluto variantes futuras». Esto es utópico; ya que intrínsecamente el software es de carácter evolutivo, cambiante y difícilmente libre de errores, tanto durante su desarrollo como durante su vida operativa.

Figura 10: Modelo de Cascada



Algún cambio durante la ejecución de una cualquiera de las etapas en este modelo secuencial implicaría reiniciar desde el principio todo el ciclo completo, lo cual redundaría en altos costos de tiempo y desarrollo. La figura 10 muestra un posible esquema del modelo en cuestión

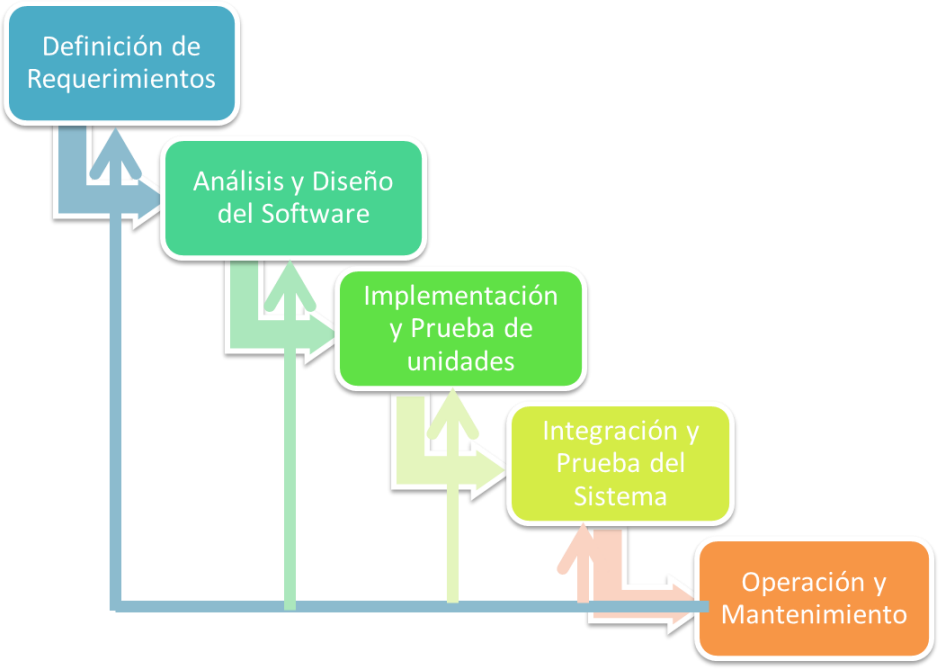
Sin embargo, el modelo cascada en algunas de sus variantes es uno de los actualmente más utilizados, por su eficacia y simplicidad, más que nada en software de pequeño y algunos de mediano porte; pero nunca (o muy rara vez) se lo usa en su "forma pura", como se dijo anteriormente. En lugar de ello, siempre se produce alguna realimentación entre etapas, que no es completamente predecible ni rígida; esto da oportunidad al desarrollo de productos software en los cuales hay ciertas incertezas, cambios o evoluciones durante el ciclo de vida. Así por ejemplo, una vez capturados y especificados los requisitos (primera etapa) se puede pasar al diseño del sistema, pero durante esta última fase lo más probable es que se deban realizar ajustes en los requisitos (aunque sean mínimos), ya sea por fallas detectadas, ambigüedades o bien por que los propios requisitos han cambiado o

evolucionado; con lo cual se debe retornar a la primera o previa etapa, hacer los reajuste pertinentes y luego continuar nuevamente con el diseño; esto último se conoce como realimentación. Lo normal en el modelo cascada será entonces la aplicación del mismo con sus etapas realimentadas de alguna forma, permitiendo retroceder de una a la anterior (e incluso poder saltar a varias anteriores) si es requerido.

De esta manera se obtiene el «modelo cascada realimentado», que puede ser esquematizado como lo ilustra la figura 11.

Lo dicho es, a grandes rasgos, la forma y utilización de este modelo, uno de los más usados y populares. El modelo cascada realimentado resulta muy atractivo, hasta ideal, si el proyecto presenta alta rigidez (pocos cambios, previsto no evolutivo), los requisitos son muy claros y están correctamente especificados.

Figura 11: Modelo Cascada Retroalimentado



Hay más variantes similares al modelo: refino de etapas (más etapas, menores y más específicas) o incluso mostrar menos etapas de las indicadas, aunque en tal caso la faltante estará dentro de alguna otra. El orden de esas fases indicadas en el ítem previo es el lógico y adecuado, pero adviértase, como se dijo, que normalmente habrá realimentación hacia atrás.

El modelo lineal o en cascada es el paradigma más antiguo y extensamente utilizado, sin embargo las críticas a él (ver desventajas) han puesto en duda su eficacia. Pese a todo, tiene un lugar muy importante en la Ingeniería de software y continúa siendo el más utilizado; y siempre es mejor que un enfoque al azar.

Desventajas del modelo cascada

- Los cambios introducidos durante el desarrollo pueden confundir al equipo profesional en las etapas tempranas del proyecto. Si los cambios se producen en etapa madura (codificación o prueba) pueden ser catastróficos para un proyecto grande.
- No es frecuente que el cliente o usuario final explicite clara y completamente los requisitos (etapa de inicio); y el modelo lineal lo requiere. La incertidumbre natural en los comienzos es luego difícil de acomodar.
- El cliente debe tener paciencia ya que el software no estará disponible hasta muy avanzado el proyecto. Un error detectado por el cliente (en fase de operación) puede ser desastroso, implicando reinicio del proyecto, con altos costos.

2.2.1.2. MODELOS EVOLUTIVOS

El software evoluciona con el tiempo. Los requisitos del usuario y del producto suelen cambiar conforme se desarrolla el mismo. Las fechas de mercado y la competencia hacen que no sea posible esperar a poner en el mercado un

producto absolutamente completo, por lo que se aconsejable introducir una versión funcional limitada de alguna forma para aliviar las presiones competitivas.

En esas u otras situaciones similares los desarrolladores necesitan modelos de progreso que estén diseñados para acomodarse a una evolución temporal o progresiva, donde los requisitos centrales son conocidos de antemano, aunque no estén bien definidos a nivel detalle.

En el modelo cascada y cascada realimentado no se tiene demasiado en cuenta la naturaleza evolutiva del software, se plantea como estático, con requisitos bien conocidos y definidos desde el inicio.

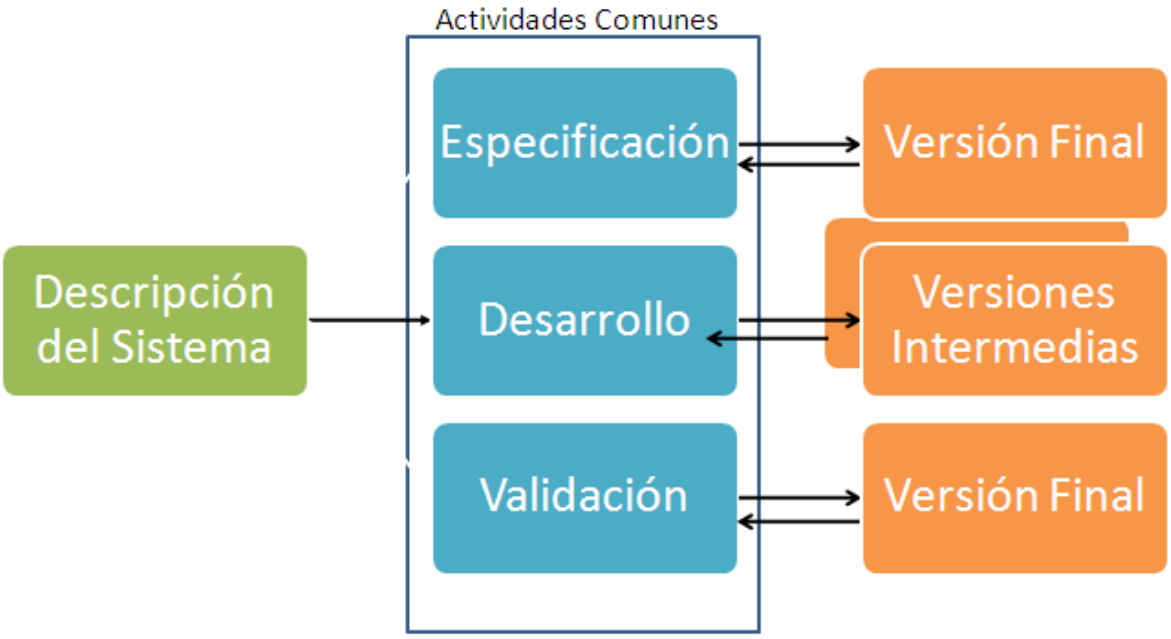
Los evolutivos son modelos iterativos, permiten desarrollar versiones cada vez más completas y complejas, hasta llegar al objetivo final deseado; incluso evolucionar más allá, durante la fase de operación.

Los modelos «iterativo incremental» y «espiral» (entre otros) son dos de los más conocidos y utilizados del tipo evolutivo.

Modelo iterativo incremental

En términos generales, se puede distinguir, en la figura 12, los pasos generales que sigue el proceso de desarrollo de un producto software. En el modelo de ciclo de vida seleccionado, se identifican claramente dichos pasos. La descripción del sistema es esencial para especificar y confeccionar los distintos incrementos hasta llegar al producto global y final. Las actividades concurrentes (especificación, desarrollo y validación) sintetizan el desarrollo pormenorizado de los incrementos, que se hará posteriormente.

Figura 12: Proceso de Desarrollo Productivo de Software

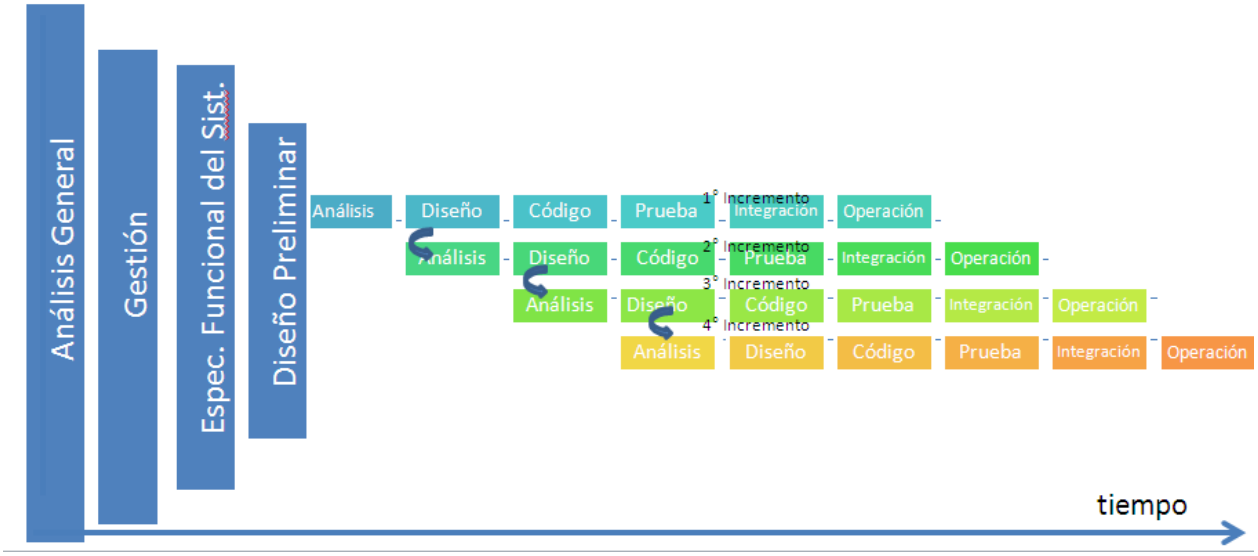


La figura 12 muestra en forma muy esquemática, el funcionamiento de un ciclo iterativo incremental, el cual permite la entrega de versiones parciales a medida que se va construyendo el producto final. Es decir, a medida que cada incremento definido llega a su etapa de operación y mantenimiento. Cada versión emitida incorpora a los anteriores incrementos las funcionalidades y requisitos que fueron analizados como necesarios.

El incremental es un modelo de tipo evolutivo que está basado en varios ciclos Cascada Realimentados aplicados repetidamente, con una filosofía iterativa. En la figura 13 se muestra un refino del diagrama previo, bajo un esquema temporal, para obtener finalmente el esquema del modelo de ciclo de vida Iterativo Incremental, con sus actividades genéricas asociadas. Aquí se observa claramente cada ciclo cascada que es aplicado para la obtención de un incremento; estos últimos se van integrando para obtener el producto final

completo. Cada incremento es un ciclo Cascada Realimentado, aunque, por simplicidad, en la figura 13 se muestra como secuencial puro.

Figura 13: Diagrama Secuencial Puro



Se observa que existen actividades de desarrollo (para cada incremento) que son realizadas en paralelo o concurrentemente, así por ejemplo, en la figura, mientras se realiza el diseño detalle del primer incremento ya se está realizando en análisis del segundo. La figura 13 es sólo esquemática, un incremento no necesariamente se iniciará durante la fase de diseño del anterior, puede ser posterior (incluso antes), en cualquier tiempo de la etapa previa. Cada incremento concluye con la actividad de «operación y mantenimiento» (indicada como «Operación» en la figura), que es donde se produce la entrega del producto parcial al cliente. El momento de inicio de cada incremento es dependiente de varios factores: tipo de sistema; independencia o dependencia entre incrementos (dos de ellos totalmente independientes pueden ser fácilmente iniciados al mismo tiempo si se dispone de personal suficiente); capacidad y cantidad de profesionales involucrados en el desarrollo; etc.

Bajo este modelo se entrega software «por partes funcionales más pequeñas», pero reutilizables, llamadas incrementos. En general cada incremento se construye sobre aquel que ya fue entregado.

Como se muestra en la figura 13, se aplican secuencias Cascada en forma escalonada, mientras progresa el tiempo calendario. Cada secuencia lineal o Cascada produce un incremento y a menudo el primer incremento es un sistema básico, con muchas funciones suplementarias (conocidas o no) sin entregar.

El cliente utiliza inicialmente ese sistema básico, intertanto, el resultado de su uso y evaluación puede aportar al plan para el desarrollo del/los siguientes incrementos (o versiones). Además también aportan a ese plan otros factores, como lo es la priorización (mayor o menor urgencia en la necesidad de cada incremento en particular) y la dependencia entre incrementos (o independencia).

Luego de cada integración se entrega un producto con mayor funcionalidad que el previo. El proceso se repite hasta alcanzar el software final completo.

Siendo iterativo, con el modelo incremental se entrega un producto parcial pero completamente operacional en cada incremento, y no una parte que sea usada para reajustar los requerimientos (como si ocurre en el modelo de construcción de prototipos).

El enfoque incremental resulta muy útil cuando se dispone de baja dotación de personal para el desarrollo; también si no hay disponible fecha límite del proyecto por lo que se entregan versiones incompletas pero que proporcionan al usuario funcionalidad básica (y cada vez mayor). También es un modelo útil a los fines de versiones de evaluación.

Nota: Puede ser considerado y útil, en cualquier momento o incremento incorporar temporalmente el paradigma MCP como complemento, teniendo así una mixtura de modelos que mejoran el esquema y desarrollo general.

Ejemplo:

Un procesador de texto que sea desarrollado bajo el paradigma Incremental podría aportar, en principio, funciones básicas de edición de archivos y producción de documentos (algo como un editor simple). En un segundo incremento se le podría agregar edición más sofisticada, y de generación y mezcla de documentos. En un tercer incremento podría considerarse el agregado de funciones de corrección ortográfica, esquemas de paginado y plantillas; en un cuarto capacidades de dibujo propias y ecuaciones matemáticas. Así sucesivamente hasta llegar al procesador final requerido. Así, el producto va creciendo, acercándose a su meta final, pero desde la entrega del primer incremento ya es útil y funcional para el cliente, el cual observa una respuesta rápida en cuanto a entrega temprana; sin notar que la fecha límite del proyecto puede no estar acotada ni tan definida, lo que da margen de operación y alivia presiones al equipo de desarrollo.

Como se dijo, el Iterativo Incremental es un modelo del tipo evolutivo, es decir donde se permiten y esperan probables cambios en los requisitos en tiempo de desarrollo; se admite cierto margen para que el software pueda evolucionar9 . Aplicable cuando los requisitos son medianamente bien conocidos pero no son completamente estáticos y definidos, cuestión es que si es indispensable para poder utilizar un modelo Cascada.

El modelo es aconsejable para el desarrollo de software en el cual se observe, en su etapa inicial de análisis, que posee áreas bastante bien definidas a cubrir, con suficiente independencia como para ser desarrolladas en etapas sucesivas. Tales áreas a cubrir suelen tener distintos grados de apremio por lo cual las mismas se deben priorizar en un análisis previo, es decir, definir cuál será la primera, la segunda, y así sucesivamente; esto se conoce como «definición de los incrementos» con base en la priorización. Pueden no existir prioridades funcionales por parte del cliente, pero el desarrollador debe fijarlas de todos

modos y con algún criterio, ya que basándose en ellas se desarrollarán y entregarán los distintos incrementos.

El hecho de que existan incrementos funcionales del software lleva inmediatamente a pensar en un esquema de desarrollo modular, por tanto este modelo facilita tal paradigma de diseño.

En resumen, un modelo incremental lleva a pensar en un desarrollo modular, con entregas parciales del producto software denominados «incrementos» del sistema, que son escogidos según prioridades predefinidas de algún modo. El modelo permite una implementación con refinamientos sucesivos (ampliación o mejora). Con cada incremento se agrega nueva funcionalidad o se cubren nuevos requisitos o bien se mejora la versión previamente implementada del producto software.

Este modelo brinda cierta flexibilidad para que durante el desarrollo se incluyan cambios en los requisitos por parte del usuario, un cambio de requisitos propuesto y aprobado puede analizarse e implementarse como un nuevo incremento o, eventualmente, podrá constituir una mejora/adecuación de uno ya planeado. Aunque si se produce un cambio de requisitos por parte del cliente que afecte incrementos previos ya terminados (detección/incorporación tardía) se debe evaluar la factibilidad y realizar un acuerdo con el cliente, ya que puede impactar fuertemente en los costos.

La selección de este modelo permite realizar entregas funcionales tempranas al cliente (lo cual es beneficioso tanto para él como para el grupo de desarrollo). Se priorizan las entregas de aquellos módulos o incrementos en que surja la necesidad operativa de hacerlo, por ejemplo para cargas previas de información, indispensable para los incrementos siguientes.

El modelo iterativo incremental no obliga a especificar con precisión y detalle absolutamente todo lo que el sistema debe hacer, (y cómo), antes de ser

construido (como el caso del cascada, con requisitos congelados). Sólo se hace en el incremento en desarrollo. Esto torna más manejable el proceso y reduce el impacto en los costos. Esto es así, porque en caso de alterar o rehacer los requisitos, solo afecta una parte del sistema. Aunque, lógicamente, esta situación se agrava si se presenta en estado avanzado, es decir en los últimos incrementos. En definitiva, el modelo facilita la incorporación de nuevos requisitos durante el desarrollo.

Con un paradigma incremental se reduce el tiempo de desarrollo inicial, ya que se implementa funcionalidad parcial. También provee un impacto ventajoso frente al cliente, que es la entrega temprana de partes operativas del software.

El modelo proporciona todas las ventajas del modelo en cascada realimentado, reduciendo sus desventajas sólo al ámbito de cada incremento.

El modelo incremental no es recomendable para casos de sistemas de tiempo real, de alto nivel de seguridad, deprocesamiento distribuido, o de alto índice de riesgos.

Modelo espiral

El modelo espiral fue propuesto inicialmente por Barry Boehm. Es un modelo evolutivo que conjuga la naturaleza iterativa del modelo MCP con los aspectos controlados y sistemáticos del Modelo Cascada. Proporciona potencial para desarrollo rápido de versiones incrementales. En el modelo Espiral el software se construye en una serie de versiones incrementales. En las primeras iteraciones la versión incremental podría ser un modelo en papel o bien un prototipo. En las últimas iteraciones se producen versiones cada vez más completas del sistema diseñado.

El modelo se divide en un número de Actividades de marco de trabajo, llamadas «regiones de tareas». En general existen entre tres y seis regiones de tareas (hay

variantes del modelo). En la figura 14 se muestra el esquema de un Modelo Espiral con 6 regiones. En este caso se explica una variante del modelo original de Boehm, expuesto en su tratado de 1988; en 1998 expuso un tratado más reciente.

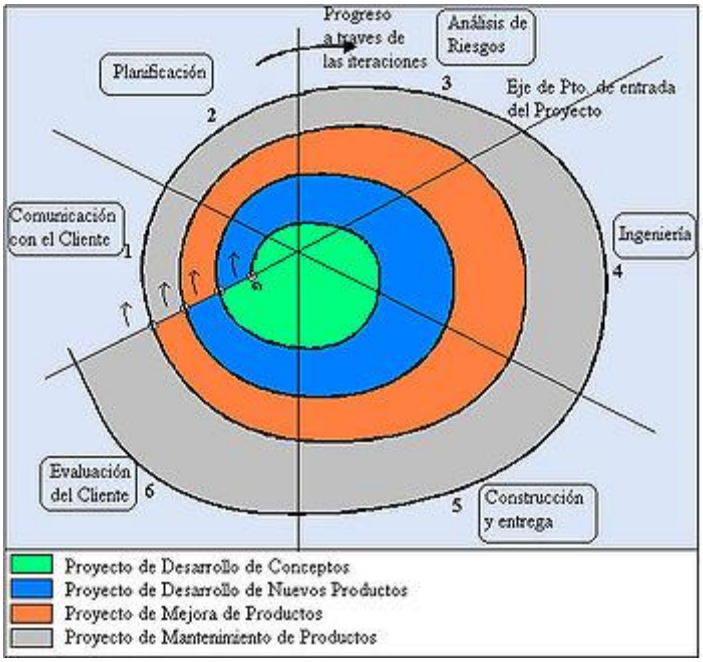


Figura 14: Modelo Espiral

- Región 1 - Tareas requeridas para establecer la comunicación entre el cliente y el desarrollador.
- Región 2 - Tareas inherentes a la definición de los recursos, tiempo y otra información relacionada con el proyecto.
- Región 3 - Tareas necesarias para evaluar los riesgos técnicos y de gestión del proyecto.
- Región 4 - Tareas para construir una o más representaciones de la aplicación software.
- Región 5 - Tareas para construir la aplicación, instalarla, probarla y proporcionar soporte al usuario o cliente (Ej. documentación y práctica).

- Región 6 - Tareas para obtener la reacción del cliente, según la evaluación de lo creado e instalado en los ciclos anteriores.

Las actividades enunciadas para el marco de trabajo son generales y se aplican a cualquier proyecto, grande, mediano o pequeño, complejo o no. Las regiones que definen esas actividades comprenden un «conjunto de tareas» del trabajo: ese conjunto sí se debe adaptar a las características del proyecto en particular a emprender. Nótese que lo listado en los ítems de 1 a 6 son conjuntos de tareas, algunas de las ellas normalmente dependen del proyecto o desarrollo en sí.

Proyectos pequeños requieren baja cantidad de tareas y también de formalidad. En proyectos mayores o críticos cada región de tareas contiene labores de más alto nivel de formalidad. En cualquier caso se aplican actividades de protección (por ejemplo, gestión de configuración del software, garantía de calidad, etc.).

Al inicio del ciclo, o proceso evolutivo, el equipo de ingeniería gira alrededor del espiral (metafóricamente hablando) comenzando por el centro (marcado con ◎ en la figura 14) y en el sentido indicado; el primer circuito de la espiral puede producir el desarrollo de una especificación del producto; los pasos siguientes podrían generar un prototipo y progresivamente versiones más sofisticadas del software.

Cada paso por la región de planificación provoca ajustes en el plan del proyecto; el coste y planificación se realimentan en función de la evaluación del cliente. El gestor de proyectos debe ajustar el número de iteraciones requeridas para completar el desarrollo.

El modelo espiral puede ir adaptándose y aplicarse a lo largo de todo el ciclo de vida del software (en el modelo clásico, o cascada, el proceso termina a la entrega del software).

Una visión alternativa del modelo puede observarse examinando el «eje de punto de entrada de proyectos». Cada uno de los circulitos (◦) fijados a lo largo del eje representan puntos de arranque de los distintos proyectos (relacionados); a saber:

- Un proyecto de «desarrollo de conceptos» comienza al inicio de la espiral, hace múltiples iteraciones hasta que se completa, es la zona marcada con verde.
- Si lo anterior se va a desarrollar como producto real, se inicia otro proyecto: «Desarrollo de nuevo Producto». Que evolucionará con iteraciones hasta culminar; es la zona marcada en color azul.
- Eventual y análogamente se generarán proyectos de «mejoras de productos» y de «mantenimiento de productos», con las iteraciones necesarias en cada área (zonas roja y gris, respectivamente).

Cuando la espiral se caracteriza de esta forma, está operativa hasta que el software se retira, eventualmente puede estar inactivo (el proceso), pero cuando se produce un cambio el proceso arranca nuevamente en el punto de entrada apropiado (por ejemplo, en «mejora del producto»).

El modelo espiral da un enfoque realista, que evoluciona igual que el software ; se adapta muy bien para desarrollos a gran escala.

El Espiral utiliza el MCP para reducir riesgos y permite aplicarlo en cualquier etapa de la evolución. Mantiene el enfoque clásico (cascada) pero incorpora un marco de trabajo iterativo que refleja mejor la realidad.

Este modelo requiere considerar riesgos técnicos en todas las etapas del proyecto; aplicado adecuadamente debe reducirlos antes de que sean un verdadero problema.

El Modelo evolutivo como el Espiral es particularmente apto para el desarrollo de Sistemas Operativos (complejos); también en sistemas de altos riesgos o críticos

(Ej. navegadores y controladores aeronáuticos) y en todos aquellos en que sea necesaria una fuerte gestión del proyecto y sus riesgos, técnicos o de gestión.

Desventajas importantes:

- Requiere mucha experiencia y habilidad para la evaluación de los riesgos, lo cual es requisito para el éxito del proyecto.
- Es difícil convencer a los grandes clientes que se podrá controlar este enfoque evolutivo.

Este modelo no se ha usado tanto, como el Cascada (Incremental) o MCP, por lo que no se tiene bien medida su eficacia, es un paradigma relativamente nuevo y difícil de implementar y controlar.

Modelo espiral Win & Win

Una variante interesante del Modelo Espiral previamente visto (Figura 14) es el «Modelo espiral Win-Win»7 (Barry Boehm). El Modelo Espiral previo (clásico) sugiere la comunicación con el cliente para fijar los requisitos, en que simplemente se pregunta al cliente qué necesita y él proporciona la información para continuar; pero esto es en un contexto ideal que rara vez ocurre. Normalmente cliente y desarrollador entran en una negociación, se negocia coste frente a funcionalidad, rendimiento, calidad, etc.

«Es así que la obtención de requisitos requiere una negociación, que tiene éxito cuando ambas partes ganan».

Las mejores negociaciones se fuerzan en obtener «Victoria & Victoria» (Win & Win), es decir que el cliente gane obteniendo el producto que lo satisfaga, y el desarrollador también gane consiguiendo presupuesto y fecha de entrega realista. Evidentemente, este modelo requiere fuertes habilidades de negociación.

El modelo Win-Win define un conjunto de actividades de negociación al principio de cada paso alrededor de la espiral; se definen las siguientes actividades:

Identificación del sistema o subsistemas clave de los directivos(*) (saber qué quieren).

Determinación de «condiciones de victoria» de los directivos (saber qué necesitan y los satisface)

Negociación de las condiciones «victoria» de los directivos para obtener condiciones «Victoria & Victoria» (negociar para que ambos ganen).

(*) Directivo: Cliente escogido con interés directo en el producto, que puede ser premiado por la organización si tiene éxito o criticado si no.

El modelo Win & Win hace énfasis en la negociación inicial, también introduce 3 hitos en el proceso llamados «puntos de fijación», que ayudan a establecer la completitud de un ciclo de la espiral, y proporcionan hitos de decisión antes de continuar el proyecto de desarrollo del software.

2.2.2. ETAPAS EN EL DESARROLLO DEL SOFTWARE

Captura, análisis y especificación de requisitos

Al inicio de un desarrollo (no de un proyecto), esta es la primera fase que se realiza, y, según el modelo de proceso adoptado, puede casi terminar para pasar a la próxima etapa (caso de Modelo Cascada Realimentado) o puede hacerse parcialmente para luego retomarla (caso Modelo Iterativo Incremental u otros de carácter evolutivo).

En simple palabras y básicamente, durante esta fase, se adquieren, reúnen y especifican las características funcionales y no funcionales que deberá cumplir el futuro programa o sistema a desarrollar.

Las bondades de las características, tanto del sistema o programa a desarrollar, como de su entorno, parámetros no funcionales y arquitectura dependen

enormemente de lo bien lograda que esté esta etapa. Esta es, probablemente, la de mayor importancia y una de las fases más difíciles de lograr certeramente, pues no es automatizable, no es muy técnica y depende en gran medida de la habilidad y experiencia del analista que la realice.

Involucra fuertemente al usuario o cliente del sistema, por tanto tiene matices muy subjetivos y es difícil de modelar con certeza o aplicar una técnica que sea «la más cercana a la adecuada» (de hecho no existe «la estrictamente adecuada»). Si bien se han ideado varias metodologías, incluso software de apoyo, para captura, elicitación y registro de requisitos, no existe una forma infalible o absolutamente confiable, y deben aplicarse conjuntamente buenos criterios y mucho sentido común por parte del o los analistas encargados de la tarea; es fundamental también lograr una fluida y adecuada comunicación y comprensión con el usuario final o cliente del sistema.

El artefacto más importante resultado de la culminación de esta etapa es lo que se conoce como especificación de requisitos software o simplemente documento ERS.

Como se dijo, la habilidad del analista para interactuar con el cliente es fundamental; lo común es que el cliente tenga un objetivo general o problema que resolver, no conoce en absoluto el área (informática), ni su jerga, ni siquiera sabe con precisión qué debería hacer el producto software (qué y cuantas funciones) ni, mucho menos, cómo debe operar. En otros casos menos frecuentes, el cliente «piensa» que sabe precisamente lo que el software tiene que hacer, y generalmente acierta muy parcialmente, pero su empecinamiento entorpece la tarea de elicitación. El analista debe tener la capacidad para lidiar con este tipo de problemas, que incluyen relaciones humanas; tiene que saber ponerse al nivel del usuario para permitir una adecuada comunicación y comprensión.

Escasas son las situaciones en que el cliente sabe con certeza e incluso con completitud lo que requiere de su futuro sistema, este es el caso más sencillo para el analista.

Las tareas relativas a captura, elicitación, modelado y registro de requerimientos, además de ser sumamente importante, puede llegar a ser dificultosa de lograr acertadamente y llevar bastante tiempo relativo al proceso total del desarrollo; al proceso y metodologías para llevar a cabo este conjunto de actividades normalmente se las asume parte propia de la Ingeniería de Software, pero dada la antedicha complejidad, actualmente se habla de una Ingeniería de requisitos , aunque ella aún no existe formalmente.

Hay grupos de estudio e investigación, en todo el mundo, que están exclusivamente abocados a la idear modelos, técnicas y procesos para intentar lograr la correcta captura, análisis y registro de requerimientos. Estos grupos son los que normalmente hablan de la Ingeniería de requisitos; es decir se plantea ésta como un área o disciplina pero no como una carrera universitaria en sí misma.

Algunos requisitos no necesitan la presencia del cliente, para ser capturados o analizados; en ciertos casos los puede proponer el mismo analista o, incluso, adoptar unilateralmente decisiones que considera adecuadas (tanto en requerimientos funcionales como no funcionales). Por citar ejemplos probables: Algunos requisitos sobre la arquitectura del sistema, requisitos no funcionales tales como los relativos al rendimiento, nivel de soporte a errores operativos, plataformas de desarrollo, relaciones internas o ligas entre la información (entre registros o tablas de datos) a almacenar en caso de bases o bancos de datos, etc. Algunos funcionales tales como opciones secundarias o de soporte necesarias para una mejor o más sencilla operatividad; etc.

La obtención de especificaciones a partir del cliente (u otros actores intervinientes) es un proceso humano muy interactivo e iterativo; normalmente a medida que se

captura la información, se la analiza y realimenta con el cliente, refinándola, puliéndola y corrigiendo si es necesario; cualquiera sea el método de ERS utilizado. EL analista siempre debe llegar a conocer la temática y el problema que resolver, dominarlo, hasta cierto punto, hasta el ámbito que el futuro sistema a desarrollar lo abarque. Por ello el analista debe tener alta capacidad para comprender problemas de muy diversas áreas o disciplinas de trabajo (que no son específicamente suyas); así por ejemplo, si el sistema a desarrollar será para gestionar información de una aseguradora y sus sucursales remotas, el analista se debe compenetrar en cómo ella trabaja y maneja su información, desde niveles muy bajos e incluso llegando hasta los gerenciales. Dada a gran diversidad de campos a cubrir, los analistas suelen ser asistidos por especialistas, es decir gente que conoce profundamente el área para la cual se desarrollará el software; evidentemente una única persona (el analista) no puede abarcar tan vasta cantidad de áreas del conocimiento. En empresas grandes de desarrollo de productos software, es común tener analistas especializados en ciertas áreas de trabajo.

Contrariamente, no es problema del cliente, es decir él no tiene por qué saber nada de software, ni de diseños, ni otras cosas relacionadas; sólo se debe limitar a aportar objetivos, datos e información (de mano propia o de sus registros, equipos, empleados, etc.) al analista, y guiado por él, para que, en primera instancia, defina el «Universo de Discurso», y con posterior trabajo logre confeccionar el adecuado documento ERS.

Es bien conocida la presión que sufren los desarrolladores de sistemas informáticos para comprender y rescatar las necesidades de los clientes/usuarios. Cuanto más complejo es el contexto del problema más difícil es lograrlo, a veces se fuerza a los desarrolladores a tener que convertirse en casi expertos de los dominios que analizan.

Cuando esto no sucede es muy probable que se genere un conjunto de requisitos erróneos o incompletos y por lo tanto un producto de software con alto grado de desaprobación por parte de los clientes/usuarios y un altísimo costo de reingeniería y mantenimiento. Todo aquello que no se detecte, o resulte mal entendido en la etapa inicial provocará un fuerte impacto negativo en los requisitos, propagando esta corriente degradante a lo largo de todo el proceso de desarrollo e incrementando su perjuicio cuanto más tardía sea su detección (Bell y Thayer 1976) (Davis 1993).

Procesos, modelado y formas de elicitación de requisitos

Siendo que la captura, elicitación y especificación de requisitos, es una parte crucial en el proceso de desarrollo de software, ya que de esta etapa depende el logro de los objetivos finales previstos, se han ideado modelos y diversas metodologías de trabajo para estos fines. También existen herramientas software que apoyan las tareas relativas realizadas por el ingeniero en requisitos.

El estándar IEEE 830-1998 brinda una normalización de las «Prácticas Recomendadas para la Especificación de Requisitos Software».14

A medida que se obtienen los requisitos, normalmente se los va analizando, el resultado de este análisis, con o sin el cliente, se plasma en un documento, conocido como ERS o Especificación de Requisitos Software, cuya estructura puede venir definida por varios estándares, tales como CMM-I.

Un primer paso para realizar el relevamiento de información es el conocimiento y definición acertada lo que se conoce como «Universo de Discurso» del problema, que se define y entiende por:

Universo de Discurso (UdeD): es el contexto general en el cual el software deberá ser desarrollado y deberá operar. El UdeD incluye todas las fuentes de información y todas las personas relacionadas con el software. Esas personas son

conocidas también como actores de ese universo. El UdeD es la realidad circunstanciada por el conjunto de objetivos definidos por quienes demandaron el software.

A partir de la extracción y análisis de información en su ámbito se obtienen todas las especificaciones necesarias y tipos de requisitos para el futuro producto software.

El objetivo de la Ingeniería de requisitos (IR) es sistematizar el proceso de definición de requisitos permitiendo elicitar, modelar y analizar el problema, generando un compromiso entre los ingenieros de requisitos y los clientes/usuarios, ya que ambos participan en la generación y definición de los requisitos del sistema. La IR aporta un conjunto de métodos, técnicas y herramientas que asisten a los ingenieros de requisitos (analistas) para obtener requerimientos lo más seguros, veraces, completos y oportunos posibles, permitiendo básicamente:

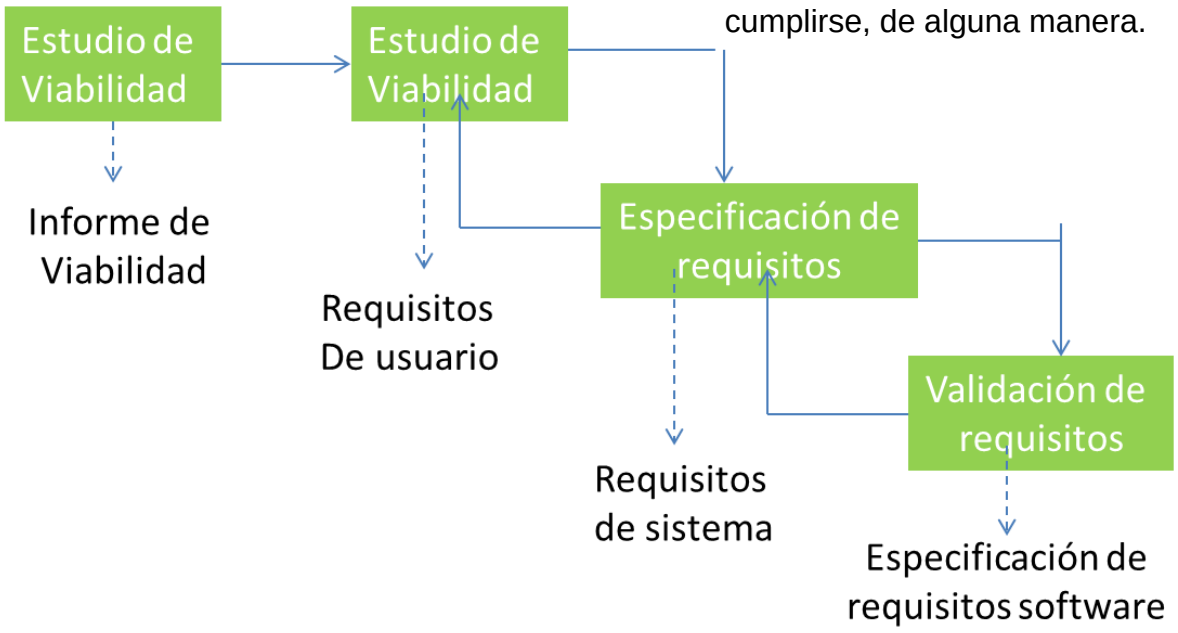
- Comprender el problema
- Facilitar la obtención de las necesidades del cliente/usuario
- Validar con el cliente/usuario
- Garantizar las especificaciones de requisitos

Si bien existen diversas formas, modelos y metodologías para elicitar, definir y documentar requerimientos, no se puede decir que alguna de ellas sea mejor o peor que la otra, suelen tener muchísimo en común, y todas cumplen el mismo objetivo. Sin embargo, lo que sí se puede decir sin dudas es que es indispensable utilizar alguna de ellas para documentar las especificaciones del futuro producto software. Así por ejemplo, hay un grupo de investigación argentino que desde hace varios años ha propuesto y estudia el uso del LEL (Léxico Extendido del Lenguaje) y Escenarios como metodología, aquí se presenta una de las tantas referencias y bibliografía sobre ello. Otra forma, más ortodoxa, de capturar y documentar requisitos se puede obtener en detalle, por ejemplo, en el trabajo de la

Universidad de Sevilla sobre «Metodología para el Análisis de Requisitos de Sistemas Software».

En la Fig. 15 se muestra un esquema, más o menos riguroso, aunque no detallado, de los pasos y tareas a seguir para realizar la captura, análisis y especificación de requerimientos software. También allí se observa qué artefacto o documento se obtiene en cada etapa del proceso. En el diagrama no se explicita metodología o modelo a utilizar, sencillamente se pautan las tareas que deben cumplirse, de alguna manera.

Figura 15: Análisis de Requerimientos



Una posible lista, general y ordenada, de tareas recomendadas para obtener la definición de lo que se debe realizar, los productos a obtener y las técnicas a emplear durante la actividad de elicitación de requisitos, en fase de Especificación de Requisitos Software es:

Una posible lista, general y ordenada, de tareas recomendadas para obtener la definición de lo que se debe realizar, los productos a obtener y las técnicas a

emplear durante la actividad de elicitación de requisitos, en fase de Especificación de Requisitos Software es:

1. Obtener información sobre el dominio del problema y el sistema actual (UdeD).
2. Preparar y realizar las reuniones para elicitación/negociación.
3. Identificar/revisar los objetivos del usuario.
4. Identificar/revisar los objetivos del sistema.
5. Identificar/revisar los requisitos de información.
6. Identificar/revisar los requisitos funcionales.
7. Identificar/revisar los requisitos no funcionales.
8. Priorizar objetivos y requisitos.

Algunos principios básicos a tener en cuenta:

- Presentar y entender cabalmente el dominio de la información del problema.
- Definir correctamente las funciones que debe realizar el Software.
- Representar el comportamiento del software a consecuencias de acontecimientos externos, particulares, incluso inesperados.
- Reconocer requisitos incompletos, ambiguos o contradictorios.
- Dividir claramente los modelos que representan la información, las funciones y comportamiento y características no funcionales.
- Definir correctamente las funciones que debe realizar el Software.

Representar el comportamiento del software a consecuencias de acontecimientos externos, particulares, incluso inesperados.

Reconocer requisitos incompletos, ambiguos o contradictorios.

Dividir claramente los modelos que representan la información, las funciones y comportamiento y características no funcionales.

Clasificación e identificación de requerimientos

Se pueden identificar dos formas de requisitos:

- Requisitos de usuario: Los requisitos de usuario son frases en lenguaje natural junto a diagramas con los servicios que el sistema debe proporcionar, así como las restricciones bajo las que debe operar.
- Requisitos de sistema: Los requisitos de sistema determinan los servicios del sistema y pero con las restricciones en detalle. Sirven como contrato.

Es decir, ambos son lo mismo, pero con distinto nivel de detalle.

Ejemplo de requisito de usuario: El sistema debe hacer préstamos Ejemplo de requisito de sistema: Función préstamo: entrada código socio, código ejemplar; salida: fecha devolución; etc.

Se clasifican en tres los tipos de requisitos de sistema:

- Requisitos funcionales

Los requisitos funcionales describen:

- Los servicios que proporciona el sistema (funciones).
- La respuesta del sistema ante determinadas entradas.
- El comportamiento del sistema en situaciones particulares.
- Requisitos no funcionales

Los requisitos no funcionales son restricciones de los servicios o funciones que ofrece el sistema (ej. cotas de tiempo, proceso de desarrollo, rendimiento, etc.)

Ejemplo 1. La biblioteca Central debe ser capaz de atender simultáneamente a todas las bibliotecas de la Universidad

Ejemplo 2. El tiempo de respuesta a una consulta remota no debe ser superior a 1/2 s

A su vez, hay tres tipos de requisitos no funcionales:

- Requisitos del producto. Especifican el comportamiento del producto (Ej. prestaciones, memoria, tasa de fallos, etc.)
- Requisitos organizativos. Se derivan de las políticas y procedimientos de las organizaciones de los clientes y desarrolladores (Ej. estándares de proceso, lenguajes de programación, etc.)
- Requisitos externos. Se derivan de factores externos al sistema y al proceso de desarrollo (Ej. requisitos legislativos, éticos, etc.)
- Requisitos del dominio.

Los requisitos del dominio se derivan del dominio de la aplicación y reflejan características de dicho dominio.

Pueden ser funcionales o no funcionales.

Ej. El sistema de biblioteca de la Universidad debe ser capaz de exportar datos mediante el Lenguaje de Intercomunicación de Bibliotecas de España (LIBE). Ej. El sistema de biblioteca no podrá acceder a bibliotecas con material censurado.

Diseño del sistema

En ingeniería de software, el diseño es una fase de ciclo de vida del software. Se basa en la especificación de requisitos producido por el análisis de los requerimientos (fase de análisis), el diseño define cómo estos requisitos se cumplirán, la estructura que debe darse al sistema de software para que se haga realidad.

El diseño sigue siendo una fase separada de la programación o codificación, esta última corresponde a la traducción en un determinado lenguaje de programación de las premisas adoptadas en el diseño.

Las distinciones entre las actividades mencionadas hasta ahora no siempre son claras cómo se quisiera en las teorías clásicas de ingeniería de software. El diseño, en particular, puede describir el funcionamiento interno de un sistema en diferentes niveles de detalle, cada una de ellos se coloca en una posición intermedia entre el análisis y codificación.

Normalmente se entiende por "diseño de la arquitectura" al diseño de "muy alto nivel", que sólo define la estructura del sistema en términos de la módulos de software de que se compone y las relaciones macroscópicas entre ellos. A este nivel de diseño pertenecen fórmulas como cliente-servidor o “tres niveles”, o, más generalmente, las decisiones sobre el uso de la arquitectura de hardware especial que se utilice, el sistema operativo, DBMS, Protocolos de red, etc.

Un nivel intermedio de detalle puede definir la descomposición del sistema en módulos, pero esta vez con una referencia más o menos explícita al modo de descomposición que ofrece el particular lenguaje de programación con el que el desarrollo se va a implementar, por ejemplo, en un diseño realizado con la tecnología de objetos, el proyecto podría describir al sistema en términos de clases y sus interrelaciones.

El diseño detallado, por último, es una descripción del sistema muy cercana a la codificación (por ejemplo, describir no sólo las clases en abstracto, sino también sus atributos y los métodos con sus tipos).

Debido a la naturaleza "intangible" del software, y dependiendo de las herramientas que se utilizan en el proceso, la frontera entre el diseño y la codificación también puede ser virtualmente imposible de identificar. Por ejemplo,

algunas herramientas CASE son capaces de generar código a partir de diagramas UML, los que describen gráficamente la estructura de un sistema software.

Codificación del software

Durante esta etapa se realizan las tareas que comúnmente se conocen como programación; que consiste, esencialmente, en llevar a código fuente, en el lenguaje de programación elegido, todo lo diseñado en la fase anterior. Esta tarea la realiza el programador, siguiendo por completo los lineamientos impuestos en el diseño y en consideración siempre a los requisitos funcionales y no funcionales (ERS) especificados en la primera etapa.

Es común pensar que la etapa de programación o codificación (algunos la llaman implementación) es la que insume la mayor parte del trabajo de desarrollo del software; sin embargo, esto puede ser relativo (y generalmente aplicable a sistemas de pequeño porte) ya que las etapas previas son cruciales, críticas y pueden llevar bastante más tiempo. Se suele hacer estimaciones de un 30% del tiempo total insumido en la programación, pero esta cifra no es consistente ya que depende en gran medida de las características del sistema, su criticidad y el lenguaje de programación elegido.7 En tanto menor es el nivel del lenguaje mayor será el tiempo de programación requerido, así por ejemplo se tardaría más tiempo en codificar un algoritmo en lenguaje ensamblador que el mismo programado en lenguaje C.

Mientras se programa la aplicación, sistema, o software en general, se realizan también tareas de depuración, esto es la labor de ir liberando al código de los errores factibles de ser hallados en esta fase (de semántica, sintáctica y lógica). Hay una suerte de solapamiento con la fase siguiente, ya que para depurar la lógica es necesario realizar pruebas unitarias, normalmente con datos de prueba; claro es que no todos los errores serán encontrados sólo en la etapa de programación, habrá otros que se encontrarán durante las etapas subsiguientes.

La aparición de algún error funcional (mala respuesta a los requerimientos) eventualmente puede llevar a retornar a la fase de diseño antes de continuar la codificación.

Durante la fase de programación, el código puede adoptar varios estados, dependiendo de la forma de trabajo y del lenguaje elegido, a saber:

Código fuente: es el escrito directamente por los programadores en editores de texto, lo cual genera el programa. Contiene el conjunto de instrucciones codificadas en algún lenguaje de alto nivel. Puede estar distribuido en paquetes, procedimientos, bibliotecas fuente, etc.

Código objeto: es el código binario o intermedio resultante de procesar con un compilador el código fuente. Consiste en una traducción completa y de una sola vez de éste último. El código objeto no es inteligible por el ser humano (normalmente es formato binario) pero tampoco es directamente ejecutable por la computadora. Se trata de una representación intermedia entre el código fuente y el código ejecutable, a los fines de un enlace final con las rutinas de biblioteca y entre procedimientos o bien para su uso con un pequeño intérprete intermedio [a modo de distintos ejemplos véase EUPHORIA, (intérprete intermedio),FORTRAN (compilador puro) MSIL (Microsoft Intermediate Language) (intérprete) y BASIC(intérprete puro, intérprete intermedio, compilador intermedio o compilador puro, depende de la versión utilizada)].

El código objeto no existe si el programador trabaja con un lenguaje a modo de intérprete puro, en este caso el mismo intérprete se encarga de traducir y ejecutar línea por línea el código fuente (de acuerdo al flujo del programa), en tiempo de ejecución. En este caso tampoco existen el o los archivos de código ejecutable. Una desventaja de esta modalidad es que la ejecución del programa o sistema es un poco más lenta que si se hiciera con un intérprete intermedio, y bastante más lenta que si existen el o los archivos de código ejecutable. Es decir no favorece el rendimiento en velocidad de ejecución. Pero una gran ventaja de la modalidad

intérprete puro, es que el esta forma de trabajo facilita enormemente la tarea de depuración del código fuente (frente a la alternativa de hacerlo con un compilador puro). Frecuentemente se suele usar una forma mixta de trabajo (si el lenguaje de programación elegido lo permite), es decir inicialmente trabajar a modo de intérprete puro, y una vez depurado el código fuente (liberado de errores) se utiliza un compilador del mismo lenguaje para obtener el código ejecutable completo, con lo cual se agiliza la depuración y la velocidad de ejecución se optimiza.

Código ejecutable: Es el código binario resultado de enlazar uno o más fragmentos de código objeto con las rutinas y bibliotecas necesarias. Constituye uno o más archivos binarios con un formato tal que el sistema operativo es capaz de cargarlo en la memoria RAM (eventualmente también parte en una memoria virtual), y proceder a su ejecución directa. Por lo anterior se dice que el código ejecutable es directamente «inteligible por la computadora». El código ejecutable, también conocido como código máquina, no existe si se programa con modalidad de «intérprete puro».

Pruebas (unitarias y de integración)

Entre las diversas pruebas que se le efectúan al software se pueden distinguir principalmente:

Prueba unitarias: Consisten en probar o testear piezas de software pequeñas; a nivel de secciones, procedimientos, funciones y módulos; aquellas que tengan funcionalidades específicas. Dichas pruebas se utilizan para asegurar el correcto funcionamiento de secciones de código, mucho más reducidas que el conjunto, y que tienen funciones concretas con cierto grado de independencia.

Pruebas de integración: Se realizan una vez que las pruebas unitarias fueron concluidas exitosamente; con éstas se intenta asegurar que el sistema completo,

incluso los subsistemas que componen las piezas individuales grandes del software funcionen correctamente al operar e inter operar en conjunto.

Las pruebas normalmente se efectúan con los llamados datos de prueba, que es un conjunto seleccionado de datos típicos a los que puede verse sometido el sistema, los módulos o los bloques de código. También se escogen: Datos que llevan a condiciones límites al software a fin de probar su tolerancia y robustez; datos de utilidad para mediciones de rendimiento; datos que provocan condiciones eventuales o particulares poco comunes y a las que el software normalmente no estará sometido pero pueden ocurrir; etc. Los «datos de prueba» no necesariamente son ficticios o «creados», pero normalmente si lo son los de poca probabilidad de ocurrencia.

Generalmente, existe una fase probatoria final y completa del software, llamada Beta Test, durante la cual el sistema instalado en condiciones normales de operación y trabajo es probado exhaustivamente a fin de encontrar errores, inestabilidades, respuestas erróneas, etc. que hayan pasado los previos controles. Estas son normalmente realizadas por personal idóneo contratado o afectado específicamente a ello. Los posibles errores encontrados se transmiten a los desarrolladores para su depuración. En el caso de software de desarrollo «a pedido», el usuario final (cliente) es el que realiza el Beta Test, teniendo para ello un período de prueba pactado con el desarrollador.

Instalación y paso a producción

La instalación del software es el proceso por el cual los programas desarrollados son transferidos apropiadamente al computador destino, inicializados, y, eventualmente, configurados; todo ello con el propósito de ser ya utilizados por el usuario final. Constituye la etapa final en el desarrollo propiamente dicho del

software. Luego de ésta el producto entrará en la fase de funcionamiento y producción, para el que fuera diseñado.

La instalación, dependiendo del sistema desarrollado, puede consistir en una simple copia al disco rígido destino (casos raros actualmente); o bien, más comúnmente, con una de complejidad intermedia en la que los distintos archivos componentes del software (ejecutables ,bibliotecas, datos propios, etc.) son descomprimidos y copiados a lugares específicos preestablecidos del disco; incluso se crean vínculos con otros productos, además del propio sistema operativo. Este último caso, comúnmente es un proceso bastante automático que es creado y guiado con herramientas software específicas (empaquetado y distribución, instaladores).

En productos de mayor complejidad, la segunda alternativa es la utilizada, pero es realizada o guiada por especialistas; puede incluso requerirse la instalación en varios y distintos computadores (instalación distribuida).

También, en software de mediana y alta complejidad normalmente es requerido un proceso de configuración y chequeo, por el cual se asignan adecuados parámetros de funcionamiento y se testea la operatividad funcional del producto.

En productos de venta masiva las instalaciones completas, si son relativamente simples, suelen ser realizadas por los propios usuarios finales (tales como sistemas operativos, paquetes de oficina, utilitarios, etc.) con herramientas propias de instalación guiada; incluso la configuración suele ser automática. En productos de diseño específico o «a medida» la instalación queda restringida, normalmente, a personas especialistas involucradas en el desarrollo del software en cuestión.

Una vez realizada exitosamente la instalación del software, el mismo pasa a la fase de producción (operatividad), durante la cual cumple las funciones para las que fue desarrollado, es decir, es finalmente utilizado por el (o los) usuario final, produciendo los resultados esperados

Mantenimiento

El mantenimiento de software es el proceso de control, mejora y optimización del software ya desarrollado e instalado, que también incluye depuración de errores y defectos que puedan haberse filtrado de la fase de pruebas de control y beta test. Esta fase es la última (antes de iterar, según el modelo empleado) que se aplica al ciclo de vida del desarrollo de software. La fase de mantenimiento es la que viene después de que el software está operativo y en producción.

De un buen diseño y documentación del desarrollo dependerá cómo será la fase de mantenimiento, tanto en costo temporal como monetario. Modificaciones realizadas a un software que fue elaborado con una documentación indebida o pobre y mal diseño puede llegar a ser tanto o más costosa que desarrollar el software desde el inicio. Por ello, es de fundamental importancia respetar debidamente todas las tareas de las fases del desarrollo y mantener adecuada y completa la documentación.

El período de la fase de mantenimiento es normalmente el mayor en todo el ciclo de vida.7 Esta fase involucra también actualizaciones y evoluciones del software; no necesariamente implica que el sistema tuvo errores. Uno o más cambios en el software, por ejemplo de adaptación o evolutivos, puede llevar incluso a rever y adaptar desde parte de las primeras fases del desarrollo inicial, alterando todas las demás; dependiendo de cuán profundos sean los cambios. El modelo cascada común es particularmente costoso en mantenimiento, ya que su rigidez implica que cualquier cambio provoca regreso a fase inicial y fuertes alteraciones en las demás fases del ciclo de vida.

Durante el período de mantenimiento, es común que surjan nuevas revisiones y versiones del producto; que lo liberan más depurado, con mayor y mejor funcionalidad, mejor rendimiento, etc. Varias son las facetas que pueden ser alteradas para provocar cambios deseables, evolutivos, adaptaciones o ampliaciones y mejoras.

Básicamente se tienen los siguientes tipos de cambios:

- Perfectivos: Aquellos que llevan a una mejora de la calidad interna del software en cualquier aspecto: Reestructuración del código, definición más clara del sistema y su documentación; optimización del rendimiento y eficiencia.
- Evolutivos: Agregados, modificaciones, incluso eliminaciones, necesarias en el software para cubrir su expansión o cambio, según las necesidades del usuario.
- Adaptivos: Modificaciones que afectan a los entornos en los que el sistema opera, tales como: Cambios de configuración del hardware (por actualización o mejora de componentes electrónicos), cambios en el software de base, en gestores de base de datos, en comunicaciones, etc.
- Correctivos: Alteraciones necesarias para corregir errores de cualquier tipo en el producto software desarrollado.

2.3. CARÁCTER EVOLUTIVO DEL SOFTWARE

El software es el producto derivado del proceso de desarrollo, según la ingeniería de software. Este producto es intrínsecamente evolutivo durante su ciclo de vida. El software evoluciona, en general, generando versiones cada vez más completas, complejas, mejoradas, optimizadas en algún aspecto, adecuadas a nuevas plataformas (sean de hardware o sistemas operativos), etc.

Cuando un sistema deja de evolucionar, eventualmente cumplirá con su ciclo de vida, entrará en obsolescencia e inevitablemente, tarde o temprano, será reemplazado por un producto nuevo.

El software evoluciona sencillamente porque se debe adaptar a los cambios del entorno, sean funcionales (exigencias de usuarios), operativos, de plataforma o arquitectura hardware.

La dinámica de evolución del software es el estudio de los cambios del sistema. La mayor contribución en esta área fue realizada por Meir M. Lehman y Belady, comenzando en los años 70 y 80. Su trabajo continuó en la década de 1990, con Lehman y otros investigadores18 de relevancia en la realimentación en los procesos de evolución (Lehman, 1996; Lehman et al., 1998; lehman et al., 2001). A partir de esos estudios propusieron un conjunto de leyes (conocidas como leyes de Lehman)9 respecto de los cambios producidos en los sistemas. Estas leyes (en realidad son hipótesis) son invariantes y ampliamente aplicables.

Lehman y Belady analizaron el crecimiento y la evolución de varios sistemas software de gran porte; derivando finalmente, según sus medidas, las siguientes ocho leyes:

- Cambio continuo: Un programa que se usa en un entorno real necesariamente debe cambiar o se volverá progresivamente menos útil en ese entorno.
- Complejidad creciente: A medida que un programa en evolución cambia, su estructura tiende a ser cada vez más compleja. Se deben dedicar recursos extras para preservar y simplificar la estructura.
- Evolución prolongada del programa: La evolución de los programas es un proceso autoregulativo. Los atributos de los sistemas, tales como tamaño, tiempo entre entregas y la cantidad de errores documentados son aproximadamente invariantes para cada entrega del sistema.
- Estabilidad organizacional: Durante el tiempo de vida de un programa, su velocidad de desarrollo es aproximadamente constante e independiente de los recursos dedicados al desarrollo del sistema.
- Conservación de la familiaridad: Durante el tiempo de vida de un sistema, el cambio incremental en cada entrega es aproximadamente constante.
- Crecimiento continuado: La funcionalidad ofrecida por los sistemas tiene que crecer continuamente para mantener la satisfacción de los usuarios.

- Decremento de la calidad: La calidad de los sistemas software comenzará a disminuir a menos que dichos sistemas se adapten a los cambios de su entorno de funcionamiento.
- Realimentación del sistema: Los procesos de evolución incorporan sistemas de realimentación multiagente y multibucle y estos deben ser tratados como sistemas de realimentación para lograr una mejora significativa del producto.

2.4. INGENIERÍA DE SOFTWARE

Ingeniería de software es aquella que ofrece métodos y técnicas para desarrollar y mantener software de calidad.

Esta ingeniería trata con áreas muy diversas de la informática y de las ciencias de la computación, tales como construcción de compiladores, sistemas operativos, o desarrollos Intranet/Internet, abordando todas las fases del ciclo de vida del desarrollo de cualquier tipo de sistemas de información y aplicables a infinidad de áreas: negocios, investigación científica, medicina, producción, logística, banca, control de tráfico, meteorología, derecho, Internet, Intranet, etc.

Una definición precisa aún no ha sido contemplada en los diccionarios, sin embargo se pueden citar las enunciadas por algunos de los más prestigiosos autores:

- Ingeniería de software es el estudio de los principios y metodologías para el desarrollo y mantenimiento de sistemas software (Zelkovitz, 1978)
- Ingeniería de software es la aplicación práctica del conocimiento científico al diseño y construcción de programas de computadora y a la documentación asociada requerida para desarrollar, operar y mantenerlos. Se conoce también como desarrollo de software o producción de software (Bohem, 1976).

- Ingeniería de software trata del establecimiento de los principios y métodos de la ingeniería a fin de obtener software de modo rentable, que sea fiable y trabaje en máquinas reales (Bauer, 1972).
- Es la aplicación de un enfoque sistemático, disciplinado y cuantificable al desarrollo, operación y mantenimiento del software; es decir, la aplicación de la ingeniería al software (IEEE, 1993).

En el 2004, en los Estados Unidos, la Oficina de Estadísticas del Trabajo (U. S. Bureau of Labor Statistics) contó 760.840 ingenieros de software de computadora.1 El término "ingeniero de software", sin embargo, se utiliza en forma genérica en el ambiente empresarial, y no todos los ingenieros de software poseen realmente títulos de ingeniería de universidades reconocidas.

Algunos autores consideran que "desarrollo de software" es un término más apropiado que "ingeniería de software" para el proceso de crear software. Personas como Pete McBreen (autor de "Software Craftmanship") cree que el término IS implica niveles de rigor y prueba de procesos que no son apropiados para todo tipo de desarrollo de software.

Indistintamente se utilizan los términos "ingeniería de software" o "ingeniería del software". En Hispanoamérica el término usado normalmente es el primero de ellos.

La creación del software es un proceso intrínsecamente creativo y la ingeniería del software trata de sistematizar este proceso con el fin de acotar el riesgo del fracaso en la consecución del objetivo creativo por medio de diversas técnicas que se han demostrado adecuadas en base a la experiencia previa.

La IS se puede considerar como la ingeniería aplicada al software, esto es, por medios sistematizados y con herramientas preestablecidas, la aplicación de ellos de la forma más eficiente para la obtención de resultados óptimos; objetivos que

siempre busca la ingeniería. No es sólo de la resolución de problemas, sino más bien teniendo en cuenta las diferentes soluciones, elegir la más apropiada.

2.4.1. IMPLICACIONES SOCIOECONÓMICAS

La ingeniería de software afecta a la economía y las sociedades de variadas formas.

Económicamente

En los Estados Unidos, el software contribuyó a una octava parte de todo el incremento del PIB durante la década de 1990 (alrededor de 90,000 millones de dólares por año), y un noveno de todo el crecimiento de productividad durante los últimos años de la década (alrededor de 33.000 millones de dólares estadounidenses por año). La ingeniería de software contribuyó a US$ 1 billón de crecimiento económico y productividad en esa década. Alrededor del globo, el software contribuye al crecimiento económico en formas similares, aunque es difícil de encontrar estadísticas fiables.

Además, con la industria del lenguaje está hallando cada vez más campos de aplicación a escala global.

Socialmente

La ingeniería de software cambia la cultura del mundo debido al extendido uso de la computadora. El correo electrónico (E-mail), la WWW y la mensajería instantánea permiten a la gente interactuar en nuevas formas. El software baja el costo y mejora la calidad de los servicios de salud, los departamentos de bomberos, las dependencias gubernamentales y otros servicios sociales. Los proyectos exitosos donde se han usado métodos de ingeniería de software

incluyen a GNU/Linux, el software del transbordador espacial, los cajeros automáticos y muchos otros.

2.4.2. Metodología

Un objetivo de décadas ha sido el encontrar procesos y metodologías, que sean sistemáticas, predecibles y repetibles, a fin de mejorar la productividad en el desarrollo y la calidad del producto software.

Etapas del proceso

La ingeniería de software requiere llevar a cabo numerosas tareas, dentro de etapas como las siguientes:

Análisis de requerimientos

Extraer los requisitos y requerimientos de un producto de software es la primera etapa para crearlo. Mientras que los clientes piensan que ellos saben lo que el software tiene que hacer, se requiere de habilidad y experiencia en la ingeniería de software para reconocer requerimientos incompletos, ambiguos o contradictorios. El resultado del análisis de requerimientos con el cliente se plasma en el documento ERS, Especificación de Requerimientos del Sistema, cuya estructura puede venir definida por varios estándares, tales como CMMI. Asimismo, se define un diagrama, en el que se plasman las principales entidades que participarán en el desarrollo del software.

La captura, análisis y especificación de requerimientos (incluso pruebas de ellos), es una parte crucial; de esta etapa depende en gran medida el logro de los objetivos finales. Se han ideado modelos y diversos procesos de trabajo para estos fines. Aunque aún no está formalizada, ya se habla de la Ingeniería de requerimientos, por ejemplo en dos capítulos del libro de Sommerville "Ingeniería del software" titulados "Requerimientos del software" y "Procesos de la Ingeniería de Requerimientos".

La IEEE Std. 830-1998 normaliza la creación de las especificaciones de requerimientos de software (Software Requirements Specification).

Especificación

La especificación de requisitos describe el comportamiento esperado en el software una vez desarrollado. Gran parte del éxito de un proyecto de software radicará en la identificación de las necesidades del negocio (definidas por la alta dirección), así como la interacción con los usuarios funcionales para la recolección, clasificación, identificación, priorización y especificación de los requisitos del software.

Entre las técnicas utilizadas para la especificación de requisitos se encuentran:

- Caso de uso,
- Historias de usuario,

Siendo los primeros más rigurosos y formales, los segundas más ágiles e informales.

Arquitectura

La integración de infraestructura, desarrollo de aplicaciones, bases de datos y herramientas gerenciales, requieren de capacidad y liderazgo para poder ser conceptualizados y proyectados a futuro, solucionando los problemas de hoy. El rol en el cual se delegan todas estas actividades es el del Arquitecto.

El arquitecto de software es la persona que añade valor a los procesos de negocios gracias a su valioso aporte de soluciones tecnológicas.

La arquitectura de sistemas en general, es una actividad de planeación, ya sea a nivel de infraestructura de red y hardware, o de software.

La arquitectura de software consiste en el diseño de componentes de una aplicación (entidades del negocio), generalmente utilizando patrones de arquitectura. El diseño arquitectónico debe permitir visualizar la interacción entre

las entidades del negocio y además poder ser validado, por ejemplo por medio de diagramas de secuencia. Un diseño arquitectónico describe en general el cómo se construirá una aplicación de software. Para ello se documenta utilizando diagramas, por ejemplo:

- Diagramas de clases
- Diagramas de base de datos
- Diagrama de despliegue
- Diagrama de secuencia

Siendo los dos primeros los mínimos necesarios para describir la arquitectura de un proyecto que iniciará a ser codificado. Depende del alcance del proyecto, complejidad y necesidades, el arquitecto elige qué diagramas elaborar.

Las herramientas para el diseño y modelado de software se denominan CASE, (Computer Aided Software Engineering) entre las cuales se encuentran:

- Enterprise Architect
- Microsoft Visio for Enterprise Architects

Programación

Reducir un diseño a código puede ser la parte más obvia del trabajo de ingeniería de software, pero no necesariamente es la que demanda mayor trabajo y ni la más complicada. La complejidad y la duración de esta etapa está íntimamente relacionada al o a los lenguajes de programación utilizados, así como al diseño previamente realizado.

Prueba

Consiste en comprobar que el software realice correctamente las tareas indicadas en la especificación del problema. Una técnica de prueba es probar por separado

cada módulo del software, y luego probarlo de forma integral, para así llegar al objetivo. Se considera una buena práctica el que las pruebas sean efectuadas por alguien distinto al desarrollador que la programó, idealmente un área de pruebas; sin perjuicio de lo anterior el programador debe hacer sus propias pruebas. En general hay dos grandes formas de organizar un área de pruebas, la primera es que esté compuesta por personal inexperto y que desconozca el tema de pruebas, de esta forma se evalúa que la documentación entregada sea de calidad, que los procesos descritos son tan claros que cualquiera puede entenderlos y el software hace las cosas tal y como están descritas. El segundo enfoque es tener un área de pruebas conformada por programadores con experiencia, personas que saben sin mayores indicaciones en qué condiciones puede fallar una aplicación y que pueden poner atención en detalles que personal inexperto no consideraría.

Documentación

Todo lo concerniente a la documentación del propio desarrollo del software y de la gestión del proyecto, pasando por modelaciones (UML),diagramas de casos de uso, pruebas, manuales de usuario, manuales técnicos, etc.; todo con el propósito de eventuales correcciones, usabilidad, mantenimiento futuro y ampliaciones al sistema.

Mantenimiento

Fase dedicada a mantener y mejorar el software para corregir errores descubiertos e incorporar nuevos requisitos. Esto puede llevar más tiempo incluso que el desarrollo del software inicial. Alrededor de 2/3 del tiempo de ciclo de vida de un proyecto2 está dedicado a su mantenimiento. Una pequeña parte de este trabajo consiste eliminar errores (bugs); siendo que la mayor parte reside en extender el sistema para incorporarle nuevas funcionalidades y hacer frente a su evolución.

2.4.3. Modelos y filosofías de desarrollo de software

La ingeniería de software dispone de varios modelos, paradigmas y filosofías de desarrollo, en los cuales se apoya para la construcción del software, entre ellos se puede citar:

- Modelo en cascada o Clásico (modelo tradicional)
- Modelo de prototipos
- Modelo en espiral
- Desarrollo por etapas
- Desarrollo iterativo y creciente o Iterativo e Incremental
- RAD (Rapid Application Development)
- Desarrollo concurrente
- Proceso Unificado
- RUP (Proceso Unificado de Rational)

Naturaleza de la IS

La ingeniería de software tiene que ver con varios campos en diferentes formas:

Matemáticas

Los programas tienen muchas propiedades matemáticas. Por ejemplo la corrección y la complejidad de muchos algoritmos son conceptos matemáticos que pueden ser rigurosamente probados. El uso de matemáticas en la IS es llamado métodos formales.

Creación

Los programas son construidos en una secuencia de pasos. El hecho de definir propiamente y llevar a cabo estos pasos, como en una línea de ensamblaje, es necesario para mejorar la productividad de los desarrolladores y la calidad final de los programas. Este punto de vista inspira los diferentes procesos y metodologías que se encuentran en la IS.

Gestión de Proyectos

El desarrollo de software de gran porte requiere una adecuada gestión del proyecto. Hay presupuestos, establecimiento de tiempos de entrega, un equipo de profesionales que liderar. Recursos (espacio de oficina, insumos, equipamiento) por adquirir. Para su administración se debe tener una clara visión y capacitación en Gestión de Proyectos.

Arte

Los programas contienen muchos elementos artísticos. Las interfaces de usuario, la codificación, etc. Incluso la decisión para un nombre de una variable o una clase. Donald Knuth es famoso por argumentar a la programación como un arte.

2.4.4. Responsabilidad

La responsabilidad en la ingeniería del software es un concepto complejo, sobre todo porque al estar los sistemas informáticos fuertemente caracterizados por su complejidad, es difícil apreciar sus consecuencias.

En la ingeniería del software la responsabilidad será compartida por un grupo grande de personas, que comprende desde el ingeniero de requisitos, hasta el arquitecto software, y contando con el diseñador, o el encargado de realizar las pruebas. Por encima de todos ellos destaca el director del proyecto. El software demanda una clara distribución de la responsabilidad entre los diferentes roles que se dan en el proceso de producción.

El ingeniero del software tiene una responsabilidad moral y legal limitada a las consecuencias directas.

2.5. PROGRAMACIÓN ORIENTADA A OBJETOS (POO)

Es un paradigma de programación o propuesta tecnológica que usa objetos y sus interacciones para diseñar aplicaciones y programas. Se basa en el uso de varios conceptos fundamentales, como clase, herencia, objeto, método, etc. Este tipo de programación también posee ciertas características, tales como la abstracción, el encapsulamiento, el polimorfismo, el principio de ocultación, entre otros. Un objeto es una abstracción de algún hecho o ente del mundo real que tiene atributos que representan sus características o propiedades, y métodos que representan su comportamiento o acciones que realizan. Todas las propiedades de métodos comunes a los objetos se encapsulan o se agrupan en clases. Una clase es una plantilla para crear objetos; por lo tanto, los objetos son instancias de las clases.

Objeto: Es una entidad provista de propiedades o atributos (datos) y de comportamiento o funcionalidad (métodos). Es muy análogo a los objetos del mundo real.

Clase: Son las definiciones de las propiedades y comportamientos de un tipo de objeto en concreto.

Método: Es un algoritmo asociado a un objeto cuya ejecución se desencadena tras la recepción de un mensaje. En últimas es lo que un objeto puede hacer.

Herencia: Es la facilidad mediante la cual una clase B hereda en ella cada uno de los atributos y operaciones de A, como si estos atributos y operaciones hubiesen sido definidos por la misma clase B; por lo tanto puede utilizar los mismos métodos y variables declaradas en A.

Polimorfismo: Es análogo a la sobrecarga de operadores en otros lenguajes de programación (C++), implica que comportamientos diferentes asociados a objetos distintos pueden compartir el mismo nombre, esto se consigue cuando al llamarlos

por ese nombre se utilice el comportamiento correspondiente al objeto que se está usando.

Encapsulamiento: Implica reunir todos los elementos que pueden considerarse pertenecientes a una misma entidad.

Modularidad: Es la propiedad que permite subdividir una aplicación en partes más pequeñas (llamadas módulos), cada una de las cuales debe ser tan independiente como sea posible de la aplicación en sí y de las partes restantes.

Ocultación: Cada objeto está aislado del exterior, cada tipo objeto expone una interfaz a los otros objetos de la clase para interactuar, esto protege las propiedades de un objeto contra la modificación de por quién no tenga derecho a acceder a ellas, solamente los propios métodos internos del objeto pueden acceder a su estado.

2.5.1. DIAGRAMA DE CLASES

Un diagrama de clases es un tipo de diagrama estático que describe la estructura de un sistema mostrando sus clases, atributos y las relaciones entre ellos. Los diagramas de clases son utilizados durante el proceso de análisis y diseño de los sistemas, donde se crea el diseño conceptual de la información que se manejará en el sistema, y los componentes que se encargaran del funcionamiento y la relación entre uno y otro.

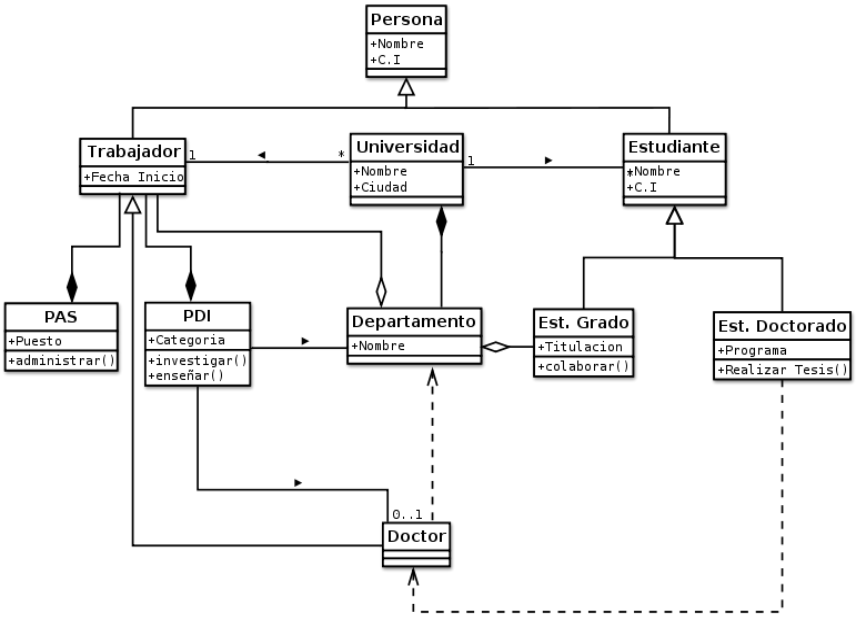
Representación de: - Requerimientos en entidades y actuaciones. - La arquitectura conceptual de un dominio - Soluciones de diseño en una arquitectura - Componentes de software orientados a objetos

Definiciones

- Propiedades, también llamados atributos o características, son valores que corresponden a un objeto, como color, material, cantidad, ubicación. Generalmente se conoce como la información detallada del objeto. Suponiendo que el objeto es una puerta, sus propiedades serían: la marca, tamaño, color y peso.

- Operaciones comúnmente llamados métodos, son aquellas actividades o verbos que se pueden realizar con/para este objeto, como por ejemplo abrir, cerrar, buscar, cancelar, acreditar, cargar. De la misma manera que el nombre de un atributo, el nombre de una operación se escribe con minúsculas si consta de una sola palabra. Si el nombre contiene más de una palabra, cada palabra será unida a la anterior y comenzará con una letra mayúscula, a excepción de la primera palabra que comenzará en minúscula. Por ejemplo: abrirPuerta, cerrarPuerta, buscarPuerta, etc.

Figura 16: Diagrama de Clases



- Interfaz es un conjunto de operaciones que permiten a un objeto comportarse de cierta manera, por lo que define los requerimientos mínimos del objeto. Hace referencia a polimorfismo.

- Herencia se define como la reutilización de un objeto padre ya definido para poder extender la funcionalidad en un objeto hijo. Los objetos hijos heredan todas las operaciones y/o propiedades de un objeto padre. Por ejemplo: Una persona puede especializarse en Proveedores, Acreedores, Clientes, Accionistas, Empleados; todos comparten datos básicos como una persona, pero además cada uno tendrá información adicional que depende del tipo de persona, como saldo del cliente, total de inversión del accionista, salario del empleado, etc.

Al diseñar una clase se debe pensar en cómo se puede identificar un objeto real, como una persona, un transporte, un documento o un paquete. Estos ejemplos de clases de objetos reales, es sobre lo que un sistema se diseña. Durante el proceso del diseño de las clases se toman las propiedades que identifican como único al objeto y otras propiedades adicionales como datos que corresponden al objeto. Con los siguientes ejemplos se definen tres objetos que se incluyen en un diagrama de clases:

Ejemplo 1: Una persona tiene número de documento de identificación, nombres, apellidos, fecha de nacimiento, género, dirección postal, posiblemente también tenga número de teléfono de casa, del móvil, FAX y correo electrónico.
Ejemplo 2: Un sistema informático puede permitir administrar la cuenta bancaria de una persona, por lo que tendrá un número de cuenta, número de identificación del propietario de la cuenta, saldo actual, moneda en la que se maneja la cuenta.
Ejemplo 3: Otro objeto pueden ser "Manejo de Cuenta", dónde las operaciones bancarias de una cuenta (como en el ejemplo 2) se manejarán realizando

diferentes operaciones que en el diagrama de clases de balurdes sólo se representan como operaciones, que pueden ser:

- Abrir
- Cerrar
- Depósito
- Retiro
- Acreditar Intereses

Estos ejemplos constituyen diferentes clases de objetos que tienen propiedades y/u operaciones que contienen un contexto y un dominio, los primeros dos ejemplos son clases de datos y el tercero clase de lógica de negocio, dependiendo de quién diseñe el sistema se pueden unir los datos con las operaciones.
El diagrama de clases incluye mucha más información como la relación entre un objeto y otro, la herencia de propiedades de otro objeto, conjuntos de operaciones/propiedades que son implementadas para una interfaz gráfica.

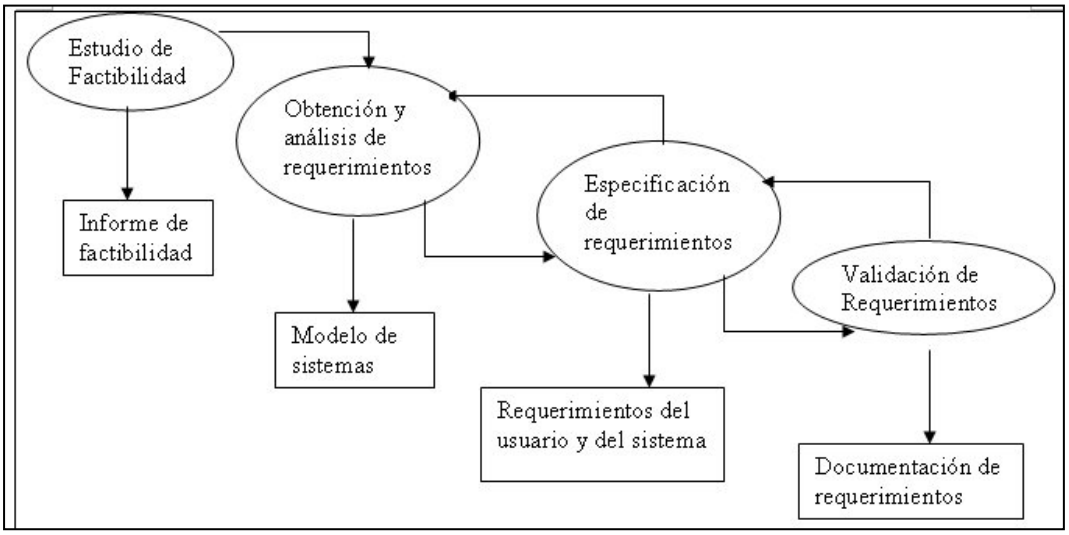
2.6. ANALISIS DE REQUERIMIENTOS

Con el fin de que un proyecto de desarrollo de software llegue a tener éxito, es necesario realizar una comprensión total de los requerimientos de la herramienta que se va a diseñar. En cada etapa de este análisis, el cliente y el desarrollador juegan un papel primordial, ya que el primero debe describir las necesidades que tiene y el segundo debe dar solución a dichas necesidades. Para llevar a cabo este objetivo, el análisis de requerimientos se divide en cuatro etapas:

- Estudio de Factibilidad.
- Obtención y Análisis de Requerimientos.
- Especificación de Requerimientos.
- Validación de Requerimientos.

Se desea realizar el diseño de una nueva herramienta de software enfocada al aprendizaje asistido del área de la Ingeniería de Petróleos denominada ANALISIS DE PRUEBAS DE PRESION. La necesidad de este proyecto nació debido a que mucho tiempo utilizado en el aprendizaje de este tópico se gasta en la realización repetitiva de gráficas y cálculos tediosos, para lo cual se propuso la creación de un software que automatizara estas labores. Por otro lado, existen en el mercado otras herramientas que ofrecen esta automatización, pero tienen un costo elevado y la mayoría de usuarios desconocen su funcionamiento interno, casi al límite de terminar utilizando una caja negra, que ciega el aprendizaje de la temática de pruebas de presión. Debido a estas dos razones, la razón de ser del proyecto es entregar una herramienta que automatice ciertas tareas repetitivas y tediosas, pero que sea flexible para que el usuario dedique este valioso tiempo en realizar tareas propias de la ingeniería y del aprendizaje de la temática de ANALISIS DE PRUEBAS DE PRESION.

Figura 17: Analisis de Requerimientos



1. Estudio de Factibilidad

Con el fin de conseguir éxito en la tarea de diseñar la herramienta, se dispuso a la realización de un estudio de factibilidad.

Se encontró que el proyecto es factible técnicamente, debido que se puede realizar con el software existente, este se planea desarrollar con un lenguaje de programación orientado a objetos, diseñado para correr en plataforma Windows. Se cuenta también con que el personal existente tiene las posibilidades de desarrollarlo, debido a que el equipo está conformado por dos estudiantes de Ingeniería de Petróleos, con buena fundamentación en análisis de pruebas de presión, programación y métodos numéricos; además de herramientas tales como equipos de cómputo propios para desarrollar el proyecto.

Se analizó la factibilidad económica, y se concluyó que también la posee, ya que el proyecto no demanda gastos monetarios elevados, debido a que se poseen equipos de cómputo, además de que se planea que sea una herramienta de software libre.

Finalmente se estudió su factibilidad operacional, la cual se comprobó debido a que la misma necesidad de creación de la herramienta y las expectativas que generó en la comunidad académica sustentaron su uso asegurado.

2. Análisis de Requerimientos

Luego de realizar la etapa de estudio de factibilidad, se comenzó con los análisis de requerimientos.

1. Debido a que la justificación del software es la creación de una herramienta de aprendizaje asistido, uno de los requerimientos es que esta sea amigable al usuario, que este último sienta que la herramienta sea de fácil acceso y operación.
2. Otro enfoque de desarrollo de la herramienta es orientado hacia un software para aprendizaje, lo que conlleva a que deben ser claros los flujos de trabajo para el análisis de las pruebas de presión, esto se logra por

medio de un trabajo conjunto entre la amigabilidad de la herramienta y un diseño claro de los manuales de usuario.

3. Ya que la herramienta es una opción de aprendizaje asistido, es necesario que el usuario tenga conocimientos previos de análisis de pruebas de presión, ya que uno de los objetivos que tiene el diseño de la herramienta es que no se convierta en una caja negra, que el usuario entienda en funcionamiento y las ecuaciones que usa la herramienta, esto se consigue con un diseño adecuado del manual de referencia del software.
4. Finalmente se espera que este proyecto sea la parte inicial del desarrollo de una herramienta más grande, que involucre tópicos más avanzados en la materia de análisis de pruebas de presión, con la ayuda de futuros desarrolladores; para lograr este objetivo, el software se debe desarrollar en un lenguaje permita la programación orientada a objetos, para que futuros trabajos permitan realizar nuevas implementaciones con módulos nuevos y mejoras en métodos numéricos.

3. Especificación de Requerimientos

En esta etapa, es necesario especificar los requisitos de una manera más técnica, separándose de las especificaciones de los clientes y acercándose más al lenguaje del desarrollador.

1. *El diseño de la interfaz de usuario deberá ser amigable al usuario*.
2. *El manual de usuario debe ser claro y conciso con respecto a los flujos de trabajo para el análisis de las pruebas de presión.*
3. *El manual de referencia debe contener todos los tópicos relacionados con los métodos de solución que utiliza el software, para que el usuario comprenda las ecuaciones utilizadas para dar solución a los análisis de las pruebas.*
4. *La herramienta de software debe estar programada en un lenguaje orientado a objetos, para posteriores mejoras y/o reparaciones.*

4. Validación de Requerimientos

1. *La interfaz del usuario debe ser visual y procedimentalmente amigable.*
2. *El manual de usuario debe tener flujos de trabajo claros.*
3. *El manual de referencia debe contener claramente todos los métodos e calculo utilizados en la herramienta.*
4. *El software debe ser programado en un lenguaje orientado a objetos.*

2.7. LICENCIA PÚBLICA GENERAL GNU, VERSION 3, 29 DE JUNIO DE 2007

Preámbulo

Las licencias para buena parte de los programas de computadora están diseñadas para limitar su libertad de compartirlos y modificarlos. En contraste, la intención de la Licencia Pública General GNU (GNU General Public License o GNU GPL) es garantizar la libertad de compartir y alterar los programas libres–asegurar que el software sea libre para todos sus usuarios. Nosotros, la Free Software Foundation, usamos la Licencia Pública General GNU para la mayor parte de nuestros programas; también se aplica a cualquier otro programa cuyos autores decidan usarla. (Algunos programas de la Free Software Foundation están cubiertos por la Licencia Pública General Menor GNU (GNU Lesser General Public License) en vez de la GNU GPL). Ud. también puede usarla para sus programas.

Cuando hablamos de software libre, hablamos de libertad, no de precio. Nuestras Licencias Públicas Generales están diseñadas para asegurar que usted tenga la libertad de distribuir copias de programas libres (y de cobrar por este servicio si lo desea), que usted reciba el código fuente o pueda obtenerlo si lo desea, que usted pueda modificar el software o usar partes de él en nuevos programas libres; y que usted sepa que puede hacer estas cosas.

Para proteger sus derechos, debemos formular requisitos que prohíban a cualquiera el negárselos, o el pedirle que renuncie a ellos. Estas restricciones se traducen en responsabilidades que Ud. deberá asumir si desea distribuir copias de un programa, o modificarlo.

Por ejemplo, si Ud. distribuye copias de uno de estos programas, ya sea en forma gratuita o percibiendo un pago por ello, Ud. deberá extender todos los derechos que Ud. tiene sobre él a quien recibe la copia. Ud. debe asegurarse de que quien recibe la copia obtenga o pueda obtener el código fuente. Y Ud. debe mostrarle estos términos, para que conozca sus derechos.

Los desarrolladores (programadores) que usan la GNU GPL protegen sus derechos en dos etapas:(1) reivindican su derecho de autor sobre el programa, y (2) le ofrecen a Ud. esta Licencia, la cual le otorga el permiso legal para copiar, distribuir y/o modificar el software.

Para protección de desarrolladores (programadores) y autores, la GPL claramente explica que no se otorgan garantías por este software libre. Si el software es modificado por alguien más y re-distribuido, la GPL asegura que los receptores hayan sido notificados que lo que ellos poseen no es el original, de modo que ninguna falla introducida por terceros afecte la reputación de los autores originales.

Algunos países han adoptado leyes que prohíben software que permita a los usuarios escapar de medidas de Gestión Digital de Restricciones (DRM). Las DRM son fundamentalmente incompatibles con el propósito de la GPL, que es proteger la libertad de los usuarios; por consiguiente, la GPL asegura que el software que cubre no va a estar sujeto, ni va a sujetar a otras obras a restricciones digitales de las cuales esté prohibido escapar.

Finalmente, todos los programas son amenazados constantemente por las patentes de software. Es nuestra intención anular el peligro de que redistribuidores de un programa libre obtengan licencias de patentes de manera individual, con el efecto de convertir el programa en privativo. Para impedir esto, la GPL deja

claramente establecido que cualquier patente debe ser licenciada de forma tal que todos puedan usarla libremente, o no ser licenciada en absoluto.
Los términos y condiciones precisos para la copia, distribución y modificación son los que siguen

0. Definiciones

Un "programa licenciado" significa cualquier programa u obra distribuida bajo esta Licencia. El "Programa" refiere a cualquiera de estos de programas u obras, y una "obra basada en el Programa" significa indistintamente el Programa o cualquier obra derivada de él según las leyes de derecho de autor: es decir, una obra que contenga el Programa o una parte de él, ya sea modificado o sin modificar. A lo largo de esta licencia, el término "modificación" incluye, sin limitaciones, la traducción y la extensión. Una "obra cubierta" significa tanto el Programa como cualquier obra basada en el Programa. Cada licenciatario será mencionado como "Ud."
"Propagar" una obra significa hacer con ella cualquier cosa que requiera permiso bajo la ley de derecho de autor aplicable, excepto ejecutarla en una computadora o realizar modificaciones privadas. Esto incluye la copia, distribución (con o sin modificaciones), sub-licenciamiento y, en algunos países, otras actividades también"".

1. Código Fuente

El "código fuente" de una obra significa aquella representación de la obra que es la preferida para modificarla. "Código objeto" significa cualquier versión no fuente de una obra.
El "Código Fuente Completo Correspondiente" una obra disponible en código objeto significa todo el código fuente necesario para comprender, adaptar, modificar, compilar, enlazar, instalar, y ejecutar la obra, excluyendo herramientas de propósito general usadas para llevar adelante estas actividades, las que no

forman parte de la obra. Por ejemplo, esto incluye cualquier todos los scripts usados para controlar esas actividades, y todas las bibliotecas compartidas o sub programas que sean enlazados dinámicamente que el programa requiera por su diseño, ya sea mediante comunicación íntima de datos o control de flujo entre esos subprogramas y otras partes de la obra, y los archivos de definición de interfaces asociados con los archivos de código fuente del programa.

El Código Fuente Completo Correspondiente también incluye todos los códigos de autorización o cifrado necesarios para instalar y/o ejecutar el código fuente de la obra, quizás modificado por Ud., en su contexto de uso recomendado o principal, de forma tal que su funcionamiento sea idéntico en todas circunstancias al de la obra, salvo por las alteraciones introducidas por Ud. Esto también incluye cualquier código de descifrado necesario para acceder a o descifrar la salida de la obra. A pesar de esto, no será necesario incluir estos códigos en aquellos casos en los que el uso de la obra normalmente implique que el usuario ya dispone de ellos.

El Código Fuente Completo Correspondiente no necesita incluir algo que el usuario pueda regenerar automáticamente a partir de otras partes del Código Fuente Completo Correspondiente.

Como excepción, el Código Fuente Completo Correspondiente no necesita incluir una subunidad en particular si (a) la subunidad idéntica es normalmente incluida como adjunto en la distribución de un componente esencial (núcleo, sistema de ventanas, etc.) del sistema operativo en el cual corre el ejecutable, o del compilador usado para producir el ejecutable, o de un intérprete de código objeto utilizado para correrlo, y (b) la subunidad (más allá de posibles extensiones) sirve únicamente para permitir el uso de la obra con esa componente del sistema o compilador o intérprete, o para implementar una interfaz estándar o ampliamente usada, cuya implementación no requiera ninguna licencia de patentes que no esté disponible en forma general para programas bajo esta licencia.

2. Permisos básicos

Todos los derechos otorgados bajo esta licencia son otorgados mientras dure el derecho del autor sobre el Programa, y son irrevocables en tanto y en cuanto las condiciones indicadas sean satisfechas. Esta licencia afirma explícitamente su libertad ilimitada para usar el Programa. Los datos producidos al ejecutarlo están cubiertos por esta licencia sólo si, dado su contenido, éstos constituyen una obra derivada del Programa. Esta licencia reconoce sus derechos de "uso justo" u otro equivalente, de acuerdo a la legislación de derechos de autor.

Esta licencia otorga permisos ilimitados para modificar y ejecutar el Programa en forma privada, siempre que usted no inicie pleito por violación de patentes contra alguien por hacer, usar o distribuir sus propias obras basadas en el Programa.

La Propagación de obras cubiertas está permitida sin límites, siempre y cuando no habilite a terceras partes a confeccionar o recibir copias. La Propagación que habilita a terceros a confeccionar o recibir copias está permitida, como "distribución", bajo las condiciones de las secciones 4 a 6, más abajo.

3. Gestión Digital de Restricciones

Como licencia de software libre, esta licencia desaprueba intrínsecamente los intentos de restringir por medios técnicos la libertad de los usuarios para copiar, modificar, y compartir obras que se encuentren bajo derechos de autor. Cada una de sus previsiones deberá ser interpretada a la luz de esta declaración específica de las intenciones de quienes otorgan esta Licencia. Sin consideración de cualquier otra previsión de esta licencia, no se otorgan permisos para distribuir obras cubiertas que invadan ilegalmente la privacidad de los usuarios, ni para modos de distribución que nieguen a los usuarios que ejecuten obras cubiertas el pleno ejercicio de los derechos legales otorgados por esta licencia.

Ninguna obra cubierta constituye parte de una medida efectiva de protección tecnológica: es decir, la distribución de una obra cubierta como parte de un sistema para generar o acceder a ciertos datos constituye un permiso general al

menos para el desarrollo, distribución y uso, bajo esta licencia, de otros programas capaces de acceder a los mismos datos.

4. Copias Idénticas

Usted puede copiar y distribuir copias idénticas del código fuente del programa tal como usted lo recibe, en cualquier medio, a condición de que publique en cada copia, de forma visible y apropiada, una nota de derecho de autor; conserve intacta toda nota de licencia y de ausencia de garantía; entregue a todos quienes reciben el Programa una copia de esta Licencia; y obedezca todas las condiciones adicionales presentes sobre partes del Programa de acuerdo con la sección 7.
Usted puede percibir una tarifa por el acto físico de transferir una copia, y tiene la opción de ofrecer garantías a cambio de remuneración.

5. Distribución de versiones modificadas del Código Fuente

Habiendo modificado una copia del Programa bajo las condiciones de la sección 2, produciendo así una obra basada en el Programa, Ud. podrá copiar y distribuir esas modificaciones u obra como código fuente bajo los términos de la sección 4 más arriba, a condición de que usted también cumpla con todas estas condiciones:

1. La obra modificada debe contener prominentes indicaciones de que Ud. modificó la obra y la fecha de esos cambios.

2. Ud. debe licenciar la obra modificada, como un todo, bajo esta Licencia, a cualquier persona que obtenga una copia. Esta Licencia debe aplicarse, sin modificaciones excepto lo permitido en la sección 7 más abajo, para toda la obra. Esta Licencia no permite licenciar esta obra de ninguna otra forma, pero no invalida permisos a ese efecto si Ud. lo recibió en forma separada.

3. Si la obra modificada tiene interfaces de usuario, cada una debe incluir una función fácil de acceder que muestre una nota de derechos de autor apropiada, y

le diga a los usuarios que no hay garantía para el programa (o que Ud. provee una garantía), que los usuarios pueden redistribuir la obra modificada bajo estas condiciones, y cómo ver una copia de esta licencia junto a la lista central (si la hay) de términos adicionales de acuerdo con la sección 7. Si la interface tiene una lista de comandos u opciones accesibles al usuario, tal como un menú, el comando para ver esta información debe figurar prominentemente en la lista. En cualquier otro caso, la obra modificada debe mostrar esta información al inicio – excepto en el caso de que el Programa tenga tales modos interactivos y no muestre esta información al inicio.

Estos requisitos se aplican a la obra modificada como un todo. Si hay secciones identificables agregadas por Ud. a esa obra que no son derivadas del Programa, y que pueden ser razonablemente consideradas obras independientes y separadas en sí mismas, entonces, esta licencia, y sus términos no se aplican a esas secciones cuando las distribuya como obra separada para ser usada sin combinación con el Programa. Pero cuando usted distribuye las mismas secciones para su uso que en combinación con obras cubiertas, independientemente de cómo se realice dicha combinación, la combinación completa debe ser licenciada bajo esta Licencia, cuyos permisos para otros licenciatarios se extienden a la obra completa y entonces a cada parte del todo. Sus secciones pueden estar sujetas a otros términos como parte de esta combinación en algunas formas limitadas, según se describe en la sección 7.

Así, no es intención de esta sección reclamar o refutar sus derechos sobre una obra completamente escrita por usted; en cambio, la intención es ejercer el derecho de controlar la distribución de obras derivadas o de obras colectivas basadas en el Programa.

Una compilación de obras cubiertas con otras obras separadas e independientes, que no son por su naturaleza extensiones de la obra cubierta, en una unidad de almacenamiento o medio de distribución, se denomina "conjunto" si el derecho de autor resultante de la compilación no se usa para limitar los derechos legales de

los usuarios de la compilación más allá de lo que la obra individual permite. La simple inclusión de una obra cubierta en un conjunto no hace que esta Licencia se aplique a las otras partes del conjunto.

6. Distribución sin Código Fuente

Usted puede copiar y distribuir una obra cubierta en su forma de Código Objeto bajo los términos de las Secciones 4 y 5, a condición de que también distribuya, en forma apropiada para ser leída por una máquina, el Código Fuente Completo Correspondiente (en adelante "Fuente Correspondiente") bajo los términos de esta Licencia, a través de una de las siguientes vías:

1. Distribuir el Código Objeto en un producto físico (incluyendo medios físicos de distribución), acompañado por el Fuente Correspondiente en un medio físico durable de los habitualmente usados para intercambiar programas; o,

2. Distribuir el Código Objeto en un producto físico (incluyendo medios físicos de distribución), acompañado por una oferta escrita, válida durante un mínimo de tres años y válida durante tanto tiempo como Ud. ofrezca servicio técnico o repuestos para ese modelo del producto, de entregar a cualquier tercero, por un precio no superior a diez veces sus costos de realizar la distribución física del código fuente, una copia del Fuente Correspondiente para todos los programas contenidos en el producto que se encuentren cubiertos por esta Licencia, en un medio físico durable de los habitualmente usados para intercambiar programas; o,

3. Distribuir en forma privada el Código Objeto con una copia de la oferta escrita de proveer el Fuente Correspondiente. Esta alternativa está permitida sólo para la distribución no comercial ocasional, y sólo en el caso de que Ud. haya recibido el Código Objeto con dicha oferta, de acuerdo a la Subsección b más arriba. O,

4. Distribuir el Código Objeto ofreciendo acceso para copiarlo desde un lugar designado, y ofrecer acceso equivalente al Fuente Correspondiente en la misma forma y a través del mismo lugar. Ud. no necesita exigir a quien recibe el programa que también copie el Fuente Correspondiente junto con el Código Objeto. [Si el lugar para copiar el Código Objeto es un servidor de red, el Fuente

Correspondiente puede estar disponible en un servidor diferente que ofrezca las mismas facilidades de copia, siempre y cuando que Ud. haya explícitamente acordado con el operador de dicho servidor que el Fuente Correspondiente permanecerá disponible por todo el tiempo que sea necesario para satisfacer estos requerimientos, y siempre y cuando usted mantenga, junto al Código Objeto, instrucciones claras que digan dónde se puede encontrar el Fuente Correspondiente.

La distribución del Fuente Correspondiente de acuerdo con esta sección debe ser realizada en un formato documentado públicamente y libre de patentes, y no debe requerir ninguna clave o contraseña especial para desempaquetarlo, leerlo o copiarlo.

El Fuente Correspondiente puede incluir partes que no estén licenciadas formalmente bajo esta Licencia, pero que califiquen para su inclusión en una obra bajo esta Licencia según los términos de la sección 7.

7. Compatibilidad con otras licencias

En el momento de publicar una obra basada en el Programa, Ud. puede incluir sus propios términos cubriendo las partes añadidas sobre las que Ud. tiene, o puede ofrecer, los apropiados permisos de derecho de autor, siempre y cuando esos términos permitan claramente todas las actividades que esta Licencia permite, o permitan el uso o re licenciamiento bajo esta Licencia. Sus términos pueden estar escritos por separado o pueden ser esta Licencia más permisos añadidos. Si Ud. licencia de esta manera sus propias partes añadidas, estas partes añadidas pueden ser usadas en forma separada bajo sus términos, pero la obra completa permanece bajo esta Licencia. Aquellos que copien la obra, u obras derivadas de ésta, deben respetar los términos fijados por Ud. de la misma manera que deben respetar esta Licencia, siempre y cuando una porción substancial de las partes añadidas a las que se aplican esté presente.

Además de permisos adicionales, sus términos pueden agregar ciertos tipos limitados de requisitos en las partes que agregue, como sigue:

1. Pueden requerir la preservación de ciertos avisos de derechos de autor, otros avisos legales, y/o atribuciones de autoría, y pueden requerir que el origen de las partes que cubren no sea tergiversada, y/o que las versiones alteradas sean marcadas en el código fuente, o que se las marque de manera específica y razonable como diferentes de la versión original.

2. Pueden indicar una negación de garantía y responsabilidad civil en términos diferentes a los usados en esta Licencia.

3. Pueden prohibir o limitar el uso de nombres específicos de contribuidores para propósitos publicitarios, y pueden requerir que ciertas marcas registradas específicas sean usadas con propósitos publicitarios solamente en las formas entendidas como "uso justo" por las leyes de marcas registradas, salvo que exista un permiso específico.

4. Pueden requerir que la obra contenga mecanismos funcionales que permitan a los usuarios obtener inmediatamente copias del Código Fuente Correspondiente Completo.

5. Pueden imponer represalias contra patentes de software, es decir que los permisos para el uso de las partes agregadas por Ud. caducan o pueden ser rescindidos total o parcialmente cuando se den condiciones específicas, para usuarios estrechamente ligados con cualquiera que haya realizado una demanda por patente de software (esto es, una demanda alegando que algún programa viola una patente). Las condiciones deben limitar las represalias a un subconjunto de estos dos casos: 1. Demandas que carecen de la justificación de ser en sí mismas represalias contra otras demandas por patentes de software que a su vez carecen de tal justificación. 2. Demandas que afectan a partes de esta obra, u otro código que fuera publicado en alguna otra parte junto con los agregados de Ud. y que en su conjunto se encontrara bajo los términos usados aquí para esas partes.

Ninguna otra condición adicional está permitida en los términos que Ud. agregue; por lo tanto, ninguna otra condición puede estar presente en obras que usen esta Licencia. Esta Licencia no intenta reforzar sus términos o definir si son válidos o exigibles por usted, simplemente no le prohíbe a Ud. emplearlos.

Cuando otros modifican la obra modificando las partes aportadas por Ud., pueden publicar esas partes de sus versiones bajo esta Licencia sin los permisos adicionales, con sólo incluir una nota al respecto, o eliminando la nota que ofrece permisos específicos agregados a esta Licencia. Entonces, cualquier permiso más amplio otorgado por sus términos que no estén otorgados en esta Licencia no se aplicará a las modificaciones de estos terceros, o a las versiones modificadas de las partes resultantes de esas modificaciones. Por el contrario, los requisitos específicos de sus términos seguirán vigentes en cualquier obra que sea derivada de las partes que usted agregó.

Salvo que la obra también permita distribución bajo alguna versión anterior de esta Licencia, todos los otros términos incluidos en la obra bajo esta sección deben ser listados, juntos, en una lista central en la obra.

8. Rescisión

Usted no tiene permitido propagar, modificar o sublicenciar el Programa salvo de la forma expresamente indicada en esta Licencia. Cualquier otro intento de propagar, modificar o sublicenciar el Programa es inválido, y cualquier titular del derecho de autor puede rescindir los derechos otorgados a Ud. bajo esta Licencia en cualquier momento después de haberlo notificado de la violación, a través de cualquier medio razonable, dentro de los 60 días cada instancia. Aun así, las terceras partes que hubieran recibo copias o derechos de usted bajo esta Licencia no verán rescindidos sus términos en tanto y en cuanto se mantengan en completo cumplimiento de sus condiciones.

9. No es un contrato

Usted no está obligado a aceptar esta Licencia para recibir una copia del Programa. Sin embargo, ninguna otra cosa le concede permiso para propagar o modificar el Programa o cualquier obra cubierta. Estas acciones violan los derechos de autor si no acepta esta Licencia. Por lo tanto, al modificar o propagar el Programa (o cualquier obra cubierta), Ud. declara su aceptación de esta Licencia para poder hacerlo, y de todos sus términos y condiciones.

10. Licenciamiento automático para receptores

Cada vez que Ud. redistribuye una obra cubierta, el receptor automáticamente recibe una licencia de los licenciadores originales para propagar y modificar la obra, sujeta a esta Licencia, incluyendo los términos adicionales incluidos de acuerdo con la Sección 7. Ud. no puede imponer ninguna restricción adicional al ejercicio de los derechos otorgados a los receptores, excepto (si usted modificó la obra) en las formas limitadas permitidas por la Sección 7. Ud. no es responsable de controlar y exigir el cumplimiento de esta Licencia por terceros.

11. Licencias de patentes

Cuando Ud. distribuye una obra cubierta, está otorgando una licencia de patentes al receptor, y a todo aquel que reciba cualquier versión de la obra permitiendo, para cualquiera y para todas las versiones de la obra cubierta, todas las actividades permitidas o contempladas por esta Licencia, tales como instalar, ejecutar, distribuir versiones de la obra y usar los datos que produzca. Esta licencia de patente es no exclusiva, libre de regalías y válida en todo el mundo, y cubre todas las declaraciones de patentes que estén bajo su control o que Ud. tenga derecho a sublicenciar, ya sea al momento en que distribuye la obra cubierta o en el futuro, que puedan ser violadas por la obra cubierta o por cualquier uso de la obra cubierta que sea contemplado como razonable.

Si Ud. distribuye una obra cubierta sabiendo que lo hace confiando en una licencia de patente que se lo permite, Ud. debe actuar para proteger a los usuarios que

reciban copias directamente o indirectamente a través de Ud. de posibles demandas por violación de patentes, demandas de las cuales Ud. está protegido por su licencia.

12. Libertad o muerte para el programa

Si a Ud. se le imponen condiciones (sea por orden judicial, acuerdo o lo que fuera) que contradicen las condiciones de esta Licencia, estas condiciones no lo excusan de las condiciones de esta Licencia. Si Ud. no puede distribuir el Programa u otra obra cubierta, de forma de satisfacer simultáneamente sus obligaciones bajo esta Licencia y cualquier otra obligación pertinente, entonces, como consecuencia, Ud. no tiene permiso para distribuir el programa en absoluto. Por ejemplo, si una licencia de patente no le permite la redistribución libre de regalías a todos aquellos que reciben una copia directa o indirectamente a través de Ud., entonces la única forma en la cual Ud. pueda satisfacer tanto la licencia de patente como esta Licencia es abstenerse completamente de la distribución.

No es objetivo esta sección inducirlo a violar patentes u otros derechos exclusivos o desafiar su validez legal. El único propósito de esta sección es proteger la integridad del sistema de distribución del software libre. Mucha gente ha hecho generosas contribuciones a la gran variedad de programas distribuidos mediante este sistema, confiando en que dicho sistema será aplicado consistentemente; la decisión acerca de si está dispuesto a distribuir programas a través de otros sistemas corresponde al autor/donante, y un licenciatario no puede imponer su propia decisión.

13. Límites geográficos

Si la distribución y/o uso del Programa está restringida en ciertos países por patentes o interfaces cubiertas por derechos de autor, el titular original del derecho de autor que publica el Programa bajo esta Licencia puede agregar una limitación geográfica explícita a la distribución, excluyendo esos países de forma que la distribución sólo sea permitida en o entre los países que no estén excluidos. En

ese caso, esta Licencia incorporará la limitación como si estuviese escrita en el cuerpo de esta Licencia.

14. Revisiones de esta licencia

De tanto en tanto, la Free Software Foundation puede publicar versiones nuevas y/o modificadas de la Licencia Pública General GNU. Estas nuevas versiones serán similares en espíritu a esta versión, pero pueden diferir en detalles para enfrentar nuevos problemas o preocupaciones.

Cada versión recibe un número de versión distintivo. Si el Programa especifica que se le aplica una determinada versión numerada de esta Licencia "o cualquier versión posterior", Ud. tiene la opción de seguir los términos y condiciones de esa versión específica o de cualquier versión posterior publicada por la Free Software Foundation. Si el Programa no especifica un número de versión de esta Licencia, Ud. puede elegir cualquier versión publicada por la Free Software Foundation.

15. Solicitud de excepciones

Si Ud. desea incorporar partes del Programa en otros programas libres cuyas condiciones de distribución sean diferentes, escriba al autor para solicitar su permiso. Si el titular de los derechos de autor es la Free Software Foundation, diríjase a la Free Software Foundation: algunas veces hacemos excepciones de este tipo. Nuestra decisión estará guiada por el doble objetivo de preservar la libertad de todas las obras derivadas de nuestro software libre y de promover el acto de compartir y reutilizar el software en general.

16. SIN GARANTÍAS

No hay garantía para el programa, hasta donde esté permitido excluirla por la legislación aplicable. Salvo indicación en contrario, el titular del derecho de autor del Programa y/u otras partes proveen el programa "tal cual es" sin garantías de ningún tipo, expresas o implícitas, incluyendo pero no limitándose a las garantías implícitas de comerciabilidad, usabilidad o utilidad para un propósito particular.

Todo el riesgo referente al uso y la calidad de programa es suyo. Si el programa resulta ser defectuoso, Ud. asume todos los costos necesarios en términos de servicios, reparaciones o correcciones.

17.

Salvo que cuando así sea requerido por las leyes aplicables o se acuerde por escrito, ni los titulares del derecho de autor, ni cualquier tercera parte que haya modificado y/o redistribuido el programa, serán responsables ante Ud. por daños, incluyendo cualquier daño general, especial, incidental o consecuente que se derive del uso o incapacidad del uso de este programa (incluyendo, pero no limitado a, la pérdida de datos o de precisión en los datos o pérdidas sufridas por Ud. o una tercera parte, o una incapacidad del programa para operar junto con otros programas), incluso si el derecho habiente u otra parte ha sido puesto sobre aviso sobre la posibilidad de tales daños.

18.

A menos que haya una mención específica al respecto, el Programa no ha sido probado para su uso en sistemas críticos a la seguridad.

2.8. LIBRERÍA GRAFICA DE LA HERRAMIENTA

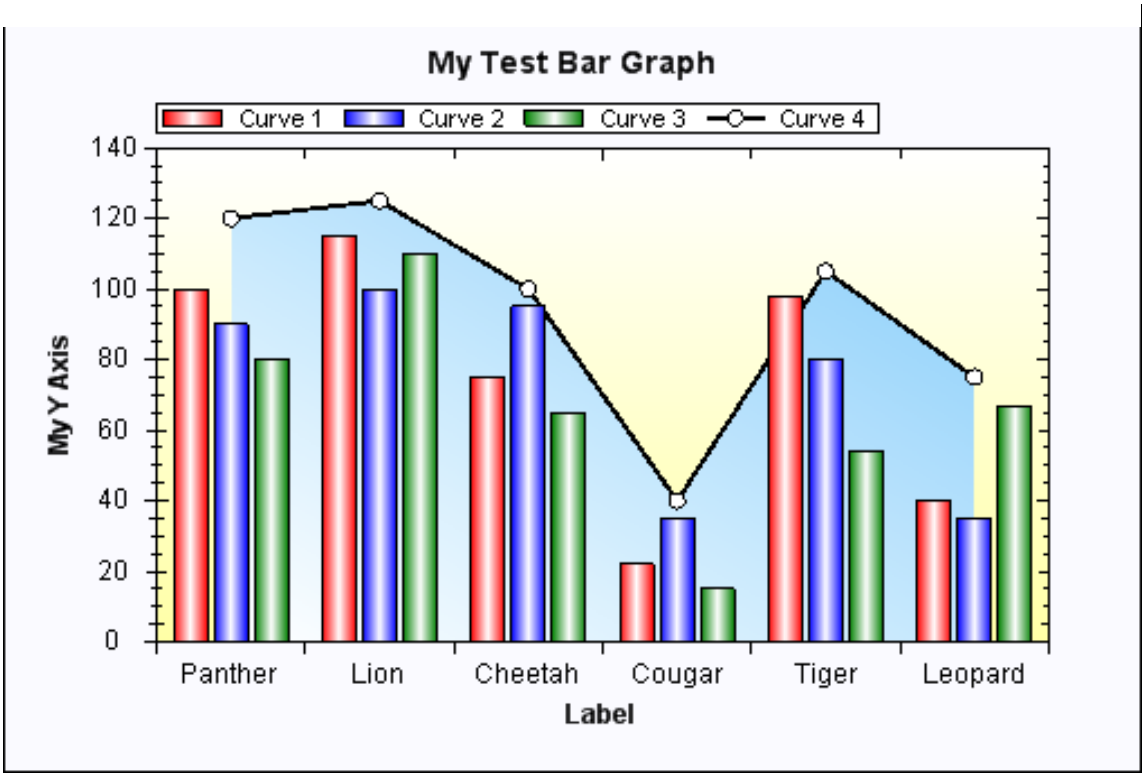
Con el fin de que la herramienta cumpliera el objetivo de ser amigable al usuario, era necesaria la utilización de un componente grafico en el software. Para esta tarea se eligió una librería grafica libre de Source Force denominada ZedGraph.dll, la cual se encargaría de generar las gráficas en dos dimensiones para los análisis necesarios de las pruebas de presión cargadas en la herramienta.

Esta librería de clases escrita en C# es útil para dibujar líneas en dos dimensiones, gráficos de barras y de pastel. Proporciona un grado de flexibilidad en los aspectos característicos de cada gráfica. Esta librería de clases permite su

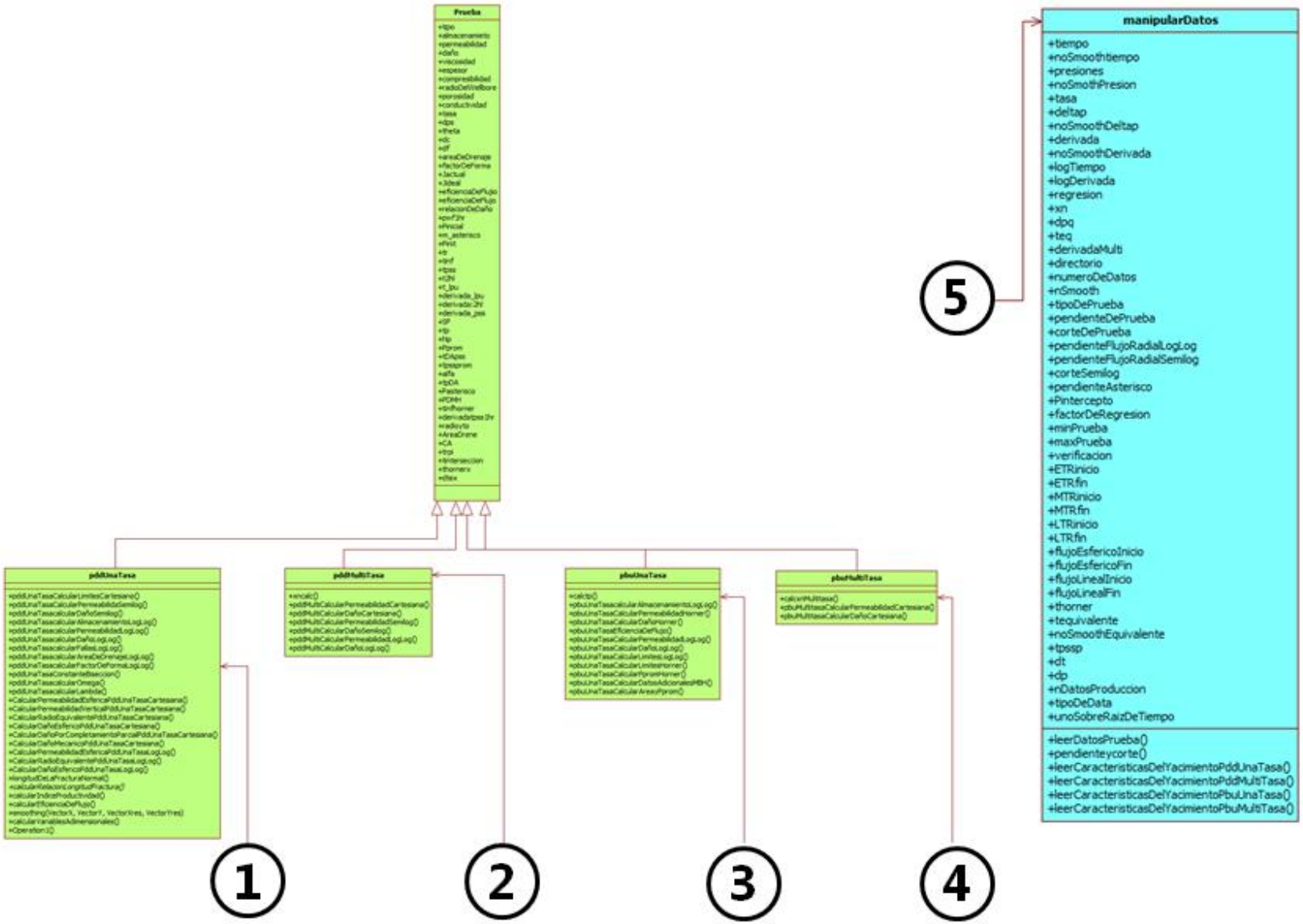
utilización en lenguajes como C#, C++ y Visual Basic, y funciona solamente para plataforma Windows.

A pesar de que la herramienta está diseñada para lenguajes orientados a objetos, su flexibilidad en este aspecto no es tan clara al intentar manipularla. Otro aspecto desfavorable son sus capacidades graficas algo limitadas. Estos aspectos fueron tenidos en cuenta al escoger esa librería de clases para incluirla en la herramienta, la razón principal de esta elección fue la facilidad de manipulación y sus controles sencillos para tareas sencillas, los cuales no se ofrecen al utilizar librerías graficas más potentes tales como OpenGL y W3.

Figura 18: Grafica Generada por la Libreria Grafica zedgraph.dll



2.9. DIAGRAMA UML


Prueba
pdd1unaTasa
pdd1MultiTasa
pbu1unaTasa
pbu1MultiTasa
manipularDatos
+tiempo
+noSmoothtiempo
+presiones
+noSmothPresion
+tasa
+deltap
+noSmoothDeltap
+derivada
+noSmoothDerivada
+logTiempo
+logDerivada
+regresion
+xn
+dpq
+teq
+derivadaMulti
+directorio
+numeroDeDatos
+nSmooth
+tipoDePrueba
+pendienteDePrueba
+corteDePrueba
+pendienteFlujoRadialLogLog
+pendienteFlujoRadialSemilog
+corteSemilog
+pendienteAsterisco
+Pintercepto
+factorDeRegresion
+miniPrueba
+maxPrueba
+verificacion
+ETRinicio
+ETR.fin
+MTRinicio
+MTR.fin
+LTRinicio
+LTR.fin
+flujoEsfericoInicio
+flujoEsfericoFin
+flujoLinealInicio
+flujoLinealFin
+thorner
+tequivalente
+noSmoothEquivalente
+tpssp
+dt
+dp
+nDatosProduccion
+tipoDeData
+unoSobreRaizDeTiempo
+leerDatosPrueba()
+pendienteycorte()
+leerCaracteristicasDelYacimientoPdd1unaTasa()
+leerCaracteristicasDelYacimientoPdd1MultiTasa()
+leerCaracteristicasDelYacimientoPbu1unaTasa()
+leerCaracteristicasDelYacimientoPbu1MultiTasa()
1
2
3
4
5

3. COMPARACION DEL SOFTWARE.

Con el objetivo de evaluar la herramienta de software se llevó a cabo comparación con 6 pruebas, 2 pruebas fueron generadas con un simulador de yacimientos, otras 2 se desarrollaron con pruebas reales de presión y la última se obtuvo de la generación de un modelo tipo black oil, de una sola capa, sin capa de gas, sin empuje de acuífero y con saturación de agua crítica. A continuación se muestran los resultados y una breve descripción sobre las mismas.

3.1. COMPARACION CON PRUEBAS SINTETICAS DE PRESION

3.1.1. CASO 1: Pozo Parcialmente Cañoneado

Se realizó la validación con datos de una prueba de presión de la literatura correspondiente a un pozo parcialmente cañoneado, a continuación se muestran los parámetros de la prueba, yacimiento y pozos usados:

Tabla 1:Parámetros de la Prueba, Yacimiento y Pozo-Caso1

Datos de la Prueba:		
h	**302**	**Ft**
B	**1.7**	**rb/STB**
r_w	**0.246**	**Ft**
ϕ	**20**	**%**
μ	**0.21**	**cP**
C_t	**3.42E-05**	**Psi^{-1}**
k	**8.4**	**md**
S	**-5.3**	

Al analizar los resultados, los valores calculados por la herramienta presentan errores bajos con respecto a los valores tomados de la literatura.

Figura 19: Grafica Flujo Lineal

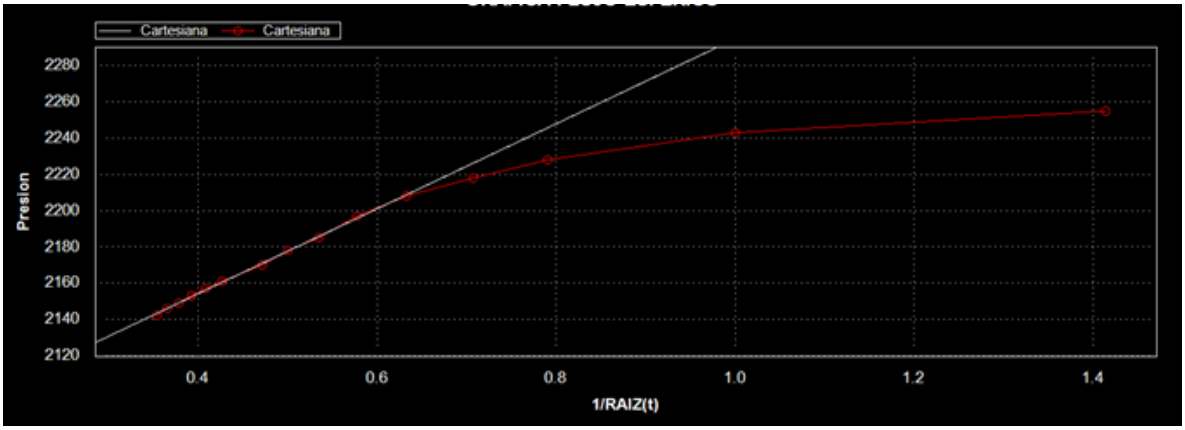


Tabla 2: Resultados Prueba de Literatura de un Pozo Parcialmente Cañoneado

Parámetros	Datos Literatura	Datos AL-PTA			
		Análisis Convencional	Error Relativo Porcentual	Análisis Moderno	Error Relativo Porcentual
k(md)	7.97	8.4	5.35%	8.724	9.41%
S	-5.04	-5.3	5.01%	-5.339	5.78%
Ksp (md)	8.13	8.151	0.26%	7.979	1.86%
Ssp	-0.85	-1.001	16.90%	-0.877	2.42%
kv (md)	8.45	7.116	15.81%	6.674	21.04%
rsw (ft)	9.69	9.693	0.00%	9.693	0.00%
Sc	8.41	8.612	2.37%	8.66	2.94%
Sm	NA	-5.146	NA	-5.214	NA
ST	NA	8.742	NA	8.724	NA

3.1.2. CASO 2: Pozo Fracturado Hidráulicamente

Se realizó la validación con datos de una prueba de presión de la literatura correspondiente a un pozo fracturado hidráulicamente, a continuación se muestran los parámetros de la prueba, yacimiento y pozos usados:

Tabla 3: Parámetros de la Prueba, Yacimiento y Pozo-Caso2

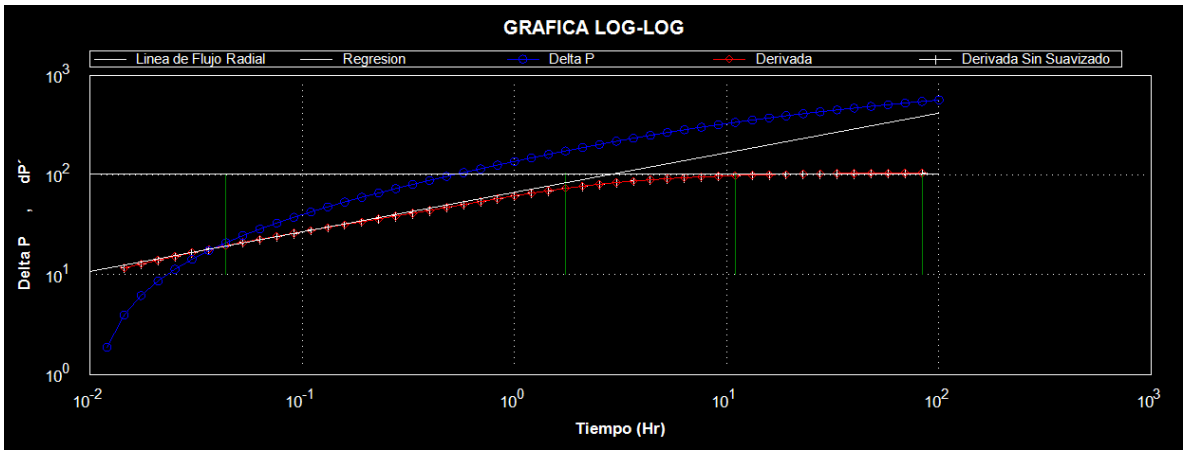
Datos de la Prueba:		
h	50	Ft
B	1.5	rb/STB
rw	0.4	Ft
ϕ	0.24	%
μ	0.4	cP
Ct	1.48E-5	Psi-1
k	12.20	md
S	-4.65	

Al analizar los resultados, los valores calculados por la herramienta presentan errores algo desfasados con respecto a los valores tomados de la literatura para la longitud de la fractura, pero la estimación de la permeabilidad y el daño son aceptables.

Tabla 4: Resultados Prueba de Literatura de un Pozo Fracturado Hidráulicamente

Parámetros	Datos Literatura	Datos AL-PTA			
		Análisis Convencional	Error Relativo Porcentual	Análisis Moderno	Error Relativo Porcentual
k(md)	12.20046	12.4	2.50%	12.383	1.50%
S	-4.65051	-4.9	3.13%	-4.943	6.29%
xf	104.5777	NA	NA	163.8118	56.64%

Figura 20: Grafica Log-Log Prueba Pozo Fracturado Hidraulicamente



3.2. COMPARACION CON PRUEBAS SINTETICAS DE PRESION

3.2.1. CASO 3: Yacimiento Circular Cerrado Homogéneo

Se desarrolló un modelo sintético de un yacimiento circular uniforme cerrado con límites de no flujo con las siguientes características:

Tabla 5: Parámetros de la Prueba, Yacimiento y Pozo-Caso 3

Datos de la Prueba:		
h	60	Ft
B	1.229	rb/STB
r_w	0.27	Ft
ϕ	18	%
μ	1.2	cP
C_t	0.0002640	Psi^{-1}
k	50	md
S	2	

Figura 21: Modelo Siintetico Yacimiento Homogeneo

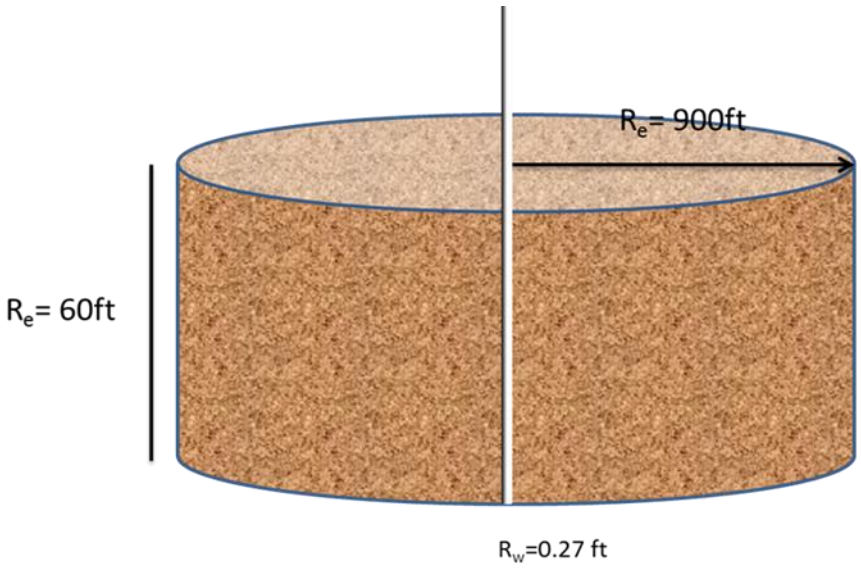
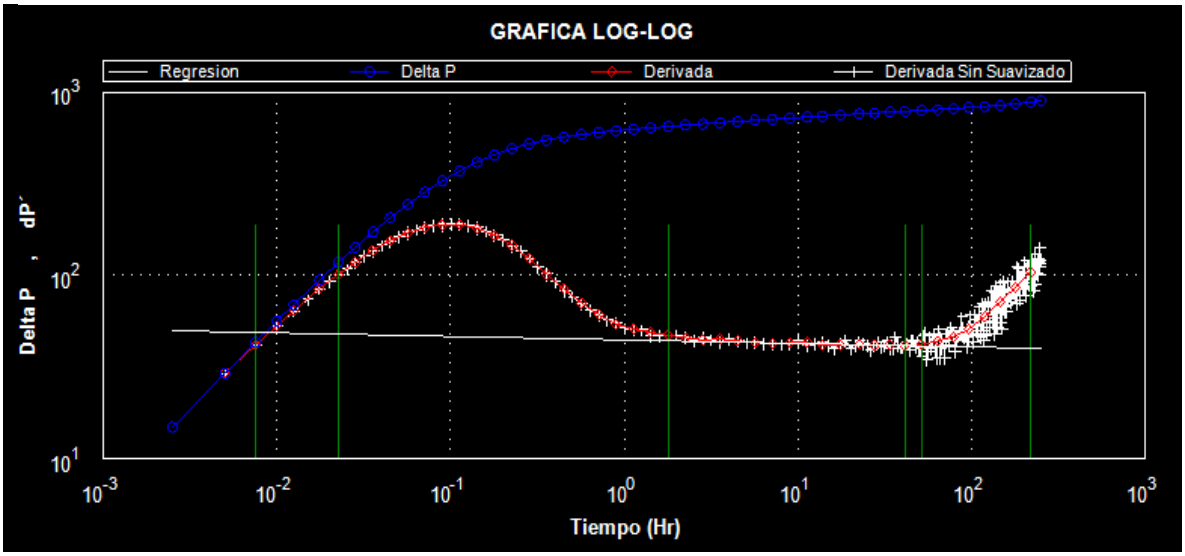


Tabla 6: Resultados Validación Modelo Homogéneo

Parámetro	ALPTA	Modelo	Error Relativo
C (bbl/Psi)	0,0105	0.01	5%
k (md)	48,076	50	3.8%
kh (md-ft)	2.884	3000	3.9%
S	6,989	2	71%

Figura 22: Grafica Log-Log Yacimiento Homogeneo Cerrado



Los resultados del modelo se ajustan a los resultados esperados, debido a que se usó un modelo homogéneo, lo que corresponde a los modelos y suposiciones de la herramienta; además de que el algoritmo de suavizado se comporta bien.

3.2.2. CASO 3: Yacimiento con Fallas Conductivas

En esa prueba se observan un régimen de flujo radial, seguido por el efecto de la falla por una curva cóncava, seguido por flujo bilineal el cual muestra una pendiente característica de 1/4 y finalmente lineal flujo lineal caracterizado por una pendiente de 1/2, la gráfica de la derivada tiene un match muy bueno exhibiendo un comportamiento idéntico al generado por Saphir.

Tabla 7: Parámetros de la Prueba, Yacimiento Y Pozo Usados en la Prueba Sintética

Figura 23: Comparación de la derivada generada por la herramienta contra Saphir

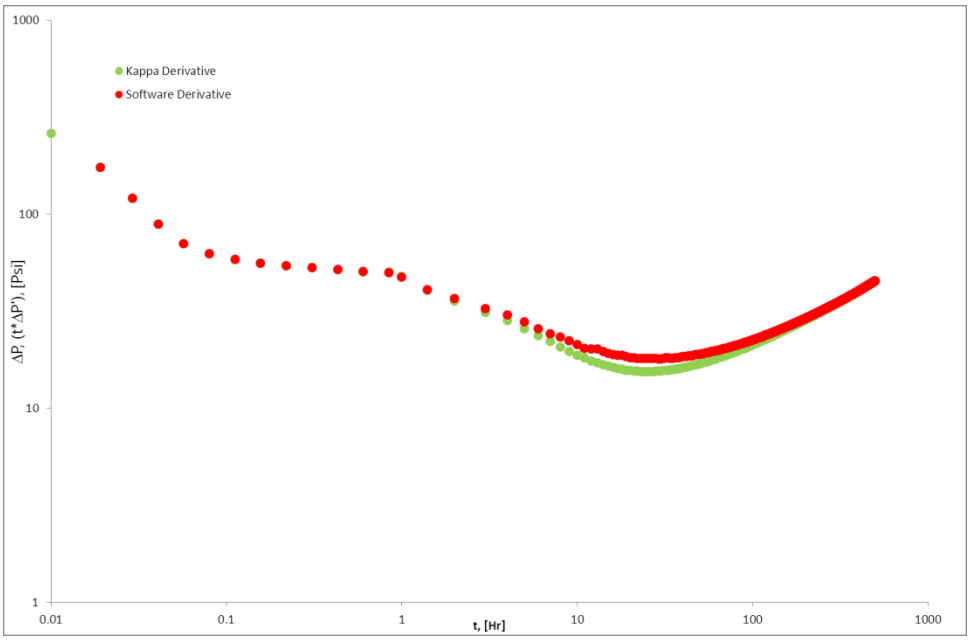


Tabla 7: Parámetros de la Prueba, Yacimiento Y Pozo Usados en la Prueba Sintética

q	500	BOPD
b	2.112	bbl/SBT
u	0.147	cp
h	130	ft
Φ	0.032	
Ct	3.175 x10-5	Psi -1
rw	0.208	ft

Tabla 8: Comparación de Resultados Caso 3

	AL	Saphir Kappa	Error %
C	N/A	N/A	N/A
K, md	1.613	6.5 md	75%
S	2.498	2	24.9 %

Se observa que la herramienta tuvo un match muy bueno en comparación con Saphir coinciden tanto en la región de tiempos medio y tiempos tardíos, hay un ligero desajuste cuando encuentran la presencia de la falla conductiva, no se observa efecto de almacenamiento, aunque los valores de las derivadas en tiempo radiales son idénticos los cálculos de permeabilidad y daño son bastante alejados, esto puede deberse al modelo de fallas y las ecuaciones internas que usa el

simulador para calcular la permeabilidad y daño, ya que por ser iguales el valor de la derivada al usar las ecuaciones de TIAB debería arrojar valores similares.

3.3. COMPARACION CON PRUEBAS REALES

A continuación se muestran los resultados y una breve descripción cualitativa y comparativa sobre las derivadas, debido a que se carece de parámetros de pozo y yacimiento debido a su confidencialidad.

3.3.1. CASO 4:

El pozo “Well1” muestra un buen comportamiento en la derivada, pero cuando este alcanza la región de tiempos medios ocurre una distorsión en la derivada. Aplicando el algoritmo de suavizado se mejora el comportamiento de la derivada y se puede observar el flujo radial.

Figura 24:Grafica Log-Log Caso 3

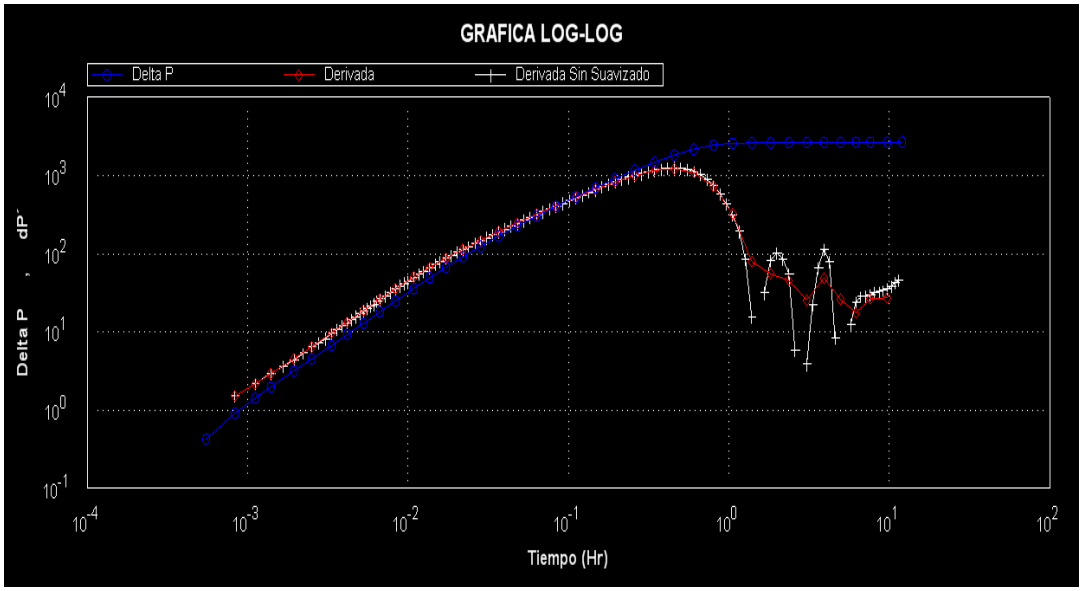
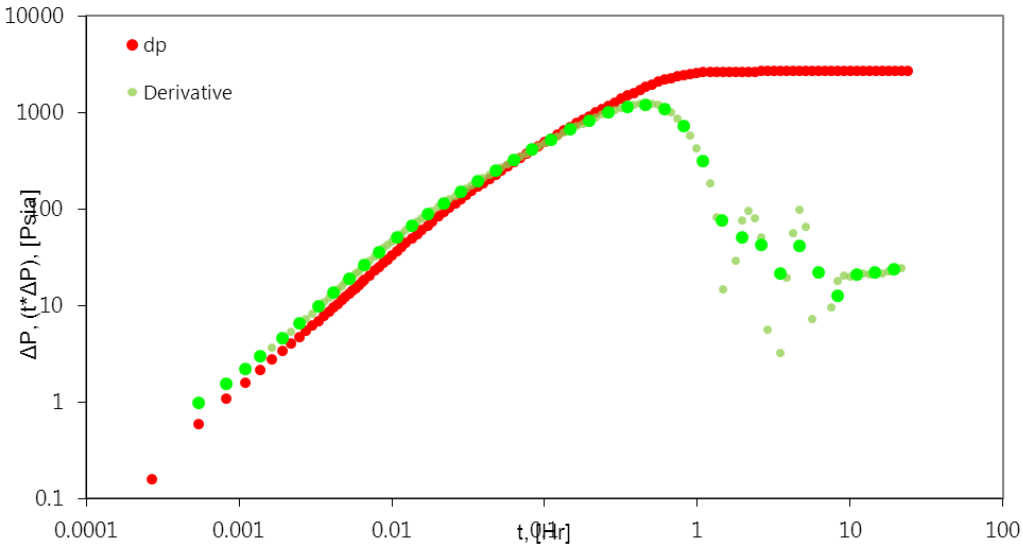


Figura 25: Comparación de la derivada generada por la herramienta contra la realizada en un hoja de Cálculo



Observamos que de nuevo la herramienta ofrece una mejora frente al cálculo tradicional de la derivada, la cual permitió disminuir el ruido en el cálculo de la derivada.

3.3.2. CASO 5:

El Pozo “Well2” de Aceite Volátil fue sometido a una Prueba de Restauración de Presión (PBU), el pozo se produce de una única formación, este se encontraba cerrado por varios meses lo cual indica que la presión del yacimiento se había estabilizado, luego fue puesto en flujo a una tasa constante durante 14.76 horas, después del flujo se procedió a hacer el cierre, la prueba tuvo una duración aproximada de 21 horas, durante este tiempo los censores de presión registraron más de 80000 puntos de presión tomados a intervalos regulares de tiempo, la figura 13 muestra los resultados de la prueba:

Figura 26: Grafica Log-Log Caso 4

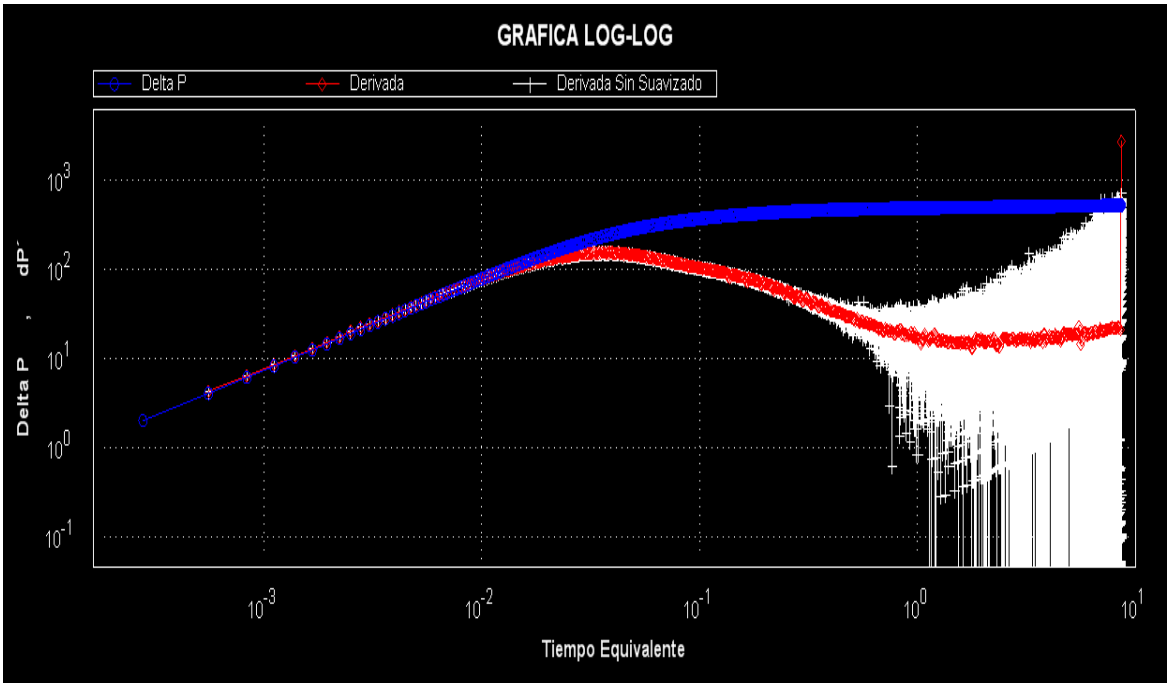


Figura 27: Comparación de la derivada generada por la herramienta contra Saphir

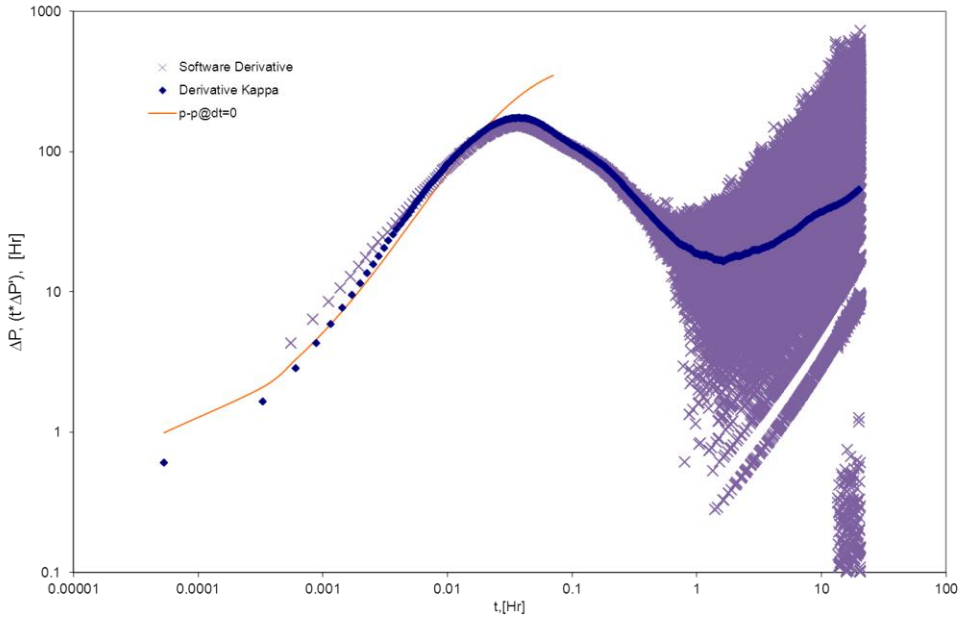
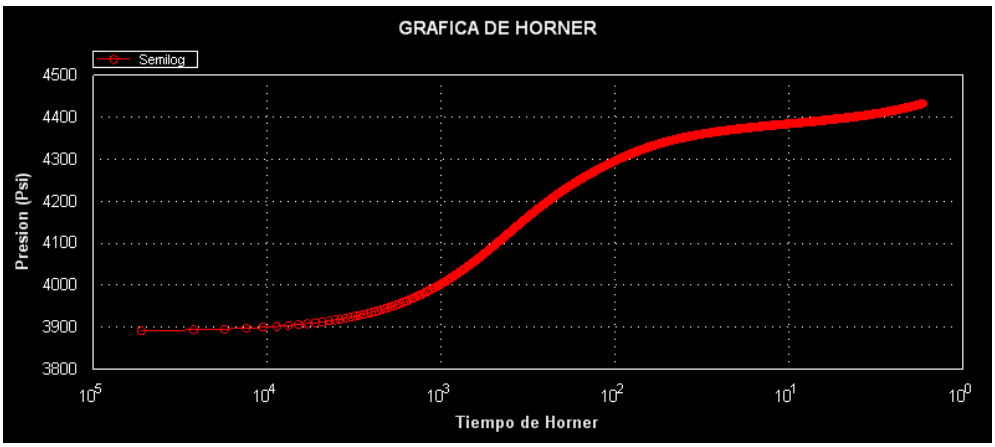


Figura 28: Grafica Semilog Caso 4



Las excelentes condiciones Operativas y de Yacimiento permitieron generar un excelente grafica que muestra un comportamiento de flujo radial y hacia el final de tiempos tardíos indica la presencia de límites que va de acuerdo con la interpretación Geológica.

Al Comparar la gráfica de la derivada se observa un buen match en los tiempos tempranos luego se observa una gran distorsión en el cálculo de la derivada, a pesar de ello las dos gráficas siguen la misma tendencia y parecen indicar la presencia de límites, haciendo un análisis más detallado excesivo ruido de la derivada puede deberse a dos razones:

1. Ya que se trata de un aceite volátil tiene cierta cantidad de gas, lo cual genera flujo multifásico y crea inestabilidad en el cálculo de la derivada.
2. Al indagar en el cálculo de la derivada la aplicación usa un algoritmo de suavizado constante en comparación a Saphir que usa un algoritmo

inteligente para el suavizado que emplea intervalos de tiempo variables en cada región.

4. CONCLUSIONES

- ✓ Se logró diseñar una herramienta de software con una interfaz amigable para el usuario final, que implico una tarea compleja y demandante de tiempo.
- ✓ La metodología de programación orientada a objetos y el diagrama UML de la herramienta permitirá su evolución, mantenimiento y la implementación de nuevos módulos.
- ✓ Se desarrolló exitosamente la construcción de los dos módulos principales para pruebas de caída de presión y restauración de presión, para tasa constante y múltiples tasas.
- ✓ Se creó un módulo de análisis especiales que permite
- ✓ Caracterizar yacimientos naturalmente fracturados, Pozos Fracturados Hidráulicamente y Completado Parcialmente.
- ✓ Se diseñó un manual técnico y un manual de usuario, acompañado de video tutoriales de cada módulo con ejemplos de la literatura.
- ✓ En el caso 1 (yacimiento circular cerrado) la comparación con Saphir arrojo valores satisfactorios con un error relativo menor al 5 % que se puede explicar debido a las condiciones ideales del modelo.
- ✓ Aunque no se planeó hacer validación con modelos de fallas conductivas caso 2, al realizar la prueba, la derivada exhibió un comportamiento similar al mostrado por Saphir, pero los parámetros de permeabilidad y daño mostraron errores relativos superiores al 25%, que pueden explicarse por la diferencia en los modelos.

\

- ✓ Los resultados de daño arrojado por la herramienta presentaron diferencias mayores al 25% con respecto a Saphir, que podrían ser causadas por la diferencias en los algoritmos de suavizados y por modelos de cálculo de daño.

- ✓ El algoritmo de suavizado ha demostrado su efectividad en la mejora de pruebas distorsionadas ocasionadas por el ruido que genera el cálculo de la derivada, como se demostró el caso 4 de la sección de comparación.
- ✓ La herramienta demostró robustez y estabilidad al cargar un conjunto de datos masivos.

5. RECOMENDACIONES Y DESARROLLO FUTURO

Las Pruebas Transientes de Presiones son un tema extenso en la literatura de la ingeniería de Petróleos y aún queda margen para futuros desarrollos, que se ha traducido en una tarea colosal en este trabajo, para trabajos posteriores sobre la herramienta se recomienda extenderla a otros análisis y optimizarla mediante las siguientes implementaciones:

- ✓ Implementar un algoritmo de regresión no lineal, también conocido como *Ajuste Automático de Curvas Tipo,* con el objetivo de encontrar un único resultado que se ajuste mejor al modelo.
- ✓ Realizar la inclusión de modelos numéricos para diversas formas y configuraciones de pozo y yacimiento, con el fin de contar con una base de datos de modelos, a través de los cuales implementar algoritmos de ajuste automático de curvas tipo.
- ✓ Mejorar el algoritmo de suavizado utilizado en la herramienta, con el fin hacerlo más eficiente y permitir un suavizado dinámico.
- ✓ Implementar algoritmos de Deconvolución, ya que esta técnica muy reciente permite transformar pruebas MultiTasa a tasa constante, eliminando la necesidad de usar el principio de superposición y permite ver más detalles de la prueba especialmente en tiempos tardíos.
- ✓ Utilizar una Nueva Librería Grafica más potente y versátil, que permita una mejor experiencia de usuario y una mayor facilidad de implementación y uso.
- ✓ Diseñar un módulo para pruebas de presión en pozos de gas.
- ✓ Utilizar métodos que permitan el análisis de pruebas de presión para flujo multifásico, como primera aproximación utilizar la aproximación de Perrine.

- ✓ Extender los módulos para incluir el análisis para pruebas de presión para yacimientos multicapa.
- ✓ Utilizar métodos que permitan el análisis para pruebas de presión teniendo en cuenta el flujo cruzado entre capas.

BIBLIOGRÁFIA

1. Abel Chacon, Abdelghani Djerouni, Djebbar Tiab.: “Determining the Average Reservoir Pressure from Vertical and Horizontal Analysis Using Tiab’s Direct Synthesis Technique”, SPE paper 88619, 2004.

2. A.C. Gringarten.: “ From Straight Lines to Deconvolution: The Evolution of the State of the Art in Well Test Analysis”, SPE paper 102079, 2006

3. Amanat U. Chaudhry.: “ Oil Well Testing Handbook”, 1ra edición , 2004

4. BOURDET, Dominique: “Well Test Analysis: The Use of Advance Interpretation Models”, 1ra edición, 2002

5. D. Bourdet.: “ A new Set of Type Curves simplifies well test analysis”, SPE paper p324, 1985

6. D. Tiab, I.N. Ispas, A. mongi.: “ Interpretation of Multirate Test by the Pressure Derivative”, SPE paper 53935, 1999.

7. Eni S.p.A.: “Well Testing Manual”, Manual, 2004.

8. ESCOBAR, Freddy H. : “Análisis Moderno de Presiones de Pozos”, 1ra edición, 2004.

9. EZEKWE, Nnaemeka.: “Petroleum Reservoir Engineering Practice.”, 1ra edición, 2011

10. G. Pedaci.: “ Advanced Well Testing handbook”,

11. Gilles Bourdarot.: “Well Testing: Interpretation Methods”. 1ra edicion, 1996.

12. Horne, R. N.: “Modern Well Test Analysis: A Computer-Aided Approach”, 2da edición, 1995

13. LEE, John. ROLLINS, John B., SPIVEY, John P.: “Well Test Analysis:

14. M. Onyekonwu.: “Application of total concept to well test in high permeability reservoirs”. SPE Distinguished Lecturer Series.

15. Ortiz Olga.: “ Apuntes Curso Análisis de Pruebas Transientes de Presión”, 2011

16. Pressure Transient Testing”, SPE Texbook Series vol. 9, 2003.

17. Schlumberger: “Introduction to Well Testing”, 2006.

18. Schlumberger. “Fundamentals of Formation Testing.”, 2006.

19. Schlumberger.: “: Well test 200 Technical Description”, Manual, 2001

20. Schlumberger. “Well Test Interpretation”, 2002.

21. Stuart McAleese.: “Operational Aspects of Oil and Gas Well Testing”, 1ra edición, 2000.

22. S. Bachar, A.H. Deghmoun, DTiab.: “A Method for Reducing Model Error When Interpreting Pressure Derivative in Well Test Analysis application to Hassi Messaoud Oil Field”, PETSOC paper 2003-078, 2003.

23. Tarek Ahmed, McKinney Paul D.: “Advanced Reservoir Engineering”, 1ea edition, 2005.

24. Software, Wikipedia la Encyclopedia Libre, http://es.wikipedia.org/wiki/Software

25. Ingeniería de software, Wikipedia la Encyclopedia Libre http://es.wikipedia.org/wiki/Ingenier%C3%ADa_de_software

26. Diagrama de clases, Wikipedia la Encyclopedia Libre http://es.wikipedia.org/wiki/Diagrama_de_clases

ANEXOS

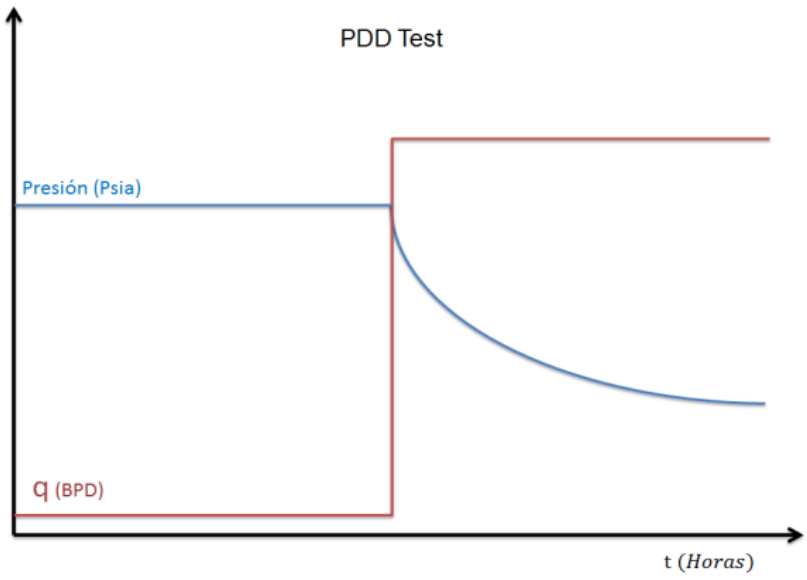
ANEXO A: MANUAL DE REFERENCIA PARA EL SOFTWARE DE ANALISIS DE PRESIONES

Este manual está diseñado para conocer las ecuaciones internas y los métodos en las diferentes pruebas de análisis de presiones de los que hace uso el Software de Análisis de Presiones desarrollado en el proyecto de grado titulado *Herramienta de Software Para El Análisis de Pruebas de Presión.* Si desea conocer el funcionamiento de la herramienta a través de ejemplos didácticos remítase al **MANUAL DE USUARIO PARA EL SOFTWARE DE ANALISIS DE PRESIONES**.

1. PDD

Una prueba de Caída de Presión (PDD, Pressure Drawdown), es aquella que se realiza en un pozo productor, que comienza idealmente con una presión uniforme o estabilizada con una tasa constante (generalmente cero), posteriormente la tasa cambia (manteniéndose constante nuevamente) y registrándose la presión de fondo fluyendo y el tiempo.

Figura 29: Representación esquemática prueba PDD



Fuente: Modificado de Análisis Moderno de Presiones. Escobar F. 2003

Un procedimiento para realizar una prueba de caída de presión sería[4]:

- Cerrar el pozo por un periodo suficiente hasta alcanzar la estabilización en todo el yacimiento (si no se alcanza la estabilización se requerirá una prueba MultiTasa).
- Se baja la herramienta a un nivel exactamente sobre las perforaciones (La herramienta debe tener mínimo 2 sensores para tener un buen control de calidad).
- Abrir el pozo para producir a tasa constante y registrar la Presión de Fondo Fluyendo (P_{wf}) y el tiempo (t).

Esta prueba puede ser denominada como larga o corta, dependiendo si su duración alcanza varias horas o días. Las pruebas cortas se realizan usualmente para obtener parámetros del yacimiento dentro del área de drenaje del mismo, tales como **Permeabilidad (*K*), Daño (*S*), Radio Efectivo de Drenaje (r_e) y Coeficiente de Almacenamiento (*C*).** Las pruebas largas se usan especialmente para obtener valores relacionados con el **Tamaño del Yacimiento, Volumen Poroso Total, Distancia a Fallas, etc.**

En una prueba de caída de presión de larga duración se pueden observar tres regiones características en la gráfica de Pwf Vs.t (Ver Figura 16) La primera región,

1.1. REGIONES DE FLUJO

1.1.1. ETR (Early Time Region)

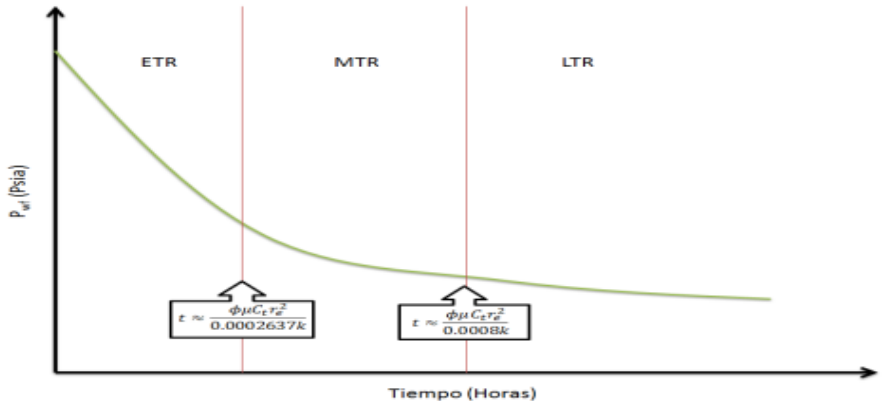
Está influenciada por los efectos en la cara del pozo y el almacenamiento. El software permite estimar este parámetro mediante el análisis de líneas rectas empleando la siguiente ecuación.

[4] ESCOBAR, Fredy Humberto. Análisis Moderno de Presiones de Pozo, Neiva, Huila, 2003, p 78.

$$C = \left(\frac{qB}{24}\right)\left(\frac{t}{\Delta p}\right)_{lpu}$$

Ecuación 10

Figura 30: Regiones de Flujo Prueba PDD



Fuente: Modificado de Petroleum Reservoir Engineering Practice. Ezekwe N. 2011

1.1.2. MTR (Middle Time Region)

Esta región se ve dominada por las respuestas del yacimiento más allá del radio del pozo pero antes de tocar los límites del yacimiento, aquí se observan los efectos del flujo radial infinito, donde es posible conocer la permeabilidad y el Skin, la herramienta emplea las siguientes Ecuaciones:

$$k = \frac{162.6qB\mu}{mh}$$

Ecuación 11

$$S = 1.151\left(\frac{P_I - P_{wf1hr}}{m} - \log\left(\frac{kt}{\varphi\mu C_t {r_w}^2}\right) + 3.23\right)$$

Ecuación 12

La aplicación también permite calcular la caída de presión debido al skin

$$\Delta P_s = \frac{141.2qB\mu}{kh}S = 0.869mS$$

Ecuación 13

1.1.3. LTR (Late Time Region)

En la tercera región, se ven reflejados los efectos de los límites del yacimiento. Allí pueden estimar el área de drenaje, volumen poroso y el factor de forma

$$V_p = \varphi Ah = \frac{0.23395qB}{C_t m*}$$

Ecuación 14

$$C_A = 5.493\left(\frac{m}{m^*}\right)e^{\left(-\frac{2.303(P_{1hr}-P_{int})}{m}\right)}$$

Ecuación 15

1.2. PDD TASA CONSTANTE

Una Prueba de Caída de Presión (PDD) para una tasa consiste en mantener el pozo cerrado hasta su estabilización, posteriormente se abre el pozo y se comienza a registrar la P_{wf} Vs t.

1.2.1. Prueba Corta

Una prueba de caída de presión (PDD) corta, es aquella en la cual, debido al tiempo de la misma, no se alcanzan a evidenciar los efectos de los límites del yacimiento.

Para el análisis de estas pruebas de caídas de presión cortas, existen métodos tales como el **Ajuste Por Curvas Tipo**, el método **Earlougher**, y el **Análisis Semilog**. Para efectos del desarrollo de este proyecto, se abordará solamente el **Análisis Semilog**.

Este análisis se usa tanto para pruebas cortas como para pruebas largas. Se asume que hay un solo pozo en un yacimiento.

1.2.2. Prueba LTR (Late Transient Reservoir)

Esta prueba de caída de presión se usa para establecer información sobre las fronteras del yacimiento. La diferencia con la prueba corta, consiste en que la LTR dura más tiempo registrando presiones y tiempos, hasta tocar todos los límites del yacimiento o hasta presentarse el comportamiento de flujo Pseudo-estable.

El análisis de la región MTR se realiza con las gráficas Semilog y mediante el análisis de líneas rectas y las ecuaciones 11 y 12 descritas en la sección 2.1.2.

1.2.3. Síntesis Directa de Tiab (TDS)

Este enfoque de análisis de pruebas de presión se basa en el análisis de una derivada de la presión. Esta derivada de la presión se grafica en la interfaz de la herramienta a través de una gráfica log-log, y su análisis consiste en la localización de ciertos puntos y pendientes características (Para mayor información consulte el apéndice para una explicación detallada del cálculo de la derivada y el funcionamiento del algoritmo de suavizado). Con respecto al cálculo

de esta derivada, existen varios métodos propuestos por varios autores, con el fin de reducir el ruido generado por las operaciones de cálculo, pero el método más utilizado es el propuesto por Horne, el cual se comporta mejor para puntos distribuidos geométricamente, y es planteado de la siguiente manera:

$$t\left(\frac{\partial P}{\partial t}\right)_i = t\left(\frac{\partial P}{\partial \ln t}\right)_i$$

$$= \left\{ \frac{\ln\left(\frac{t_i}{t_{i-1}}\right)\Delta P_{i+1}}{\ln\left(\frac{t_{i+1}}{t_i}\right)\ln\left(\frac{t_{i+1}}{t_{i-1}}\right)} + \frac{\ln\left(\frac{t_{i+1} * t_{i-1}}{t_i^2}\right)\Delta P_i}{\ln\left(\frac{t_{i+1}}{t_i}\right)\ln\left(\frac{t_{i1}}{t_{i-1}}\right)} \right.$$

$$\left. - \frac{\ln\left(\frac{t_{i+1}}{t_i}\right)\Delta P_{i-1}}{\ln\left(\frac{t_i}{t_{i-1}}\right)\ln\left(\frac{t_{i+1}}{t_{i-1}}\right)} \right\}$$

Ecuación 16

Las características más importantes hacen referencia a una pendiente unitaria al inicio de la prueba que reflejan los efectos del almacenamiento, con la cual se puede calcular el coeficiente de almacenamiento.

$$C = \left(\frac{qB}{24}\right) \frac{t}{(t * \Delta P')_{lpu}}$$

Ecuación 17

La “joroba” observada en la curva representa los efectos del daño en la cara del pozo y su disminución en la capacidad de flujo generando una caída de presión. Otro aspecto notable es la línea horizontal que muestra la aparición del flujo radial infinito con el cual se puede obtener un valor de permeabilidad. Al leer un tiempo en la línea horizontal, junto con su respectivo delta de presión y derivada, más la permeabilidad otros parámetros del yacimiento y el fluido, es posible hacer una estimación satisfactoria del valor del daño.

$$S = 0.5\left(\frac{\Delta P_r}{(t * \Delta P')_r} - \ln\left(\frac{kt_r}{\varphi\mu C_t {r_w}^2}\right) + 7.43\right)$$

Ecuación 18

$$d_c = 0.0122\sqrt{\frac{kt_{inf}}{\varphi\mu C_t}}$$

Ecuación 19

Distancia a la falla más alejada:

$$d_F = \sqrt{\left(\frac{0.0002637kt_{2HL}}{\varphi\mu C_t}\right)\left(\frac{\theta^{1.62}}{145494}\right)}$$

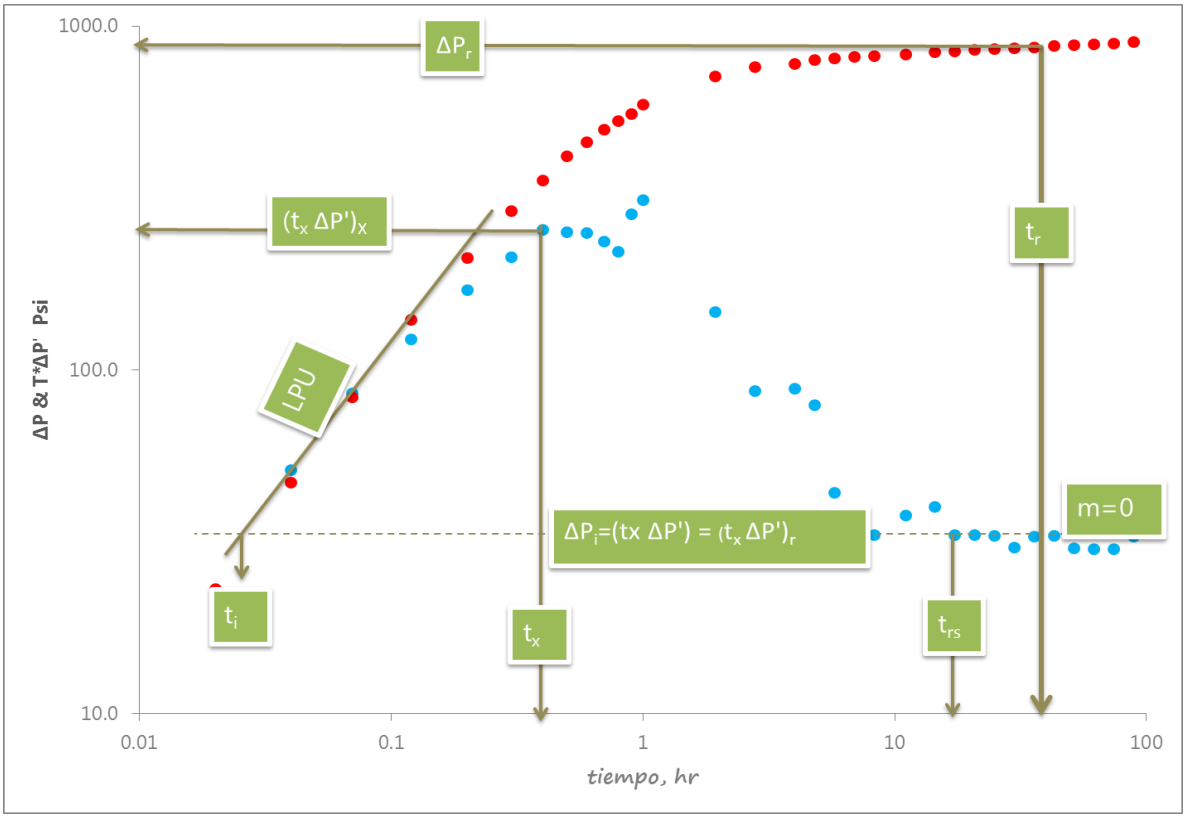
Ecuación 20

$$\theta = \left(\frac{141.2q\beta\mu}{kh}\right)\frac{180}{(t * \Delta P')_{2HL}}$$

Ecuación 21

En la región de tiempos tardíos (LTR) se puede encontrar barreras de no flujo, por medio de esta ecuación incluida en la herramienta, se calcula la distancia y el ángulo entre ellos

Figura 31: Puntos Característicos de la gráfica de la derivada.



Fuente: Grafica modificada Ortiz Olga, Apuntes Curso Análisis de Presiones

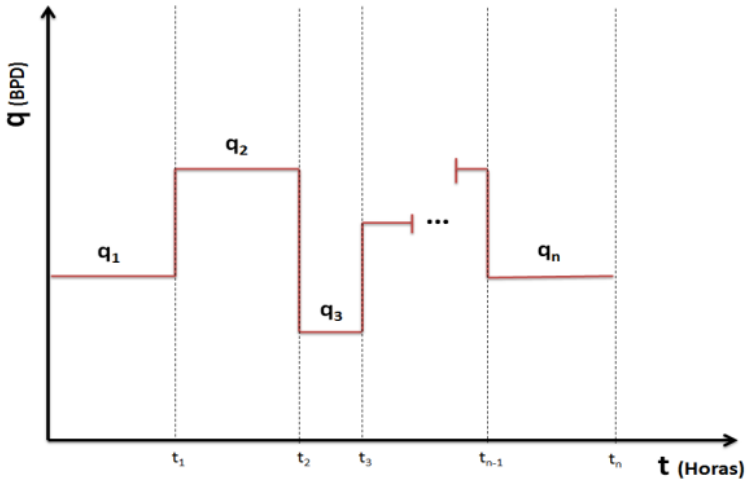
1.3. PDD TASA VARIABLE

Operacionalmente, la prueba de caída de presión a tasa constante ocasiona varios inconvenientes para su correcta realización: Mantener un pozo cerrado por varias horas, incluso días, es difícil de lograr; además de que conseguir que se estabilice la presión de un yacimiento no es tarea fácil.

Por este y otros motivos surge la utilización de pruebas de caída de presión para múltiples tasas (Ver Figura 18). Estas pruebas consisten en mantener una tasa constante de preparación, posteriormente variar la producción a otra tasa,

registrando en todo momento los valores de presión y tiempo, de esta manera para tantas tasas distintas como fuera diseñada la prueba.

Figura 32: Representación esquemática prueba PDD MultiTasa



Fuente:. Modificado de Petroleum Reservoir Engineering Practice. Ezekwe N.

1.3.1. Análisis Cartesiano

Para realizar el análisis de una prueba MultiTasa, es necesario hacer uso del principio de superposición en el tiempo anteriormente descrito. Después de hacer unas manipulaciones matemáticas se llega a:

$$P_i - P_{pwf} = m_m q_1[\log(t) + b_m] + m_m(q_2 - q_1)[\log(t - t_1) + b_m]$$

$$+m_m(q_3 - q_2)[\log(t - t_2) + b_m] +$$

$$\ldots + m_m(q_n - q_{n-1})[\log(t - t_{n-1}) + b_m]$$

Tomando en cuenta que:

$$m = \frac{162.6B\mu}{kh}$$

$$b_m = \log\left(\frac{k}{\phi\mu c_t r_w^2}\right) - 3.23 + 0.8691s$$

$$P_i - P_{wf} = m_m q[\log(t) + b_m]$$

Lo cual se puede condensar y reorganizar de la siguiente manera:

$$\frac{P_i - P_{pwf}}{q_n} = m_m \sum_{i=1}^{n}\left[\frac{(q_j - q_{n-1})}{q_n} log(t - t_{j-1})\right] + b_m\left[log\left(\frac{k}{\phi\mu c_t r_w^2}\right) - 3.23 + 0.8691s\right]$$

Ecuación 22

1.3.2. Análisis Semilog

La ecuación que gobierna este método es

$$\frac{P_i - P_{wf}(t_n)}{q_n} = m' * \log(t_{eq}) + b'$$

Donde, t_{eq} puede ser calculado mediante la siguiente ecuación:

$$t_{eq} = \prod_{i=1}^{n}(t_n - t_{n-1})^{\left(\frac{q_i - q_{i-1}}{q_n}\right)} = 10^{x_n}$$

Ecuación 23

Por simplicidad para computar el tiempo equivalente el software calcula la función sumatoria y luego eleva a la 10 el resultado de la función sumatoria, evitando así calcular la productoria que es un procedimiento más complejo.

1.3.3. Síntesis Directa de Tiab Para PDD MultiTasa

Recientemente se han desarrollado metodologías basadas en la Síntesis de Tiab, las cuales aplican el principio de superposición junto con técnicas numéricas y suposiciones del análisis Semilog; con lo cual es posible calcular coeficientes de almacenamiento, permeabilidad y daño, el Software incorpora el análisis para calcular esta derivada pero los resultados no son lo suficientemente robustos comparación con las pruebas de tasa constante, se debe tener cuidado a la hora de realizar el análisis de la prueba

Para este caso el software calcula la derivada con la derivada cartesiana realizando una gráfica log-log de:

$$(t_{eq} * \Delta P_q{}') \; vs \; t_{eq}$$

$$(t_{eq} * \Delta P_q{}') = te_i \left\{ \frac{(te_i - te_{i-1})\Delta P_q{}'_{i+1}}{(te_{i+1} - te_i)(te_{i+1} - te_{i-1})} + \frac{(te_{i+1} - 2te_i + te_{i-1})\Delta P_q{}'_i}{(te_{i+1} - te_i)(te_i - te_{i-1})} - \frac{(te_{i+1} - te_i)\Delta P_q{}'_{i-1}}{(te_i - te_{i-1})(te_{i+1} - te_{i-1})} \right\}$$

Ecuación 24

Para realizar el cálculo de almacenamiento, permeabilidad y daño la herramienta utiliza las Ecuaciones 10, 11 y 12 .

2. PBU

Muchos ingenieros han comenzado a preferir el uso de las pruebas de restauración de presión (PBU, Pressure Buildup),debido a que la única tasa estable que se puede alcanzar en la realidad es cero, que equivale a cerrar el pozo. En una típica prueba PBU, un pozo productor es cerrado y su presión de fondo fluyendo es medida como una función del tiempo.

Una prueba de restauración de presión convencionalmente se corre de la siguiente manera:

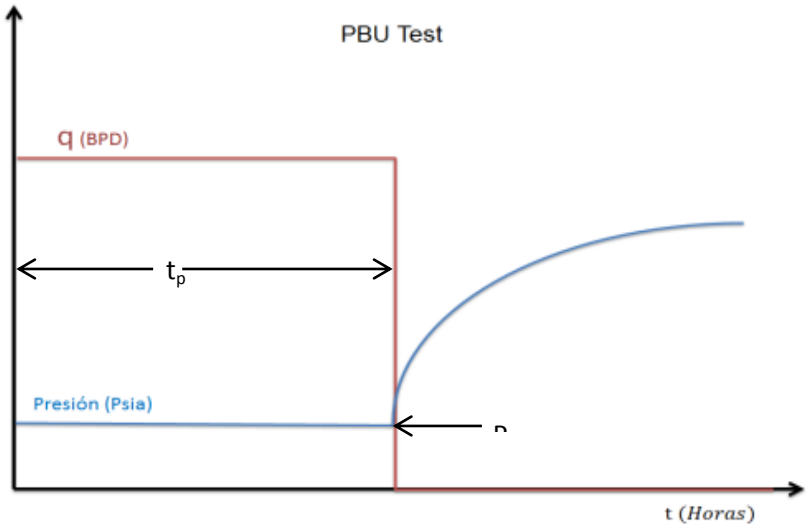
1. Determinar la ubicación de los empaques, tamaño de la tubería de producción y la tubería de revestimiento, profundidad del pozo
2. Estabilizar el pozo a una tasa constante q.
3. Cerrar el pozo y registrar el valor de P_{wf} justo antes del cierre.
4. Leer la presión de cierre P_{ws}, a intervalos cortos de 15 segundos para los primeros minutos (10-15 minutos), luego cada 10 minutos durante la primera hora; posteriormente, para las siguientes 10 horas tomar la presión cada 1 hora y finalmente los intervalos de toma de datos se pueden extender hasta 5 horas.

   Una prueba PBU ideal asume que la tasa de producción antes de que el pozo fuera cerrado era constante. Esto raramente es posible, ya que es extremadamente difícil mantener constante la tasa de producción de un pozo para un periodo de tiempo largo.

2.1. PBU TASA CONSTANTE

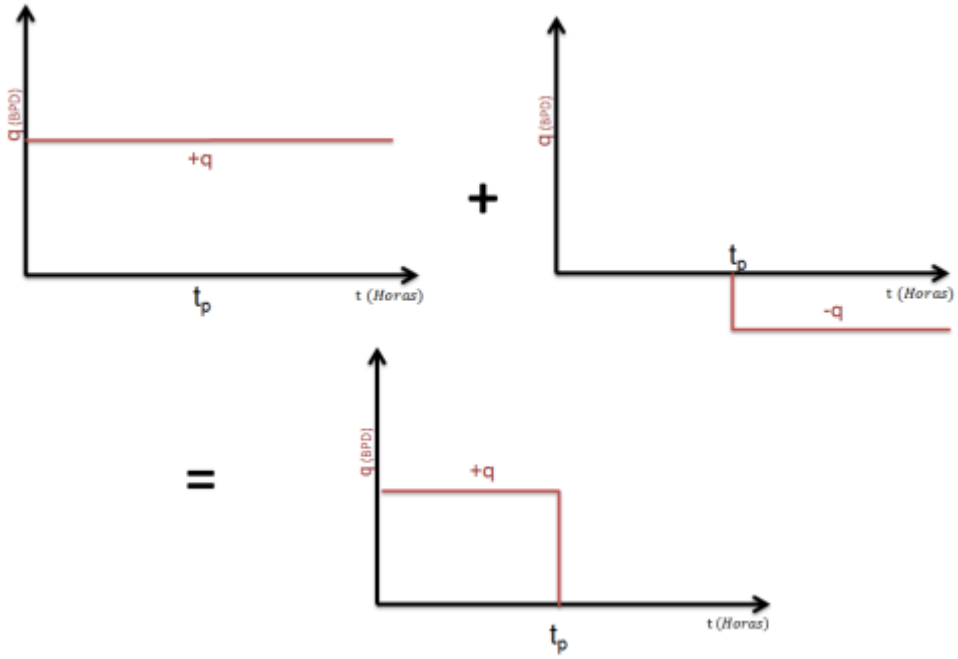
Las ecuaciones para el análisis de las pruebas de restauración de presión son derivadas basadas en la suposición de que la tasa de producción antes del cierre es constante. Posteriormente se realizaran modificaciones y se presentará ecuaciones para el análisis de pruebas PBU con tasa variable antes del cierre.

Figura 33: Representación esquemática prueba PBU



Fuente:. Modificado de Análisis Moderno de Presiones. Escobar F. 2003

Figura 34: Representación Esquemática de la Superposición en el Tiempo



Fuente: Modificado de Presentaciones Clases Análisis de Presiones. Ortiz O. 2011

2.1.1. Método de Horner

Se parte de la suposición de que antes del cierre el pozo ha producido a una tasa constante durante un tiempo t_p y se aplica el principio de superposición en el tiempo para ajustar las tasas. La ecuación que modela este comportamiento está dada por:

$$P_{pws} = P_i - 70.6\frac{qB\mu}{kh}\left[ln\left(\frac{t_{p+\Delta t}}{\Delta t}\right)\right]$$

Este método asume que el tiempo total de producción t_p es mucho mayor que el tiempo de cierre Δt y que el periodo de flujo fue llevado a tasa constante y que la presión se estabilizó en el tiempo t_p, lo cual raramente ocurre.

2.1.2. Tiempo Equivalente

Debido a las suposiciones hechas para el método de Horner, se desarrolló una expresión "normalizada" de la ecuación, en la cual se resta la caída de presión antes del cierre de la ecuación de Horner, obteniendo la siguiente expresión:

$$\Delta P = P_{ws} - P_{ws(\Delta t=0)} = m\left[log\left(\frac{t_p\Delta t}{t_p + \Delta t}\right) + log\left(\frac{k}{\phi\mu c_t r_w^2}\right) - 3{,}23 + 0{,}869 * S\right]$$

Y el tiempo equivalente está definido como:

$$\Delta t_e = \frac{t_p}{\left(\frac{t_{p+\Delta t}}{\Delta t}\right)} = \frac{t_p\Delta t}{t_p + \Delta t}$$

Ecuación 25

Por lo tanto la ecuación quedaría de esta manera:

$$\Delta P = P_{ws} - P_{ws(\Delta t=0)} = m\left[log(\Delta t_e) + log\left(\frac{k}{\phi\mu c_t r_w^2}\right) - 3{,}23 + 0{,}869 * S\right]$$

Si el tiempo de cierre tiende a infinito, el tiempo equivalente tiende al tiempo total de producción. El software no incorpora otros métodos como MDH, ya que está comprobado que Horner es más preciso para cualquier tiempo de producción.

2.1.3. ETR

Para calcular el almacenamiento se usa una ecuación similar a la ecuación 2.3 con ligeras modificaciones:

$$C = \frac{qB}{24}\left(\frac{\Delta t_N}{\Delta P_N}\right)_{lpu}$$

Ecuación 26

2.1.4. MTR

Para realizar el cálculo del skin se usa la ecuación 2.2, con ligeras modificaciones:

$$S = 1.151\left(\frac{P_{1hr} - P_{ws(\Delta t=0)}}{m} - \log\left(\frac{kt}{\varphi\mu C_t {r_w}^2}\right) + 3.23\right)$$

Ecuación 27

2.1.5. LTR

La aplicación permite calcular distancia a límites en la gráfica de Horner.

$$d = 0.01217\sqrt{\frac{kt_p}{\varphi\mu C_t}\left(\frac{t_p + \Delta t}{\Delta t}\right)_x}$$

Ecuación 28

El subíndice x corresponde al intercepto del tiempo de Horner entre las dos líneas rectas.

2.2. CALCULO DE LA PRESIÓN PROMEDIO DEL YACIMIENTO

Esta es la presión que alcanzaría el yacimiento si idealmente se cerraran todos los pozos por un tiempo infinito. Esta es útil para caracterizar el yacimiento, obtener volúmenes de aceite in-situ, para predecir comportamientos futuros y es un parámetro de información indispensable para procesos de recobro secundario y terciario. Este análisis ofrece un estimado de la presión promedio de la zona de drenaje.

2.2.1. Método MBH (Matthews-Bronz & Hazebrock)

Para el método convencional del cálculo de la presión promedio la herramienta usa el método MBH. Este método es considerado el más exacto y utiliza un gráfico Horner. Se aplica en la mayoría de situaciones donde se desea hallar la presión promedia en un yacimiento cerrado para cualquier localización de pozo dentro de una variedad de formas de drenaje. El método asume que no hay variaciones en la movilidad o compresibilidad del fluido dentro de la región de drene. Esta limitación se puede sobrellevar usando un tiempo de producción t_p igual t_{pss}. El procedimiento es:

1. Calcule

$$t_p = \frac{24N_p}{q}$$

Ecuación 29

2. El valor de t_p debe ser comparado con el tiempo requerido para alcanzar el estado pseudoestable. Por lo tanto obtenga $(tDA)_{pss}$ de **las Tablas 3 y 4**, de la columna “exacto para tDA > “. Para esto debe conocerse previamente la forma del yacimiento.

3. Calcule el tiempo para alcanzar el estado pseudoestable, t_{pss}:

$$t_{pss} = \frac{\varphi \mu C_t A (t_{DA})_{pss}}{0.0002637k}$$

Ecuación 30

4. Obtenga la relación α, $\alpha = t_p/t_{pss}$. Si $\alpha > 2.5$ entonces, haga $t = t_{pss}$. Si $\alpha < 2.5$ (para ratas muy altas, el mejoramiento en el cálculo de la presión promedia es significativo cuando α está comprendido entre 2.5 y 5) entonces haga $t = t_p$. Luego grafique P_{ws} vs. $(t+\Delta t)/\Delta t$. Como se vio anteriormente, el uso de t_{pss} en el método de Horner puede incrementar la longitud de la recta semilog, contrario al gráfico MDH.

5. Con el tiempo, t, definido en el paso anterior determine t_pDA

$$t_{pDA} = \frac{0.0002637k}{\varphi \mu C_t A} t$$

Ecuación 31

6. Extrapole la recta de la gráfica semilogarítmica del gráfico Horner y halle P*

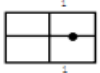
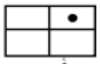
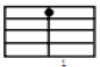
7) Determine PDMBH de la **Figuras. 15 a 18** usando el tpDA calculado en el paso 5.

8) Calcule la presión promedia:

$$\mathbf{P_{prom} = P* - \left(\frac{m}{2.3025}\right) P_{DMBH}}$$

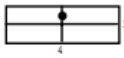
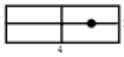
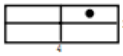
Ecuación 32

Tabla 9: Factores de forma para varias áreas de drenaje de pozos sencillos

Yacimientos finitos	C_A	Exacto Para t_{DA} >	Menos de 1 % error para t_{DA} >	Use solución de sistema infinito con menos de 1 % error for t_{DA} >
	31.62	0.1	0.06	0.1
	31.6	0.1	0.06	0.1
	27.6	0.2	0.07	0.09
	27.1	0.2	0.07	0.09
	21.9	0.4	0.12	0.08
	0.098	0.9	0.6	0.015
	30.8828	0.1	0.05	0.09
	12.9851	0.7	0.25	0.03
	4.5132	0.6	0.30	0.025
	3.3351	0.7	0.25	0.01
	21.8369	0.3	0.15	0.025
	10.8374	0.4	0.15	0.025
	4.5141	1.5	0.50	0.06
	2.0769	1.7	0.5	0.02
	3.1573	0.4	0.15	0.005

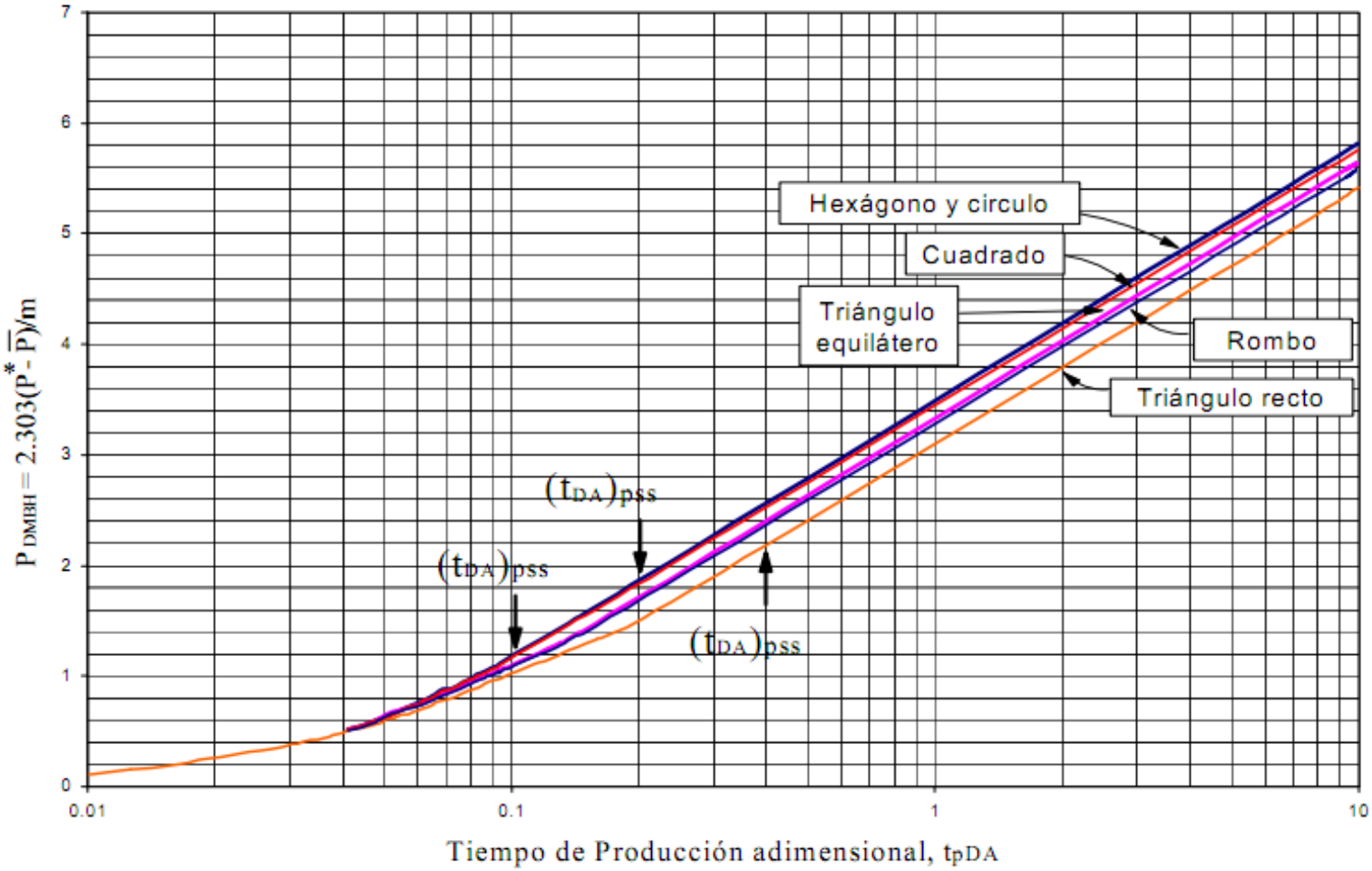
Fuente: Análisis Moderno de Presiones de Pozos, Freddy Humberto Escobar M., PhD.

Tabla 10: Continuación Factores de forma para varias áreas de drenaje de pozos sencillos

	C_A	Exacto para t_{DA} >	Menos de 1 % error para t_{DA} >	Use solución de sistema infinito con menos de 1 % error for tDA >
	0.5813	2.0	0.6	0.02
	0.1109	3.0	0.6	0.005
	5.379	0.8	0.3	0.01
	2.6896	0.8	0.3	0.01
	0.2318	4.0	2.0	0.03
	0.1155	4.0	2.0	0.01
	2.3606	1.0	0.4	0.025
Vertical-Fractured reservoirs		Use $(Xe/Xf)^2$ in place of A/rw^2 for fractured reservoirs		
	2.6541	0.175	0.08	Cannot use
	2.0348	0.175	0.09	Cannot use
	1.9986	0.175	0.09	Cannot use
	1.662	0.175	0.09	Cannot use
	1.3127	0.175	0.09	Cannot use
	0.7887	0.175	0.09	Cannot use
Water-Drive reservoirs 	19.1	--	--	--
Unknown Drive mechanism 	25.0	--	--	--

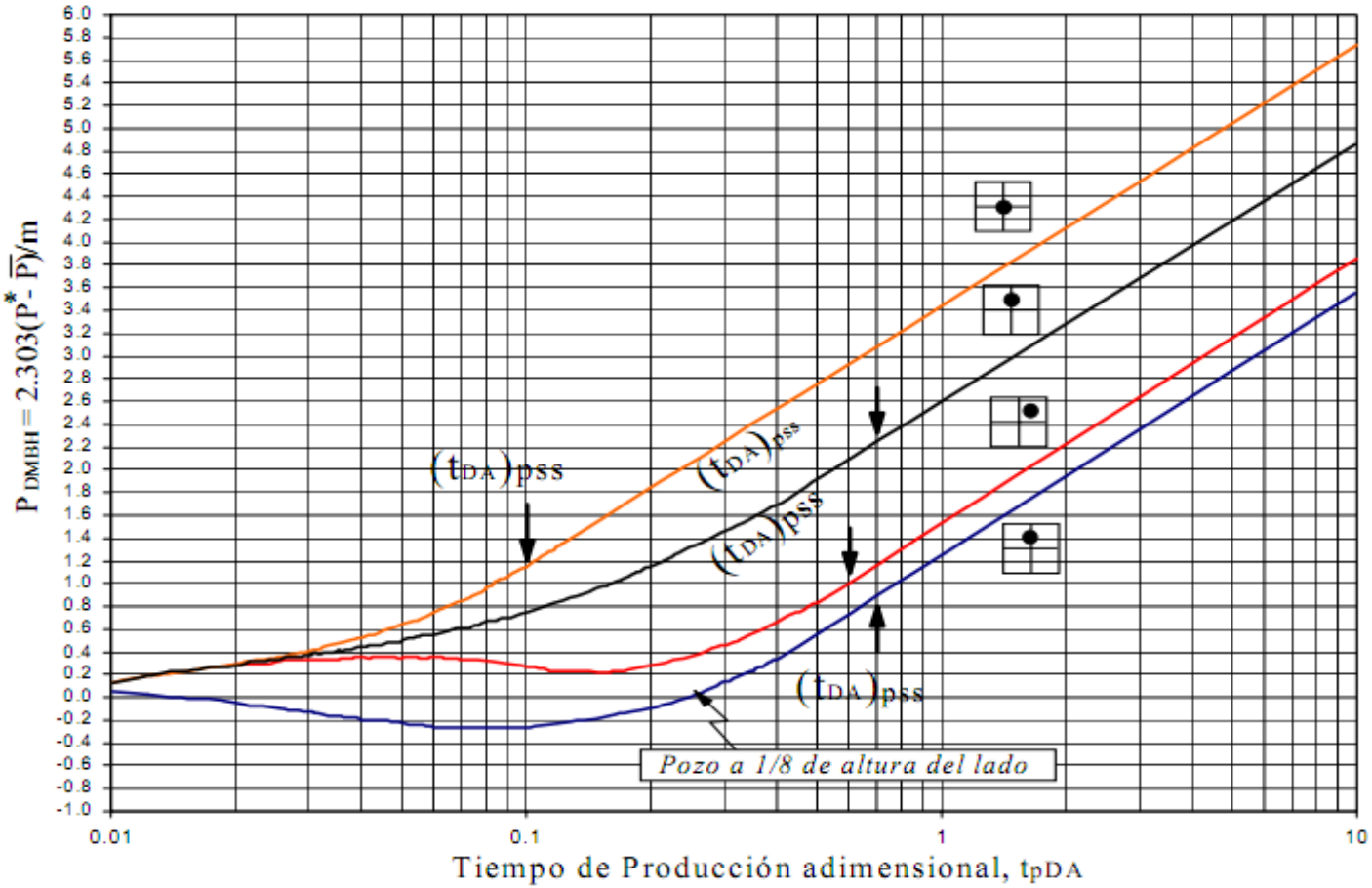
Fuente: Análisis Moderno de Presiones de Pozos, Freddy Humberto Escobar M., PhD

Figura 11: P_{DMBH} para un pozo en el centro de áreas de drene equiláteras



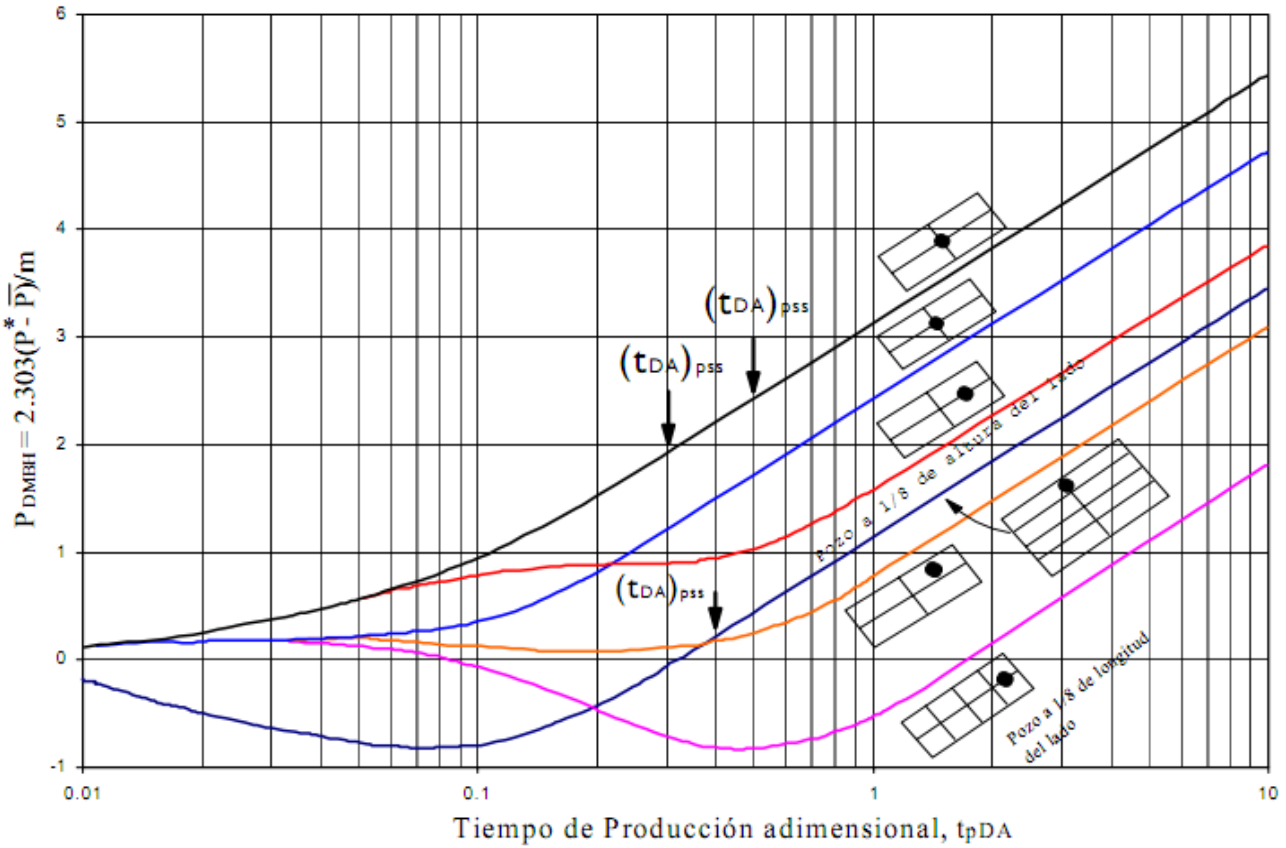
Fuente: Freddy Humberto Escobar M., PhD. Análisis Moderno de Presión de Pozos

Figura 16. P_{DMBH} para un pozo en el centro de áreas de drene cuadradas



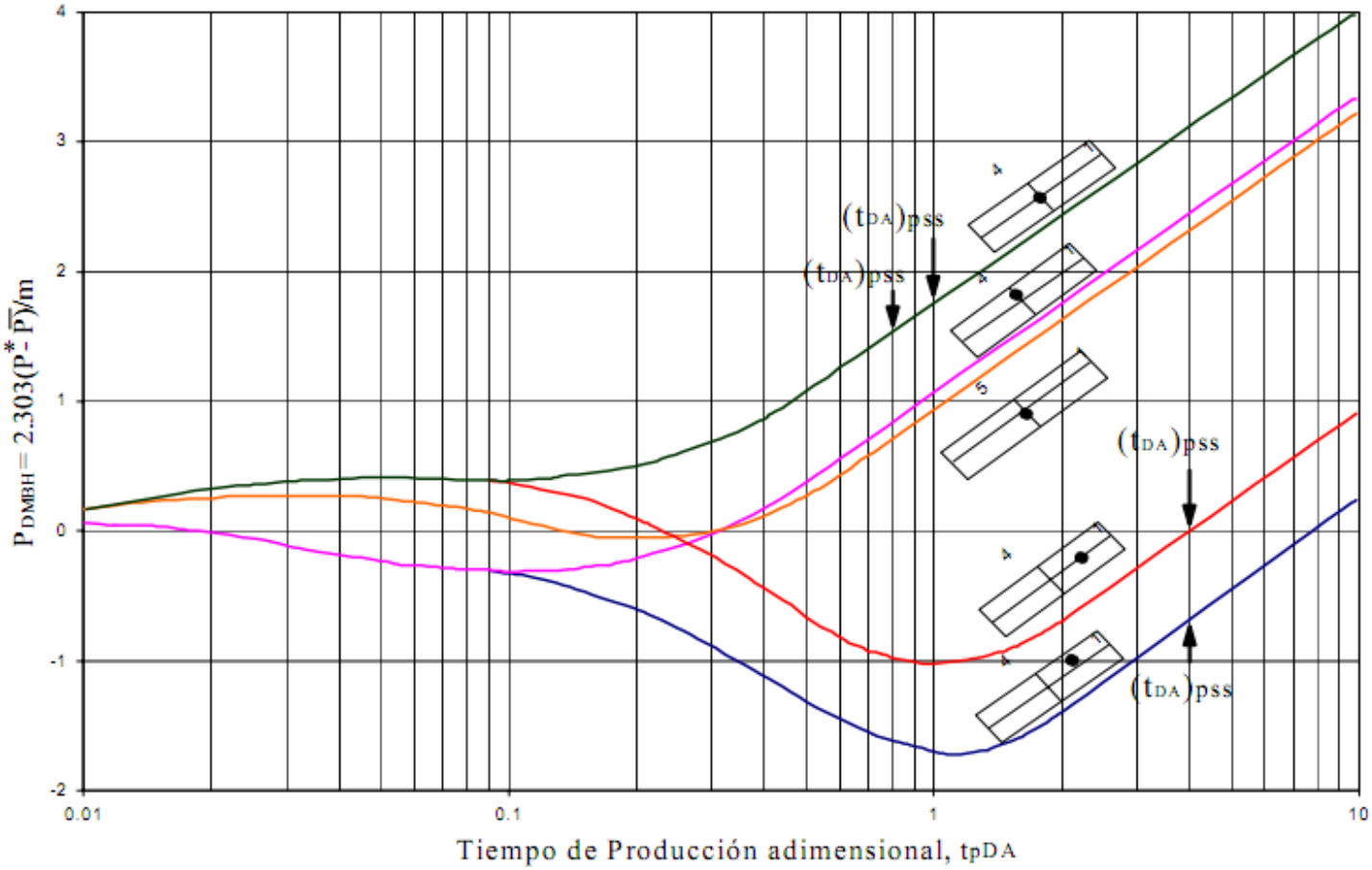
Fuente: Freddy Humberto Escobar M., PhD. Análisis Moderno de Presión de Pozos

Figura 17 P_{DMBH} para un pozo en el centro de áreas de drene rectangulares con relación de lado 2:1



Fuente: Freddy Humberto Escobar M., PhD. Análisis Moderno de Presión de Pozos

Figura 18. P_{DMBH} para un pozo en el centro de áreas de drene rectangulares con relación de lado 4:1 y 5:1

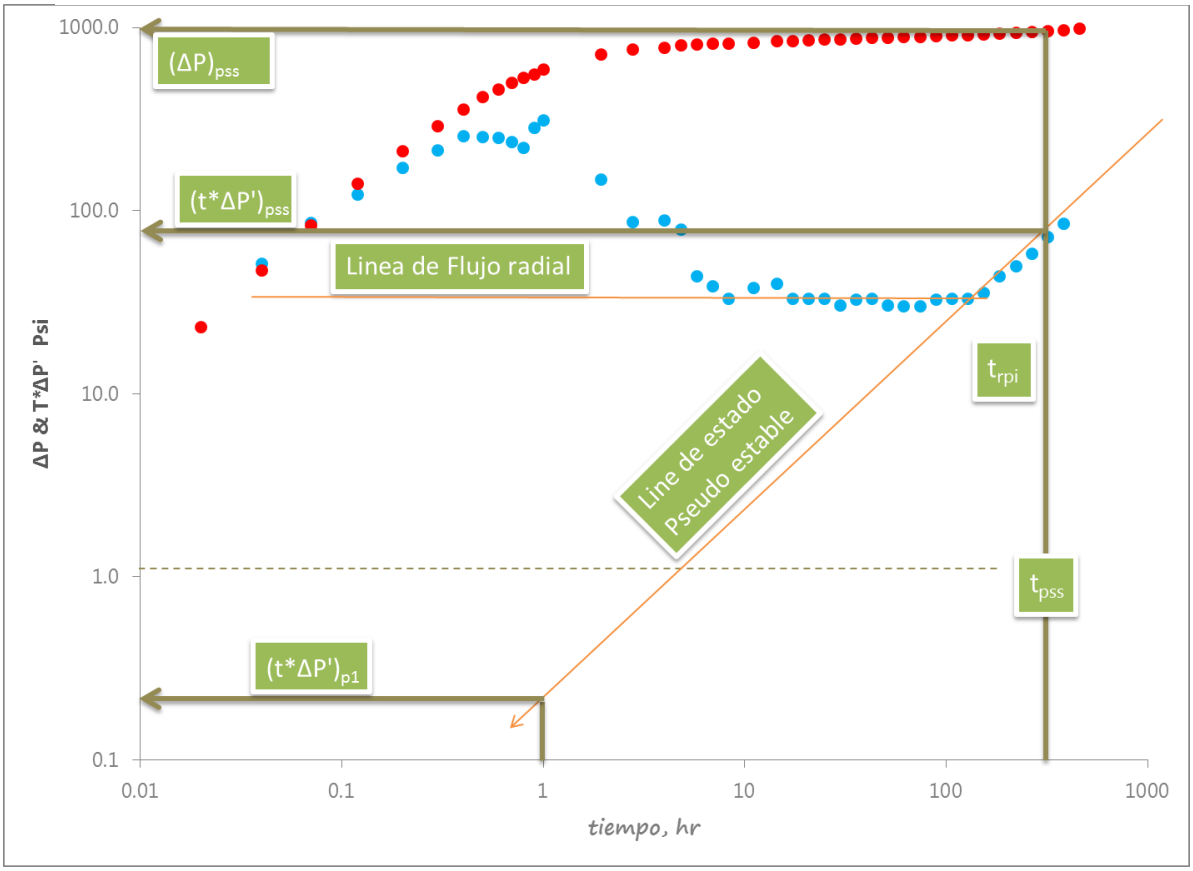


Fuente: Freddy Humberto Escobar M., PhD. Análisis Moderno de Presión de Pozos

2.2.2. Síntesis Directa de Tiab Durante el Estado Pseudoestable

El cálculo de la presión promedio con la técnica TDS se realiza en base a la gráfica log-log de ΔP y derivada de la presión vs el tiempo equivalente.

Figura 35: Prueba de Presión en Estado Pseudoestable



Fuente: Editado de Análisis Moderno de Presiones, Freddy Humberto Escobar PhD.

Yacimientos Circulares Cerrados

Para este tipo de configuración, se procede a calcular el área de drenaje con el valor de la derivada de la presión a tiempo t=1 hora o su extrapolación.

$$A = \frac{0.234\ qB}{\phi C_t h (t * \Delta P')_{p1}}$$

Ecuación 33

Posteriormente, se procede a calcular el valor de la presión promedio, eligiendo algún tiempo de estado pseudoestable.

$$\bar{P} = P_i - \frac{141.2\ q\mu B}{kh}\left[\left(\frac{(t * \Delta P')_{pss}}{(\Delta P)_{pss} - (t * \Delta P')_{pss}}\right) ln\left(\frac{r_e}{r_w} - \frac{3}{4}\right)\right]$$

Ecuación 34

Yacimientos Rectangulares Cerrados

Para esta configuración se obtienen los valores del factor de forma y la presión promedio del yacimiento de esta manera:

$$C_A = \frac{2.2458A}{r_w^2}\left\{e^{\left[\frac{0.003314\ kt_{pss}}{\phi\mu C_t A}\left(\frac{(\Delta P)_{pss}}{(t*\Delta P')_{pss}} - 1\right)\right]}\right\}^{-1}$$

Ecuación 35

$$\bar{P} = P_i - 70.6\frac{q\mu B}{kh}\left[\left(\frac{(t * \Delta P')_{pss}}{(\Delta P)_{pss} - (t * \Delta P')_{pss}}\right) ln\left(\frac{2.2458A}{C_A r_w^2}\right)\right]$$

Ecuación 36

Para yacimientos circulares o rectangulares, se puede usar el punto de intersección entre la línea de flujo radial y la línea del estado pseudoestable (t_{rpi}). La aplicación busca este punto de intercepción y realiza el correspondiente cálculo de la presión promedio.

$$A = \frac{\pi k}{948.05\phi\mu C_t} t_{rpi}$$

Ecuación 37

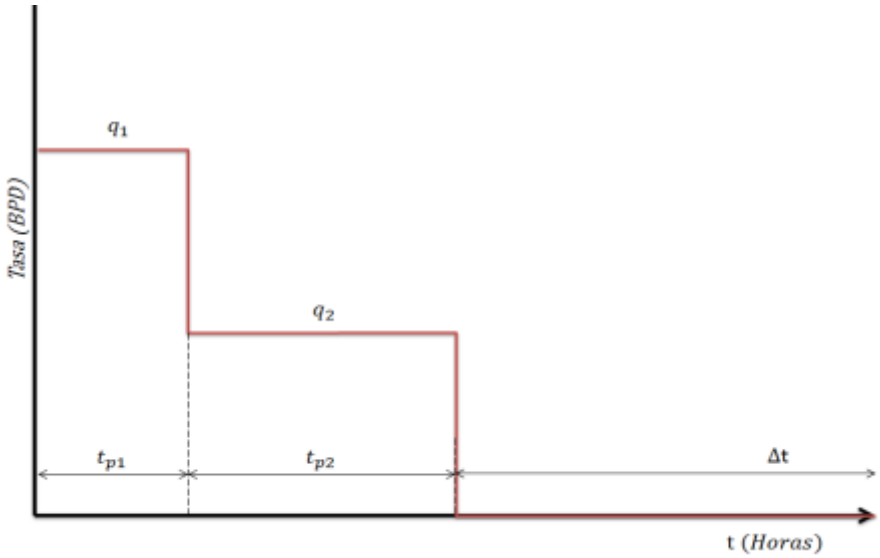
$$\bar{P} = P_i - 70.6\left(\frac{q\mu B}{kh}\right)$$

Ecuación 38

2.3. PRUEBAS PBU CON N-1 TASAS DIFERENTES DE FLUJO

Debido a que operacionalmente es complicado mantener una tasa constante estable antes del cierre, se han venido desarrollando métodos para analizar pruebas de restauración de presión teniendo en cuenta las distintas tasas antes del cierre.

Figura 36: Prueba PBU con n-1 tasas antes del cierre



Fuente: Análisis Moderno de Presiones. Escobar F. 2003

2.3.1. Método de Superposición

Se puede desarrollar una técnica similar de análisis usando la superposición en el tiempo para n-1 tasas antes de la prueba PBU y la formulación para pruebas MultiTasa pero con q_n =0 y n-1 tasas antes del cierre, se tiene:

$$P_i - P_i = \frac{192.6\ q_{n-1}\mu B}{kh}\left[\left(\frac{q_1}{q_{n-1}}\right)log\left(\frac{t}{t-t_1}\right)+\left(\frac{q_2}{q_{n-1}}\right)log\left(\frac{t-t_1}{t-t_2}\right)\right.$$
$$\left.+\left(\frac{q_2}{q_{n-1}}\right)log\left(\frac{t-t_1}{t-t_2}\right)+\left(\frac{q_{n-2}}{q_{n-1}}\right)log\left(\frac{t-t_{n-3}}{t-t_{n-2}}\right)+log\left(\frac{t-t_{n-2}}{t-t_{n\theta\theta-1}}\right)\right]$$

Realizando las respectivas simplificaciones para el caso PBU se obtiene una función de la siguiente manera

$$x=\sum_{j=1}^{n}\frac{q_i}{q_n}\log\left(\frac{t_n-t_{j-1}+\Delta t}{t_n-t_j+\Delta t}\right)$$

Ecuación 39

Siendo n el último dato y tasa registrado antes del cierre. Este método es el más preciso por lo tanto se evitan aproximaciones en la aplicación programando la función de superposición directamente.

Se calcular de MTR permeabilidad y daño

$$k=\frac{162.6q_{n-1}\mu_0 B_0}{mh}$$

Ecuación 40

En este caso n-1 se refiere a la última tasa,

2.3.2. Aproximación de Horner

Esta aproximación parte de la utilización de un tiempo de *Pseudoproduccion*, el cual plantea que todas las tasas previas de producción se pueden condensar con la introducción de este término, de la manera siguiente:

$$t_{pH} = \frac{24\ N_p}{q_{last}}\ \left[\frac{STB}{Dia}\right]$$

Ecuación 41

Este enfoque pretende modelar el efecto de cada tasa, donde la tasa, q, es reemplazada por la ultima tasa, q_{last}, y el tiempo actual de producción, t, es reemplazado por el *Tiempo de Pseudoproduccion de Horner*, t_{pH}. La ecuación planteada que modela el fenómeno es:

$$P_{ws} = P_i - \frac{162.6\ q_{last} B\mu}{kh} log\left(\frac{t_{pH} + \Delta t}{\Delta t}\right)$$

Esta aproximación asume que el tiempo total de producción t_p es mucho mayor que el tiempo de cierre, y que antes del build up, el periodo de flujo fue llevado a tasa constante y que la presión se estabilizó al tiempo t_p. Lo cual raramente ocurre.

3. ANALISIS COMPLEMENTARIOS

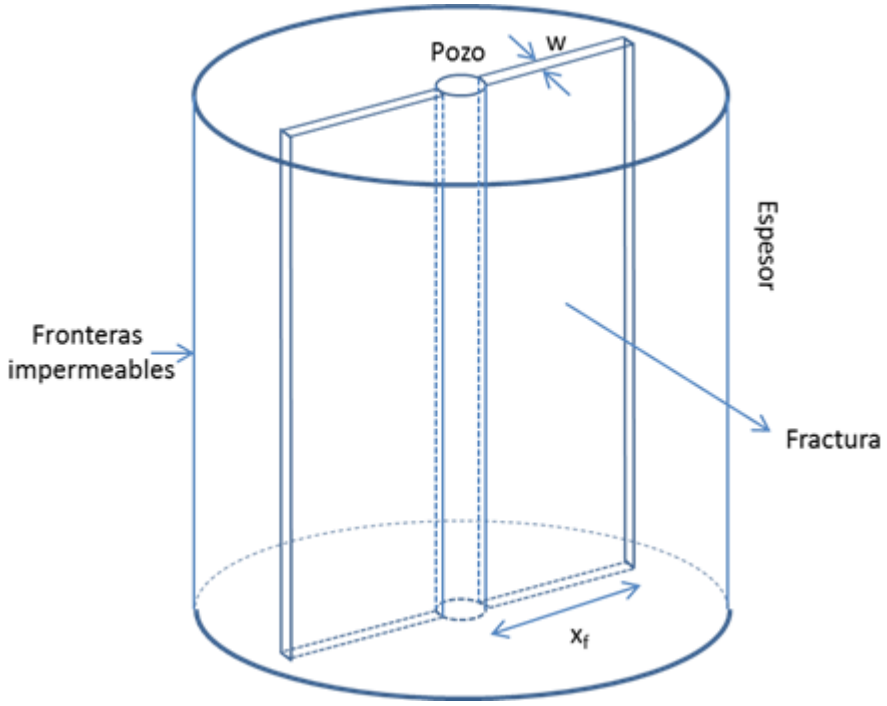
En esta sección presentamos algunas de las características adicionales que el software ofrece, estas son realizadas sobre la pruebas PDD Y PBU luego de realizar el análisis habitual se procede a calcular parámetros adicionales de pozo o yacimiento.

En la herramienta se implementan los siguientes modelos:

3.1. PRUEBAS DE PRESIÓN PARA FRACTURAMIENTO HIDRÁULICO

- El principal objetivo de la estimulación, es aumentar la productividad del pozo creando un camino altamente conductivo a una distancia más allá de la zona de daño.
- La fractura crea más área superficial al Wellbore sin perforar otro pozo
- Debido a que hay más área del yacimiento en comunicación directa con el Wellbore, se puede producir una mayor cantidad de fluido por unidad de tiempo.
- El objetivo básico no ha cambiado desde que el fracturamiento hidráulico fue introducido en los 50's.
- Al principio el fracturamiento hidráulico fue considerado un buen medio para aumentar la productividad de los pozos completados en yacimientos de baja permeabilidad
- Ahora, se ha convertido en parte integral de la mayoría de los completamientos.

Figura 37: fractura vertical ideal



Fuente: Editado de Presentaciones Clases Análisis de Presiones UIS. Ortiz O. 2011

El producto de la permeabilidad de la fractura k_f y su amplitud w_f, se conoce como la conductividad de la fractura, C_f. $C_f=k_f*w_f$

Un fracturamiento hidráulico es exitoso si la conductividad de la fractura es al menos 10000 veces más grande que la conductividad de la formación (kh)

$C_f=k_f*w_f=10000\ kh$

El fracturamiento también se considera exitoso si el factor skin se reduce por lo menos a -3.

Es importante enfatizar que el fracturamiento no altera la permeabilidad del yacimiento de ninguna manera. Básicamente, el fracturamiento aumenta el radio efectivo del pozo rw'.

Se consideran tres tipos principales de fracturas:

- Fracturas de flujo uniforme

- Fracturas de conductividad infinita
- Fracturas de conductividad finita

3.1.1. Fracturas de Flujo Uniforme

En algunas fracturas, el fluido entra a ésta a una tasa uniforme por unidad de área de la cara de la fractura, de manera que hay una caída de presión en la fractura.

En este caso la fractura se conoce como “fractura de flujo uniforme”.

Esta situación se da prácticamente por conveniencia matemática ya que la distribución del flujo a lo largo de la fractura está lejos de ser uniforme.

3.1.2. Fracturas de Conductividad Infinita

Se asume que algunas fracturas tienen permeabilidad infinita (conductividad) y, por lo tanto presión uniforme a lo largo de ella.

Las fracturas con una conductividad adimensional (C_{fD}) mayor de 300 se consideran de conductividad infinita.

$$C_{fD} = F_{CD} = \frac{k_f w_f}{k x_f}$$

Ecuación 42

$x_f = Longitud\ media\ de\ la\ fractura$

$w_f = Amplitud\ de\ la\ fractura$

$k_f = permeabilidad\ de\ la\ fractura$

El siguiente criterio se puede usar para estimar la efectividad de un fracturamiento:

Tabla 8: Cuantificación del éxito de un fracturamiento

F_{CD}‹ 10	Fracturamiento pobre
10 ‹ F_{CD} ‹50	Fracturamiento bueno a excelente
F_{CD} › 50	Fracturamiento excelente

Fuente: Apuntes Curso Análisis de Presiones, Ortiz Olga

Este tipo de fractura se da con mucha frecuencia, especialmente en formaciones con baja permeabilidad. Razón por la cual fue el modelo seleccionado a aplicar en la herramienta.

La respuesta de presión de un pozo que atraviesa una fractura de conductividad infinita es muy similar al caso de la fractura de conductividad finita, excepto por la presencia del flujo bilineal, el cual no está presente.

La respuesta en una fractura de conductividad infinita se caracteriza por un verdadero flujo lineal, Tal respuesta muestra una línea recta de pendiente ½ en una gráfica log-log de caída de presión vs tiempo.

Más allá del periodo de flujo lineal, la repuesta pasará por una transición hacia flujo radial infinito.

En variables reales la ecuación que modela el flujo lineal viene dada por:

$$P_W = P_i + m_{vf}\sqrt{t} \qquad m_{vf} = \frac{-4.064\mathrm{qB}}{\mathrm{h}}\sqrt{\frac{\mu}{\mathrm{k}\emptyset\mathrm{c}_t\mathrm{x}^2{}_f}}$$

Ecuación 43

3.1.3. Síntesis Directa de TIAB

Por medio de puntos característicos en la derivada se puede caracterizar el éxito del fracturamiento hidráulico, el flujo de trabajo es el siguiente:

Paso 1. Grafique la derivada (t*ΔP') y el cambio de presión del pozo (ΔP) vs el tiempo.

Paso 2. Lea el valor de la derivada en flujo radial ($t*\Delta P')_r$

Paso 3. Calcule la permeabilidad de la Ec. 2.2

Paso 4. Obtener el valor de (t*ΔP') a un tiempo t = 1 hr de la línea de flujo lineal (extrapolada si es necesario), ($t*\Delta P')_{L1}$

Paso 5. Calcule la longitud media de la fractura con la siguiente ecuación

$$x_f = \frac{2.032\beta q}{h(t * \Delta P')_{l1}}\left(\frac{\mu}{\varphi C_t k}\right)^{0.5}$$

Ecuación 44

Paso 6. – Determinar el tiempo de intersección de la línea de flujo lineal y radial de la gráfica, por ejemplo t_{LRi}

Paso 7. Calcule la relación x_f^2/k, calcule de nuevo esta misma relación con los pasos 3 y 5 Si esas relaciones son significativamente diferentes, trasladar una o ambas líneas rectas, y repetir los pasos del 2 al 7 hasta que las relaciones sean iguales. Es recomendable trasladar la línea de flujo lineal debido a que esta presenta mayor incertidumbre por ruido en la deriva, estado mecánico del pozo, Skin y efectos de almacenamiento.

$$\frac{x^2{}_f}{k} = \frac{t_{LRi}}{1207\varphi\mu C_t}$$

Ecuación 45

Paso 8. Lea el valor de ($t*\Delta P')_{p1}$ de la línea de estado pseudoestable (extrapolada si es necesario). Calcule el área de drene.

Si la prueba alcanza el estado pseudoestable es posible calcular el área de drenaje.

3.1.4. Fracturas de Conductividad Finita

Este modelo es aplicable la mayoría de los casos, a menos que la permeabilidad de la formación sea extremadamente baja (en el rango de microdarcys).

3.1.4.1. Regímenes de Flujo en Pozos Fracturados

El flujo de fluidos en pozos verticalmente fracturados puede ser representado por cuatro regímenes de flujo como muestra la figura 24 los 4 regímenes de flujo son:

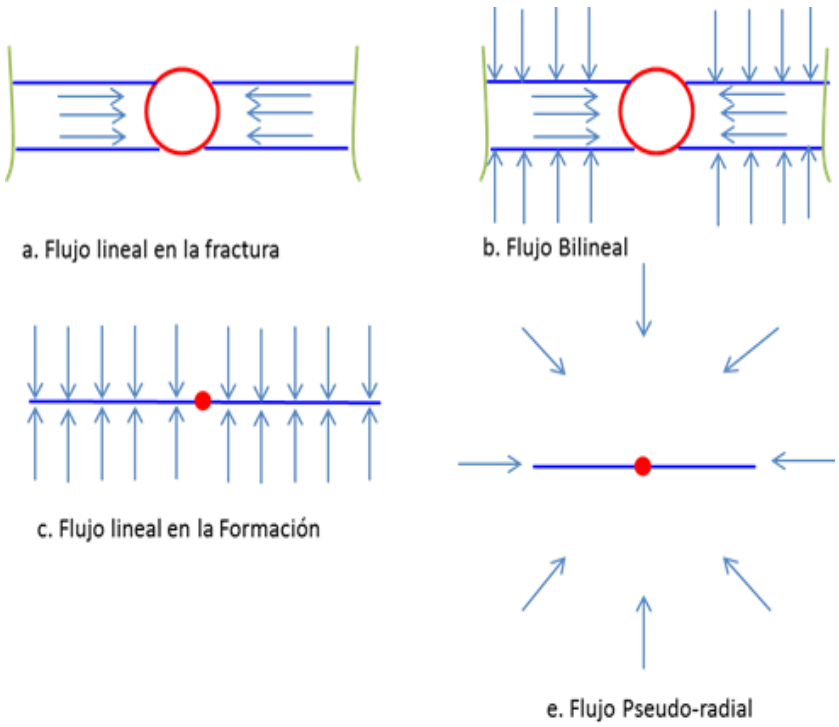
1. Flujo lineal en la fractura
2. Flujo Bilineal
3. Flujo Lineal en la formación
4. Flujo Pseudo Radial

El flujo Bilineal ocurre en las fracturas de conductividad finita (C_{fD}<300), Esta caracterizado por tener una porción recta de pendiente 0.25 en una gráfica Log-Log. Con el flujo bilineal es posible calcular la conductividad de la fractura mediante la siguiente ecuación:

$$k_f w_f = \frac{121.74}{\sqrt{\varphi \mu C_t k}} \left(\frac{q \mu B}{h(t * \Delta P')_{BL1}{}^2} \right)$$

Ecuación 46

Figura 38: Tipos de Flujo presentes en una fractura



Fuente: Editado de Petroleum Reservoir Engineering Practice. Ezekwe N. 2011

3.2. YACIMIENTOS NATURALMENTE FRACTURADOS (YNF)

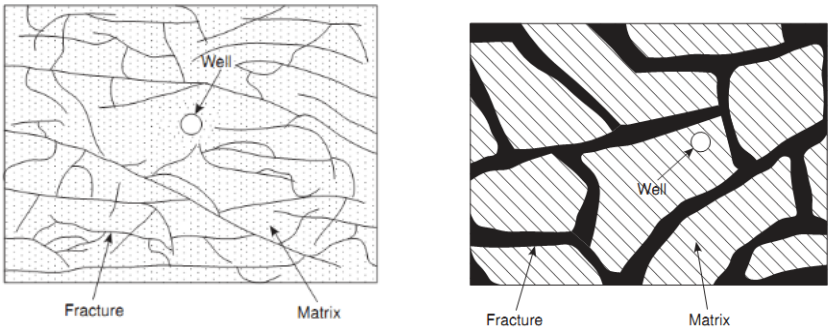
Muchos yacimientos en el mundo son naturalmente fracturados, la productividad de estos yacimientos depende de estas fracturas, las fracturas actúan como conductos que llevan el fluido de la matriz al pozo, las tasas de producción de estos pozos son relativamente altas, aunque estos pozos tienen vidas cortas debido a la canalización de agua o gas a través del sistema de fracturas hasta el pozo.

La naturaleza de las fracturas y su distribución tiene considerable influencia sobre la respuesta del pozo en una prueba de presión. En pruebas de análisis de presión, los yacimientos naturalmente fracturados pueden ser representados con diferentes modelos basados en el tipo y la distribución de fracturas dentro del sistema estos modelos son:

3.2.1. Modelo de Yacimiento Homogéneo

Puede ser usado para representar formaciones masivamente fracturadas con pequeños bloques de matriz o formaciones donde la mayoría de los fluidos están almacenados en la fractura en este tipo de YNFs, la fractura y la matriz de la roca se comporta como un solo sistema indistinguible. Consecuentemente todas las técnicas de pruebas de presión compuestas de método de líneas rectas o ajustes de curvas tipo pueden ser aplicados a YNFs

Figura 39: Modelo de Yacimiento Homogéneo

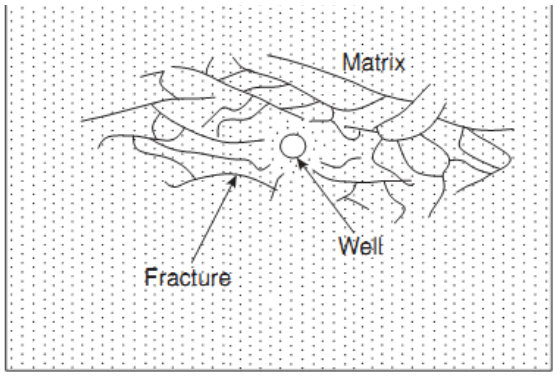


Fuente: Tomado de Petroleum Reservoir Engineering Practice. Ezekwe N. 2011

3.2.2. Modelo de Múltiples Regiones o Yacimiento Compuesto

En la figura 26 se observa un yacimiento compuesto de dos regiones. La formación esta fracturada en una región, mientras que el resto del yacimiento no está fracturado. La región fracturada probablemente exhibirá una transmisibilidad más alta que la región no fracturada.

Figura 40: Modelo de yacimiento compuesto



Fuente: Tomado de Petroleum Reservoir Engineering Practice. Ezekwe N. 2011

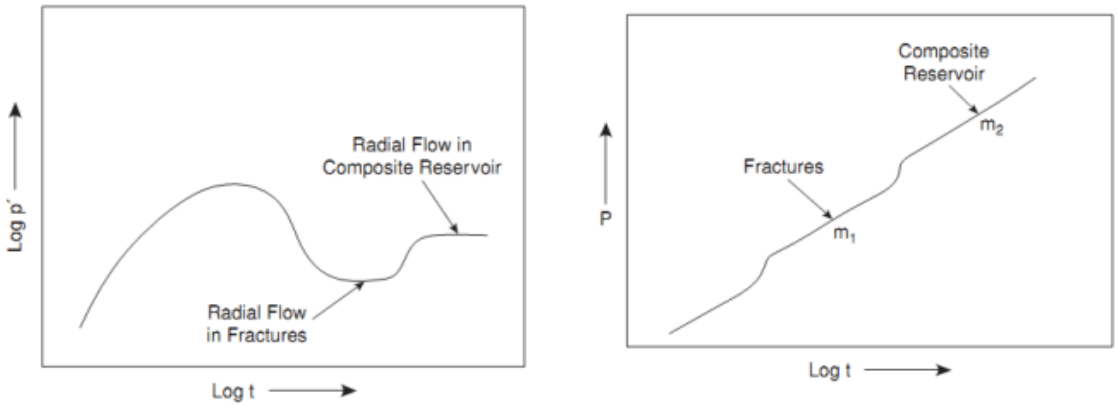
Si se conduce una prueba sobre un pozo localizado en la región fracturada, la respuesta de la presión será influenciada inicialmente por las fracturas cercanas, y más tarde por la región no fracturada si la duración de la prueba es lo suficientemente larga. En la gráfica log-log de la derivada de presión la primera línea horizontal representa el flujo radial del sistema de la fractura después de que los efectos de almacenamiento han terminado. Después de un periodo de transición, la segunda línea horizontal representa el flujo radial de la región fracturada y sin fracturas (Sistema Compuesto) del yacimiento. En la gráfica semilog de la respuesta de presión la primera línea recta representa el sistema de fracturas y la segunda línea representa el yacimiento compuesto. Las capacidades de flujo pueden ser calculadas de las pendientes de la línea recta.

3.2.3. Modelo de Yacimiento Anisotropico

Las fracturas en algunos yacimientos están alineadas en una dirección particular. En NFR exhibiendo un comportamiento Anisotropico, las pruebas de interferencia son los mejores métodos para determinar K_{max} y k_{min} más la orientación del eje principal de la permeabilidad

.

Figura 41: Comportamiento tipo de un yacimiento naturalmente fracturado en la derivada y en grafico semilog



Fuente: Tomado de Petroleum Reservoir Engineering Practice. Ezekwe N. 2011

3.2.4. Modelo de Fractura Simple

Este sistema de fractura puede consistir de una falla permeable a través de la cual los fluidos pueden fluir al pozo desde otras partes del yacimiento.

Distintos parámetros del yacimiento y del pozo pueden ser calculados de un análisis de una prueba de presión en este yacimiento. En la gráfica Log-Log de la derivada de la presión hay un régimen de flujo radial representando el sistema entero. Después de un periodo de transición, la respuesta a la falla permeable parece indicar un límite de presión constante. Esto es seguido por un flujo bilineal representado a través del sistema de fractura. Distintas graficas especializadas son útiles para el análisis de pruebas de flujo sobre el pozo. Esto incluye una gráfica semilog para calcular la capacidad de flujo, kh, o calcular la conductividad de la fractura.

3.2.5. Pruebas de Análisis de Presión basados en el modelo de doble porosidad

Algunos yacimientos naturalmente fracturados exhiben flujo transitorio de la matriz a la fractura. Las condiciones de flujo transiente implican que la tasa de cambio de presión varía con el tiempo y la ubicación de la matriz. Los métodos semilog son frecuentemente imprácticos debido a que los efectos de almacenamiento de pozo esconden la línea recta que aparece a tiempos tempranos. El grafico de la derivada proporciona un método mucha más práctico para estimar ω y λ. La posición del mínimo en la derivada (el dip característico del comportamiento de doble porosidad) define completamente los valores ω y λ. El mínimo en la derivada puede ser expresado en términos adimensionales.

$$\left[t_D \frac{dP_D}{dt_D}\right]_{min} = \frac{1}{2}\left[1 + \omega^{\frac{1}{1-\omega}} - \omega^{\frac{\omega}{1-\omega}}\right]$$

Ecuación 47

Y el valor de λ puede ser obtenido de

$$\lambda = \frac{\omega}{t_{Dmin}} \ln\left(\frac{1}{\omega}\right)$$

Ecuación 48

Para resolver la ecuación 47 el software hace cálculos de la derivada y el tiempo en su forma adimensional, se determina el punto mínimo de la derivada y luego se emplea una técnica numérica para resolver esta ecuación no lineal y hallar ω y seguidamente hallar λ.

3.3. POZOS PARCIALMENTE CAÑONEADOS

Suposiciones

- El intervalo sobre el yacimiento que fluye dentro del pozo es más corto que el espesor del yacimiento, debido al completamiento parcial.
- El modelo tiene en cuenta el efecto de almacenamiento y el factor skin, y supone un yacimiento infinito.

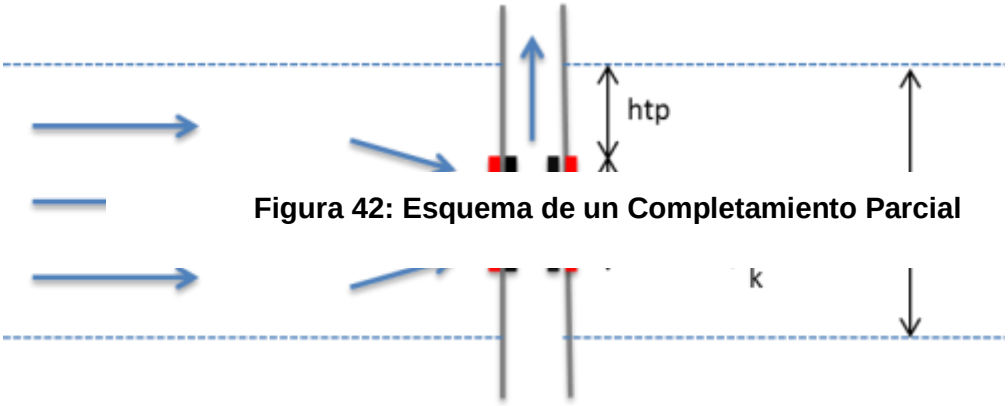


Figura 42: Esquema de un Completamiento Parcial

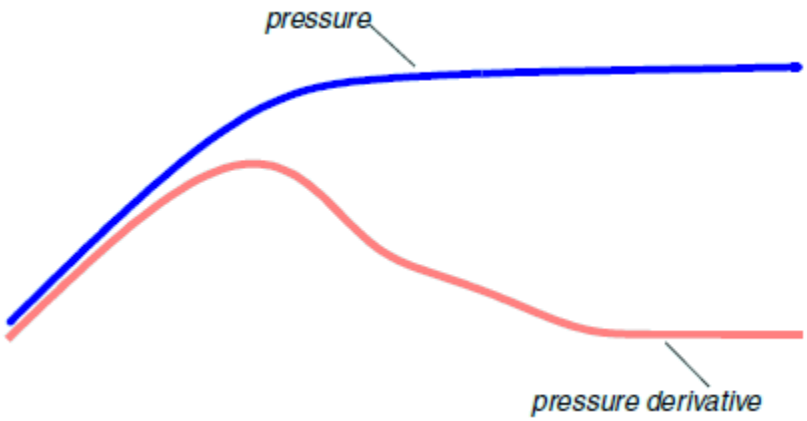
Fuente: imagen editada de Welltest 200 Technical Description. 2001

Comportamiento

A tiempos tempranos, después de que los efectos de almacenamiento son vistos, el flujo es esférico o hemisférico, dependiendo de la posición del intervalo de flujo. El flujo hemisférico se desarrolla cuando uno de los limites verticales de no flujo está mucho más cerca que el otro al intervalo de flujo. Cualquiera de estos dos regímenes de flujo se caracteriza por una pendiente de -0,5 sobre una gráfica log-log de la derivada de la presión.

A tiempos tardíos, el flujo es radial cilíndrico. El comportamiento es como el de un pozo totalmente completado en un yacimiento infinito con un skin igual al skin total del sistema.

Figura 43: Comportamiento de la derivada de un pozo parcialmente completado



Fuente: Tomado de Welltest 200 Technical Description. 2001

3.3.1. Análisis Convencional

Para realizar un análisis de Penetración parcial y completamiento parcial se debe hacer el análisis convencional y calcular permeabilidad y daño con las ecuaciones del capítulo 2, para calcular la permeabilidad esférica se debe realizar un gráfico cartesiano de:

$$P_{wf}\ vs\ \frac{1}{\sqrt{t}}$$

Este grafico produce una línea recta, con el valor de la pendiente obtenida se puede calcular la permeabilidad esférica.

$$K_{sp} = \left(\frac{2453 q\mu\beta}{m}\sqrt{\varphi\mu c_t}\right)^{2/3}$$

Ecuación 49

La permeabilidad vertical se estima de la siguiente relación:

$$K_{sp} = \sqrt[3]{k_v {k_h}^2}$$

Ecuación 50

Para el cálculo del r_{sw}

$$r_{sw} = \frac{bh}{2\ln\left(\frac{bh}{r_w}\right)}$$

Ecuación 51

Una vez obtenido este parámetro es posible calcular el Skin esférico:

$$S_{sp} = \frac{(P_i - I)k_{sp}r_{sw}}{70.6q\mu\beta} - 1$$

Ecuación 52

Con el valor de la permeabilidad vertical se puede estimar el daño causado por penetración parcial

$$h_D = \left(\frac{k_h}{k_v}\right)^{0.5}\left(\frac{h}{r_w}\right) \quad Ec.4.9$$

Ecuación 53

$$G = 2.948 - 7.363(b) - 11.45(b)^2 - 4.675(b)^3$$

Ecuación 54

$$S_c = \left(\frac{1}{\frac{b}{h}} - 1\right)[lnh_D - G]$$

Ecuación 55

3.3.2. Síntesis Directa de TIAB

En el método de Tiab se leen los puntos característicos del flujo esférico

La permeabilidad esférica:

$$k_{sp} = \left(1227 \frac{qB\mu}{(t * \Delta P')_{sp}} \sqrt{\frac{\varphi \mu C_t}{t_{sp}}} \right)^{2/3}$$

Ecuación 56

Donde $(t * \Delta P')_{sp}$ es un punto de la derivada en flujo esférico en un valor conveniente

El cálculo el daño:

$$S_{sp} = 34.74 \sqrt{\frac{\varphi \mu C_t r_{sw}^2}{k_{sp} t_{sp}}} \left[\frac{(\Delta P_w)_{sp}}{2(t * \Delta P')_{sp}} \right] - 1$$

Ecuación 57

La técnica de Tiab puede emplear las ecuaciones 55-57 para calcular el daño creado por el completamiento.

La aplicación está en la capacidad de disgregar el daño en sus componentes. Existen diferentes correlaciones no lineales para lograr este fin, sin embargo la más usada es la ecuación lineal de la cual hace uso la aplicación.

$$S_t = \frac{Sm}{b} - S_c + S_{sp}$$

Ecuación 58

De esta manera es posible cuantificar el efecto de cada tipo de skin y su contribución al daño total, el software reporta cada uno de estos valores.

ANEXO B: MANUAL DE USUARIO PARA EL SOFTWARE DE ANALISIS DE PRESIONES

1. DESCRIPCIÓN DE LA HERRAMIENTA

Este manual está diseñado para el uso adecuado del Software de Análisis de Presiones desarrollado en el proyecto de grado titulado Herramienta de Software Para El Análisis de Pruebas de Presión. Si desea conocer las metodologías y ecuaciones utilizadas en esta herramienta, remítase al MANUAL DE REFERENCIA PARA EL SOFTWARE DE ANALISIS DE PRESIONES.

2. REQUERIMIENTOS

Para la utilización correcta de esta herramienta, es necesario cumplir ciertos requisitos de software:

- Sistema Operativo Windows XP/Vista/7.
- .Net Framework 4.0 o superior
- Adobe Acrobat 9 o Superior.
- Editor de Archivos Planos.
- Reproductor de Windows Media Player.

2.1. DESCRIPCIÓN DE LA HERRAMIENTA

La herramienta está diseñada para el análisis de pruebas de aceite negro, en yacimientos de una sola capa, sin flujo cruzado y monofásico.

La herramienta consta de 4 módulos:

1. Análisis de pruebas de caída de presión (PDD) para una sola tasa.
2. Análisis de pruebas de caída de presión para múltiples tasas.
3. Análisis de pruebas de restauración de presión (PBU) para una sola tasa.
4. Análisis de pruebas de restauración de presión para múltiples tasas.

Dentro del módulo 1 se podrán realizar análisis para yacimientos naturalmente fracturados, Pozos parcialmente cañoneados e Hidráulicamente Fracturados

Cada módulo cuenta con una interfaz de usuario independiente y permite la introducción de los datos de la prueba a través de un archivo de texto plano (*.txt), la visualización de los datos a través de gráficas generadas, la introducción de parámetros de yacimiento por medio de una ventana de lectura. La interfaz de cada módulo permite el cálculo de regresión lineal de los conjuntos de puntos elegidos por el usuario, el valor de la pendiente, factor de regresión y cuando es relevante, el valor del corte con el eje Y.

Cada interfaz ofrece la posibilidad de la introducción de puntos característicos necesarios para el análisis de las gráficas generadas por el software, con el fin de realizar los cálculos pertinentes (convencionales y modernos). Los resultados se presentan en la interfaz propia, a través de una tabla que puede ser copiada en el portapapeles, las gráficas generadas por la herramienta también permiten ser exportadas en formato de imagen (*png), con el fin de ser incluidas en algún reporte.

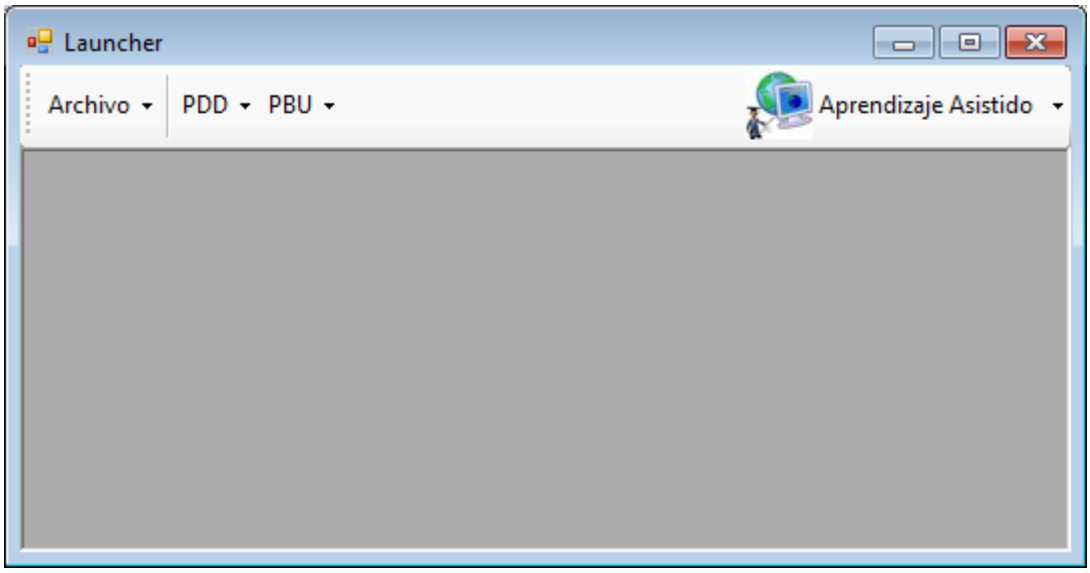
2.2. CARACTERÍSTICAS Y CAPACIDADES DEL SOFTWARE

Tabla 9: Características y Capacidades del Software

Parámetro	PDD 1 Tasa	PDD MultiTasa	PBU 1 Tasa	PBU MultiTasa
k	OK	OK	OK	OK
s	OK	OK	OK	OK
C	OK	OK	OK	-
j_{ideal}	OK	-	OK	-
j_{actual}	OK	-	OK	-
EF	OK	-	OK	-
DR	OK	-	OK	-
θ	OK	-	-	-
dc	OK	-	OK	-
df	OK	-	-	-
A	OK	-	-	-
C_A	OK	-	-	-
P_{prom}	-	-	OK	-
ω	OK	-	-	-
λ	OK	-	-	-
x_f	OK	-	-	-
k_{sp}	OK	-	-	-
S_{sp}	OK	-	-	-
K_v	OK	-	-	-
S_c	OK	-	-	-

2.3. INTERFAZ INICIAL

Figura 44: Interfaz Inicial



Aquí se pueden observar los botones de acceso a los módulos para el análisis de pruebas PDD, PBU, además de la posibilidad de ejecutar los manuales de usuario y de referencia, junto con tutoriales en video para facilitar la utilización de la herramienta.

2.4. INTERFAZ MÓDULO PDD TASA CONSTANTE

2.4.1. INTRODUCCIÓN

La interfaz del módulo para el análisis de Pruebas de Caída de Presión (PDD) con tasa constante, permite la carga de los datos a través de un archivo de texto plano (*.txt), además de la visualización de los mismos en el panel de datos y en forma gráfica a través de las curvas generadas: cartesiana (P_{wf} vs t), semilog (P_{wf} vs t en el eje logarítmico) y log-log (ΔP & (t* ΔP') vs t) La interfaz ofrece la posibilidad de desplegar graficas individuales, sobre las cuales se puede dibujar rectas, calcular pendientes con su respectivo factor de regresión, definir regiones de flujo (ETR,

MTR y LTR), introducir puntos característicos de las curvas y realizar cálculos de los parámetros de interés, principalmente k y S.

2.4.2. LECTURA DE DATOS

La forma de leer los datos de la prueba de presión se debe realizar por medio de un archivo de texto plano (*.txt), en el cual se ingresan los valores en 2 columnas sin encabezado y separadas por el símbolo (;). La primera columna debe corresponder al tiempo, la segunda a la presión de fondo medida al tiempo correspondiente:

Tabla 3: Conjunto de datos para una prueba PDD una tasa

0	;	2733
0.1	;	2703
0.2	;	2672
0.3	;	2644
0.4	;	2616
0.65	;	2553
1	;	2500
1.5	;	2440
2	;	2398
3	;	2353
4	;	2329
5	;	2312
7	;	2293
9.6	;	2291
12	;	2290
16.8	;	2287
33.6	;	2282
50	;	2279

72	;	2276
85	;	2274
100	;	2272

La forma de cargar los datos se lleva a cabo activando el botón *Archivo/Cargar Data*, luego de haber generado el archivo con los datos de la prueba.

Posteriormente, observará una ventana que le solicitará la elección de un factor de suavizado (el cual deberá estar entre 0 y 0.5 según la literatura[5]), por medio del cual se eliminarán puntos que generen "ruido" en la gráfica de la derivada, para más información, remítase a *Smoothing* en el manual de referencia. Un factor recomendado es de 0.2, pero si desea no realizar suavizado, digite un valor de 0.

Luego, la interfaz le mostrará una ventana de cuadro de diálogo, en la cual el usuario deberá buscar el archivo previamente preparado con los datos de la prueba de presión y seleccionarlo para cargarlo.

Finalmente la interfaz PDD una tasa se mostrará nuevamente con los datos cargados en una tabla, junto con los datos calculados y las gráficas de los mismos.

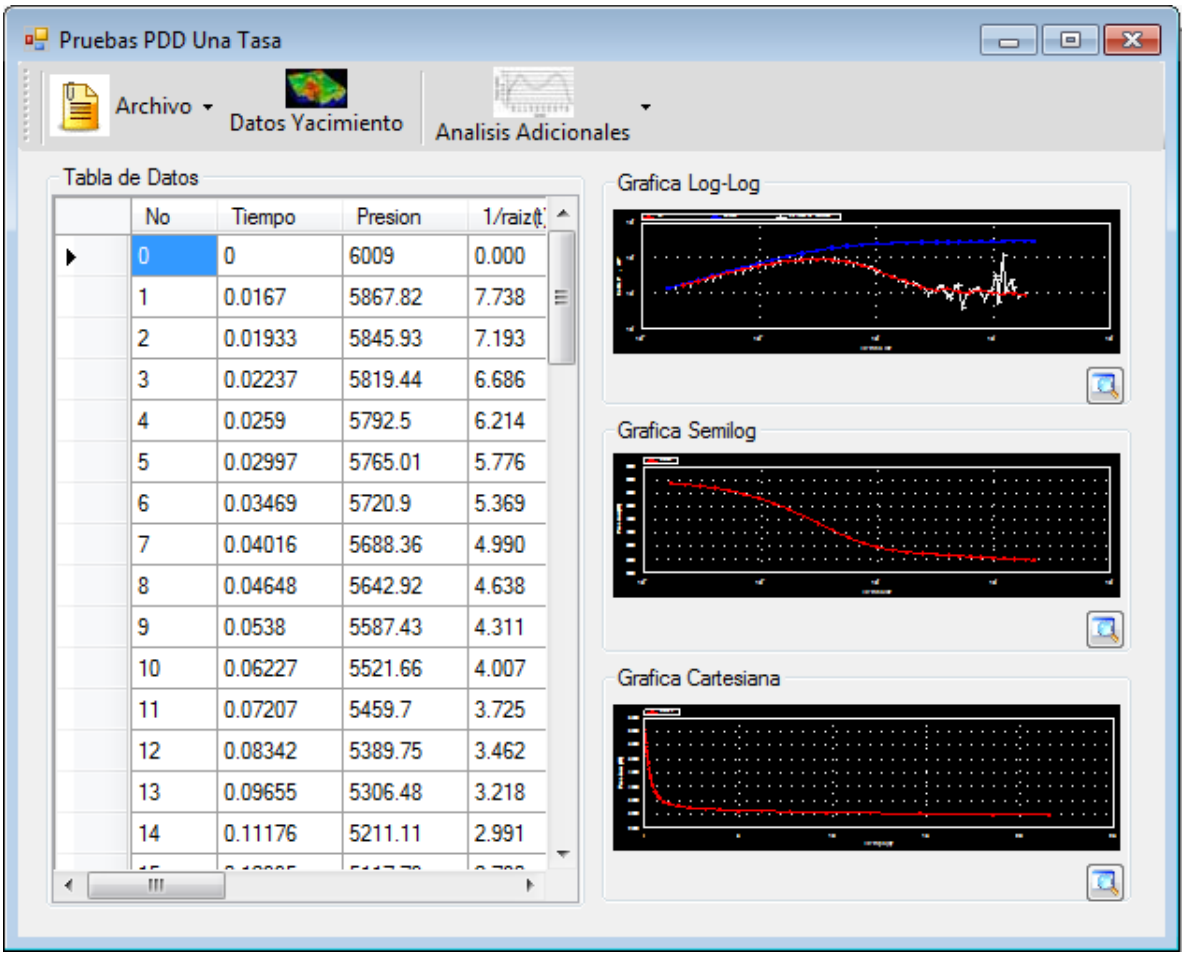
2.4.3. VISUALIZACIÓN DE DATOS

Luego de la lectura de los datos de la prueba de presión (Sección 2.4.2), la interfaz principal del módulo PDD Una Tasa permite la visualización del panel de datos con la información de la prueba y con otros parámetros calculados para el análisis, además de mostrar de manera simultánea las gráficas cartesiana, semilog y log-log; aunque también es posible ver cada grafica en una ventana

[5] Roland N. Horne. Modern Well Test Analysis A Computer – Aided Approach. 1990 4 Ed. Pág. 49.

independiente para su análisis, por medio de un botón en la parte inferior de cada grafica en la ventana principal.

Figura 45: Interfaz Modulo PDD Una Tasa



No	Tiempo	Presion	1/raiz(t
0	0	6009	0.000
1	0.0167	5867.82	7.738
2	0.01933	5845.93	7.193
3	0.02237	5819.44	6.686
4	0.0259	5792.5	6.214
5	0.02997	5765.01	5.776
6	0.03469	5720.9	5.369
7	0.04016	5688.36	4.990
8	0.04648	5642.92	4.638
9	0.0538	5587.43	4.311
10	0.06227	5521.66	4.007
11	0.07207	5459.7	3.725
12	0.08342	5389.75	3.462
13	0.09655	5306.48	3.218
14	0.11176	5211.11	2.991

2.4.4. LECTURA DE PARÁMETROS DE POZO Y YACIMIENTO

Para realizar un análisis de la prueba de presión, es necesario introducir parámetros de la prueba, del pozo y yacimiento que son adicionales:

- Viscosidad del fluido (cP).
- Factor Volumétrico de Formación del Aceite (Rb/STB).

- Espesor de la Formación (ft).
- Porosidad (Fracción).
- Compresibilidad Total (Psi^{-1}).
- Radio del Wellbore (ft).
- Tasa de Producción a la cual se tomó la prueba (STBD).

Luego de introducir los datos requeridos, se oprime el botón de cargar datos, la herramienta se encargará de realizar una validación de los datos, que consiste en evaluar las valores ingresados con respecto a los rangos establecidos, tal como la porosidad que se encuentra entre 0-0.4

2.4.5. FLUJO DE TRABAJO PARA EL ANÁLISIS DE LA PRUEBA DE CAIDA DE PRESION CON UNA SOLA TASA

Luego de haber realizado la carga de los datos de la prueba a través del archivo de texto plano (Sección 2.4.2) y del ingreso de los parámetros adicionales (Sección 2.4.4), el siguiente paso es el análisis de la prueba de presión. Con el fin de llevar a cabo este objetivo, se propone una serie de flujos de trabajo a seguir con esta herramienta, uno para análisis convencional y otro para el análisis moderno.

2.4.6. ANÁLISIS CONVENCIONAL

Uno de los enfoques y objetivos de la herramienta es permitir realizar un análisis de una prueba de presión de manera convencional, con la combinación de las gráficas *Cartesiana, Semilog, Log-Log*. Por medio de este análisis es posible hallar parámetros como Coeficiente de Almacenamiento, *Daño, Permeabilidad, Área de Drenaje, Factor de Forma y Distancia a la Primera Frontera*.

2.4.6.1. ANALISIS CON LINEAS RECTAS

Con el fin de realizar una identificación clara de las regiones de flujo, la herramienta permite la utilización de un algoritmo de regresión lineal para el análisis de estos regímenes, esto con el fin de identificar pendientes características de cada régimen de flujo; para conseguir este objetivo, la herramienta presenta el valor de la pendiente del intervalo de puntos escogidos y un valor del factor de regresión, donde un factor cercano a uno (1) indica un buen ajuste, esto con el fin de que el usuario determine a su criterio el intervalo con el mejor ajuste.

Los valores de inicio y fin del intervalo a analizar pueden ser leídos de la gráfica con solo colocar el cursor de mouse sobre el punto, esto se puede facilitar realizando un zoom en la ventana de la gráfica con ayuda del mouse y un cuadro de zoom que se muestra al mantener presionado el botón izquierdo del mouse y desplazarse a través de la ventana de la gráfica. Para facilitar la experiencia de usuario, al colocar el cursor cerca del punto de interés, se mostrará el número del punto, la pareja ordenada que corresponde, y la serie a la cual pertenece el punto. Al ingresar los datos de tiempo a los cuadros de texto, se ingresa el punto al cual pertenece el tiempo t(i), esto para facilitar la introducción de los datos.

Luego de identificar los puntos de inicio del intervalo de prueba, estos se deben ingresar en los cuadros de texto en la sección *Análisis con Líneas Rectas* (situado debajo del panel gráfico) y presionar el botón *Dibujar Recta*, disponible en la interfaz, estos valores deben ser datos de la prueba, de otra manera se desplegará una ventana de error, pidiéndole que ingrese de nuevo los valores.

Al final, en la interfaz aparecerá la recta de regresión con el intervalo escogido por el usuario, junto con el valor de la pendiente y el factor de regresión.

De esta manera se pueden realizar la evaluación de rectas para evaluar diversos intervalos de la prueba con el fin de identificar regímenes de flujo claves para el análisis.

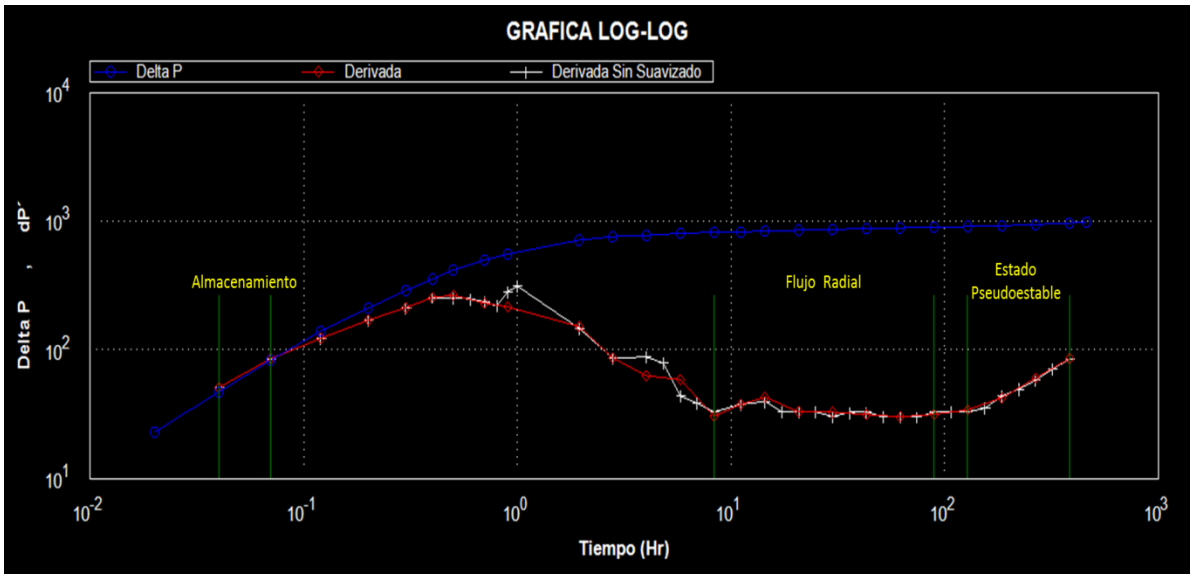
2.4.6.2. IDENTIFICACIÓN DE REGÍMENES DE FLUJO

Con el fin de realizar la identificación de las regiones de flujo en la prueba de presión a analizar se recomienda que esta se realice a través de la gráfica Log-Log, en la cual la derivada permite mostrar con un mayor detalle los regímenes y utilizar la herramienta para rectas de prueba en la Sección 2.4.6.1.

Para definir las regiones de flujo, se diseñaron 6 cuadros de texto agrupados por regiones de flujo (ETR, MTR, LTR). Para realizar los cálculos es esencial definir *por lo menos una región de flujo*; la cual se realiza introduciendo los valores de puntos de tiempo del inicio y del fin del régimen identificado.

La introducción de los valores de los regímenes de flujo es análoga a la forma descrita en la sección anterior. Cuando el usuario desee definir una región en especial, deberá ingresar los valores en los cuadros de texto correspondientes a la región y oprimir el botón *Dibujar Intervalos*, de esta manera, se dibujarán dos líneas verdes verticales por cada región definida, que corresponden al inicio y fin de la región de flujo seleccionada. Las regiones se pueden establecer en cualquiera de las 3 gráficas y automáticamente los intervalos definidos se dibujaran en las otras gráficas.

Figura 46: Grafica Log-Log Con Regímenes de Flujo



2.4.6.3. ANÁLISIS GRAFICA LOG-LOG

El análisis de la gráfica Log-Log también se usa para calcular parámetros relacionados con el comportamiento en tiempos tempranos, tales como el *Almacenamiento*. Si en la prueba que el usuario está analizando, no se definió una región de flujo para tiempos tempranos, omita este numeral. Luego de definir la región de flujo de tiempos tempranos (ETR), debe ingresar un punto correspondiente a un tiempo en la línea de pendiente unitaria (tlpu). Finalmente se procede a dar clic al botón *Cálculos Log-Log* de la gráfica Log-Log, inmediatamente la herramienta arrojará en el panel de resultados, el valor del *Coeficiente de Almacenamiento.*

2.4.6.4. ANÁLISIS GRAFICA SEMILOG

Al realizar el análisis de la gráfica Semilog es posible estimar valores de *Permeabilidad* y D*año* además de otros parámetros, tales como *Relación de Daño,*

Eficiencias de Flujo, etc. Si el usuario no define una región de flujo para tiempos medios, omita este numeral. Luego de realizar los pasos anteriores sólo es necesario oprimir el botón *Cálculos Semilog* de la ventana, inmediatamente la herramienta arrojará en el panel de resultados, una estimación del valor de *Permeabilidad y Daño*, además de otras variables derivadas de interés.

2.4.6.5. ANÁLISIS GRAFICA CARTESIANA

Por medio del análisis con la gráfica cartesiana es posible estimar valores para el *Área de Drenaje y Factor de Forma del Yacimiento*. Si el usuario no definió una región de flujo para tiempos tardíos omita este numeral. El análisis por medio de la gráfica cartesiana consiste en utilizar nuevamente la herramienta de rectas de prueba (Sección 2.4.6.1) para trazar una recta con los valores de la región de tiempos tardíos calcular su pendiente y su corte e introduciros en los cuadros de texto correspondientes a estos valores y luego oprimir el botón *Cálculos Semilog*, después de lo cual, la herramienta arrojara en el panel de resultados, una estimación del valor del *Área de Drenaje y del Factor de Forma*.

2.4.7. ANÁLISIS MODERNO

El enfoque de análisis moderno de esta herramienta, está basado en el uso de una técnica desarrollada en las últimas dos décadas conocida como *Síntesis Directa de Tiab (TDS)*, que consiste en la utilización de la gráfica Log-Log de la derivada y la localización de puntos característicos en la gráfica. Según esta metodología, es posible calcular parámetros de todos los regímenes de flujo con solamente esta gráfica sin la necesidad de usar curvas tipo, además de estimaciones de distancia y Angulo entre barreras.

A continuación se presentara un flujo de trabajo sugerido para un análisis de una prueba de presión por medio del método TDS.

2.4.7.1. RECTAS DE PRUEBA

El procedimiento de trazar rectas de prueba para identificar regiones de flujo es idéntico al descrito en la Sección 2.4.6.1.

2.4.7.2. IDENTIFICACIÓN DE REGÍMENES DE FLUJO

El procedimiento de identificar y definir las regiones de flujo es idéntico al descrito en la Sección 2.4.6.2.

2.4.7.3. IDENTIFICACION DE PUNTOS CARACTERISTICOS

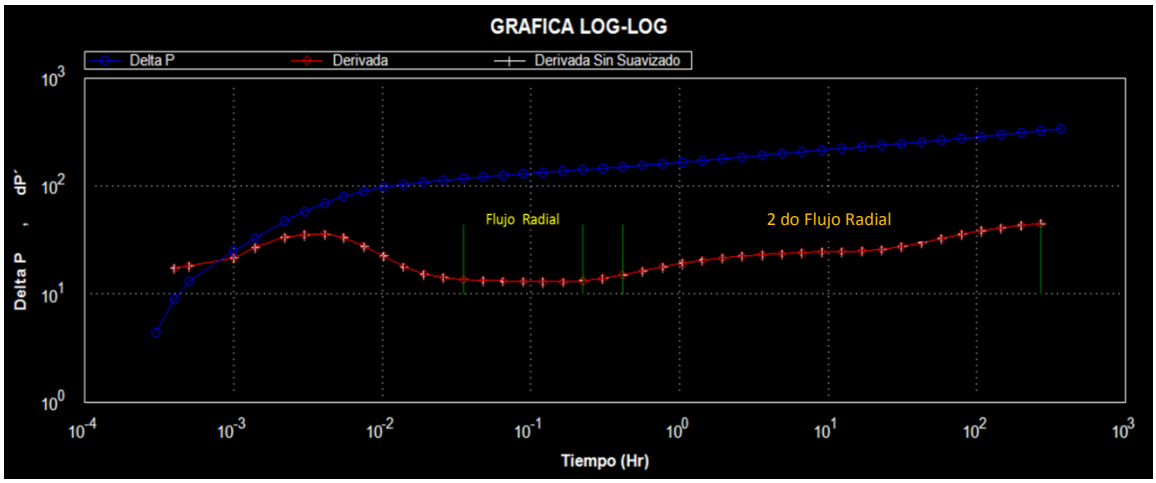
Debido a que la metodología propuesta por Tiab solo emplea la gráfica de la derivada para realizar los análisis, por medio de este método es posible calcular los mismos parámetros obtenidos en el método convencional.

El primer paso en el análisis moderno es la identificación de puntos característicos de la gráfica de la derivada en la gráfica de la derivada, estos puntos se enuncian a continuación.

- **tpu:** Es cualquier punto de tiempo sobre la línea de pendiente unitaria de la región ETR.
- **tpss**: Se al tiempo de inicio del estado pseudoestable.
- **tr**: Se refiere a cualquier punto de tiempo en el flujo radial.
- **tinf**: Hace referencia a un punto de tiempo en el cual haya un cambio de concavidad en la curva en la transición (tiempo de inflexión), el cual se da cuando la onda de perturbación que recorre el yacimiento debido a la prueba, ha atravesado completamente un límite y está a punto de presentar de nuevo flujo radial.

- **t2hl**: Se refiere al punto de tiempo en el cual se inicia la segunda línea horizontal, debido al alcance de un nuevo régimen de flujo radial, debido a que la onda de presión atravesó totalmente el limite encontrado.

Figura 47: Grafica Log-Log Con Segundo Flujo Radial



La definición de estos puntos es optativa para el análisis, si no se definen puntos para el análisis de límites, la herramienta no realizará los cálculos correspondientes; de la misma manera, si el usuario no define un tiempo radial (tr) no se realizaran los cálculos de correspondientes a los parámetros de flujo radial.

Finalmente, definidos los parámetros conocidos se oprime el botón de cálculos, y los resultados serán presentados en la tabla correspondiente a la gráfica Log-Log.

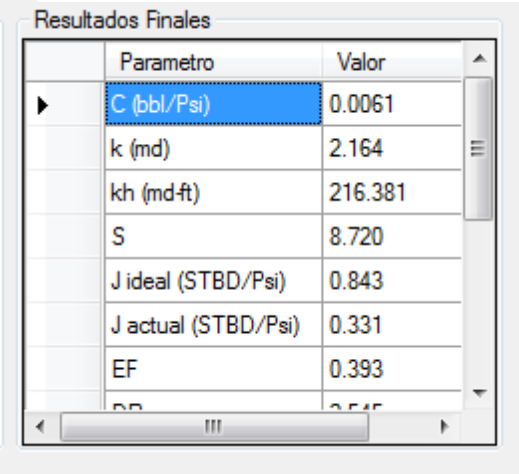
2.5. ANALISIS ESPECIALES

2.5.1. INTRODUCCIÒN

La interfaz del módulo para el análisis de pruebas de caída de presión con una sola tasa (PDD Una Tasa) permite la posibilidad de realizar una serie de análisis

especiales, tales como *Pruebas PDD de Una Para Pozos Parcialmente Cañoneados, Pruebas Pdd de Una Tasa Para Pozos Fracturados Hidráulicamente (Modelo de Fractura Infinita), Pruebas Pdd de Una Tasa Para Yacimientos Naturalmente Fracturados.* Estos análisis especiales se realizan sobre la interfaz de PDD Una Tasa utiliza las mismas graficas del módulo (Cartesiana, Semilog y Log-Log), permitiendo así, estimar propiedades adicionales de cada tipo de análisis en especiales, tales como: Permeabilidad y Daño esféricos (Pozos Parcialmente Cañoneados), Coeficiente Adimensional de Almacenamiento en las Fracturas (Yacimientos Naturalmente Fracturados) y Longitud Media de la Fractura (Pozos Fracturados Hidráulicamente).

Figura 48: Resultados Finales

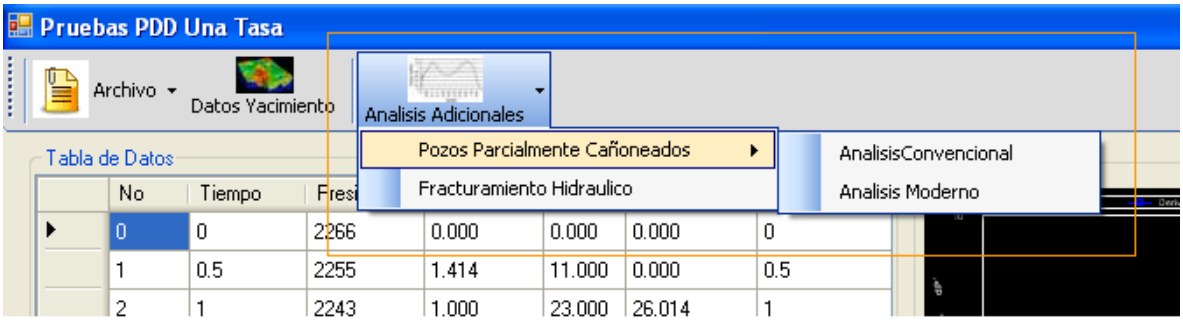


Se accede a los módulos de análisis especiales de Pozos parcialmente cañoneados y fracturados hidráulicamente, a través de un botón en la barra de herramientas de la interfaz PDD Una Tasa. El análisis de Pruebas para yacimientos naturalmente fracturados, se realiza sobre la interfaz de la gráfica log-log.

2.5.1.1. ANALISIS DE POZOS PARCIALMENTE CAÑONEADOS

Con el fin de realizar un análisis para pozos parcialmente cañoneados, se siguieron tanto la metodología convencional y la moderna, cuyos flujos de trabajo se presentaran a continuación.

Figura 49: Botones Análisis Especiales



ANÁLISIS CONVENCIONAL

Uno de los enfoques y objetivos de la herramienta es permitir realizar un análisis de una prueba de presión para pozos parcialmente cañoneados, con la combinación de las gráficas *Cartesiana, Semilog, Log-Log*. Por medio de este análisis es posible hallar parámetros como *Permeabilidad Esférica, Daño Esférico, Permeabilidad Vertical, Radio Equivalente, Daño Por Completamiento y Daño Mecánico*.

1) CALCULOS PREVIOS

Para poder realizar la estimación de parámetros relacionados con los pozos parcialmente cañoneados, se debe hacer una estimación previa del daño y la permeabilidad horizontal, por cualquiera de los dos métodos enunciados en las secciones 2.4.6 o 2.4.7, lo que incluye la lectura de datos y parámetros, trazado de rectas de prueba, definición de regiones de flujo y puntos característicos si es

necesario. Luego de este paso, se debe ejecutar la ventana *Análisis Adicionales/Pozos Parcialmente Cañoneados/Análisis Convencional*.

2) RECTAS DE PRUEBA

Al estar en la ventana de análisis convencionales para pozos parcialmente cañoneados, la cual es una gráfica cartesiana de Pwf Vs el inverso de la raíz del tiempo, se debe realizar el procedimiento de trazar rectas, enunciado en el paso 1 de la Sección 2.4.6.1, caracterizando el flujo esférico que se observaría como una línea recta en esta gráfica.

3) IDENTIFICACION DE PUNTOS CARACTERISTICOS

En la interfaz de análisis convencional de pozos con completamiento parcial, es necesario definir cuatro parámetros para la estimación de las variables de interés en este tipo de análisis:

- **Pendiente (m):** En este cuadro de texto se debe introducir el valor de la pendiente de la recta de prueba generada del paso anterior luego de caracterizar el flujo esférico en esta gráfica.
- **Intercepto(I):** Aquí se introduce el valor del intercepto con el eje Y de la recta de caracterización de flujo.
- **b (h/hp):** Es la relación entre el espesor de la formación y la longitud del intervalo cañoneado.
- **Pi:** Es la estimación de la presión inicial del yacimiento.

4) CALCULO DE PARAMETROS

Luego de haber definido los puntos característicos requeridos para el cálculo, es necesario oprimir el botón de *Cálculos* para que, en el panel de resultados se muestren estimaciones para las variables de interés.

ANALISIS MODERNO

Uno de los enfoques y objetivos de la herramienta es permitir realizar un análisis de una prueba de presión para pozos parcialmente cañoneados, con el uso de la metodología de la *Síntesis Directa de* Tiab, la cual permite realizar estimaciones de Permeabilidad *Esférica, Daño Esférico, Permeabilidad Vertical, Radio Equivalente, Daño Por Completamiento y Daño Mecánico*.

1) CALCULOS PREVIOS

Para poder realizar la estimación de parámetros relacionados con los pozos parcialmente cañoneados, se debe hacer una estimación previa del daño y la permeabilidad horizontal, por cualquiera de los dos métodos enunciados en las secciones 2.4.6 o 2.4.7, lo que incluye la lectura de datos y parámetros, trazado de rectas de prueba, definición de regiones de flujo y puntos característicos si es necesario. Luego de este paso, se debe ejecutar la ventana *Análisis Adicionales/Pozos Parcialmente Cañoneados/Análisis Convencional*.

2) RECTAS DE PRUEBA

Al estar en la ventana de análisis moderno para pozos parcialmente cañoneados, la cual es una gráfica log-log de (t*dp') Vs tiempo, se debe realizar el procedimiento de trazar rectas, enunciado en el Paso 1 de la Sección 2.4.6.1, caracterizando el flujo esférico que se observaría como una línea recta de pendiente de -1/2 en esta gráfica.

3) IDENTIFICACION DE PUNTOS CARACTERISTICOS

En la interfaz de análisis convencional de pozos con completamiento parcial, es necesario definir cuatro parámetros para la estimación de las variables de interés en este tipo de análisis:

- **tsp:** En este cuadro de texto se debe introducir el valor del punto de tiempo en flujo esférico.
- **b (h/hp):** Es la relación entre el espesor de la formación y la longitud del intervalo cañoneado.

4) CALCULO DE PARAMETROS

Luego de haber definido los puntos característicos requeridos para el cálculo, es necesario oprimir el botón de *Cálculos* para que, en el panel de resultados se muestren estimaciones para las variables de interés.

2.5.1.2. ANALISIS YACIMIENTOS NATURALMENTE FRACTURADOS

Con el fin de realizar un análisis para yacimientos naturalmente fracturados, se utilizó la metodología de la *Síntesis Directa de Tiab*. Este análisis se realiza en la misma interfaz de la gráfica Log-Log de PDD Una Tasa.

ANÁLISIS MODERNO

Uno de los enfoques y objetivos de la herramienta es permitir realizar un análisis de una prueba de presión para yacimientos naturalmente fracturados a través de una gráfica *Log-Log*. De esta manera es posible Estimar valores del *Coeficiente de Almacenamiento Adicional de la Fractura y el Coeficiente de Flujo Interporoso.*

1) CALCULOS PREVIOS

Para poder realizar la estimación de parámetros relacionados con los yacimientos naturalmente fracturados, se debe hacer una estimación previa del daño y la permeabilidad horizontal, por medio del método enunciados en la 2.4.7, lo que incluye la lectura de datos y parámetros, trazado de rectas de prueba, definición de regiones de flujo y puntos característicos.

2) IDENTIFICACION DE PUNTOS CARACTERISTICOS

Debido a que el análisis se realiza en la misma interfaz de la gráfica log-log de PDD Una Tasa, solo basta definir un punto característico para que se realicen los cálculos.

- **tmin:** Se debe introducir el valor del punto de tiempo donde la derivada es mínima.

3) CALCULO DE PARAMETROS

Luego de haber definido los puntos característicos requeridos para el cálculo, es necesario oprimir el botón de *Cálculos* para que, en el panel de resultados se muestren estimaciones para las variables de interés.

2.5.1.3. ANALISIS DE POZOS FRACTURADOS HIDRAULICAMENTE

Con el fin de realizar un análisis para pozos fracturados hidráulicamente, se utilizó la metodología de la *Síntesis Directa de Tiab*. Este análisis se realiza en la misma interfaz de la gráfica Log-Log de PDD Una Tasa.

ANÁLISIS MODERNO

Uno de los enfoques y objetivos de la herramienta es permitir realizar un análisis de una prueba de presión para pozos fracturados hidráulicamente a través de una gráfica *Log-Log*. De esta manera es posible Estimar valores *Longitud Media de la Fractura*. La herramienta utiliza el modelo de fractura infinita.

1) CALCULOS PREVIOS

Para poder realizar la estimación de parámetros relacionados con los pozos fracturados hidráulicamente, se debe hacer una estimación previa del daño y la permeabilidad horizontal, por medio del método enunciado en la sección 2.4.7, lo que incluye la lectura de datos y parámetros, trazado de rectas de prueba, definición de regiones de flujo y puntos característicos.

2) RECTAS DE PRUEBA

Al estar en la ventana de análisis moderno para pozos fracturados hidráulicamente, la cual es una gráfica log-log de (t*dp') Vs tiempo, se debe realizar el procedimiento de trazar rectas, en la Sección 2.4.7.1, caracterizando el flujo lineal que se observaría como una línea recta de pendiente de 1/2 en esta gráfica. Al definir la recta de prueba, la herramienta calculara el valor de la derivada a 1 hora sobre la extrapolación de la línea recta de flujo lineal.

3) IDENTIFICACIÓN DE REGÍMENES DE FLUJO

El procedimiento de identificar y definir las regiones de flujo es idéntico al descrito en la Sección 2.4.7.2, pero solo es necesario definir en este caso, el inicio y el final del flujo lineal.

4) IDENTIFICACION DE PUNTOS CARACTERISTICOS

En la interfaz de análisis convencional de pozos fracturados hidráulicamente, es necesario definir dos parámetros para la estimación de las variables de interés en este tipo de análisis:

- **(t*dp')L1:** En este cuadro de texto se debe introducir el valor de la derivada a 1 hora sobre la extrapolación de la línea recta de flujo lineal, luego de

caracterizar el flujo lineal en esta grafica con las rectas de prueba. Este valor será escrito automáticamente en el cuadro de texto, pero el usuario podrá modificarlo según su criterio.

- **tLRi:** Aquí se introduce el valor del punto de tiempo del corte entre la línea de flujo radial con la línea de flujo lineal.

5) CALCULO DE PARAMETROS

Luego de haber definido los puntos característicos requeridos para el cálculo, es necesario oprimir el botón de *Cálculos* para que, en el panel de resultados se muestren estimaciones para las variables de interés.

2.6. INTERFAZ MODULO PDD PARA MULTIPLES TASAS

2.6.1. INTRODUCCIÒN

La interfaz del módulo para el análisis de Pruebas de Caída de Presión (PDD) para múltiples tasas, permite la carga de los datos a través de un archivo de texto plano (*.txt), permite la visualización de los mismos en formato de tabla y en gráficas cartesiana, semilog y log-log. A través de esta, se pueden desplegar graficas individuales, sobre las cuales se puede dibujar rectas, calcular pendientes y factores de regresión, definir regiones de flujo (ETR, MTR y LTR), introducir valores característicos de las curvas y realizar cálculos de las propiedades físicas de interés (k y S).

2.6.2. LECTURA DE DATOS

La forma de ingresar los datos de la prueba se debe realizar a través de un archivo de texto plano (*.txt), en el cual se ingresan los valores en 3 columnas sin encabezado y separadas por el símbolo (;). La primera columna corresponder al tiempo, la segunda a la presión de fondo medida al tiempo correspondiente y la tercera es la tasa de flujo correspondiente al tiempo y a la presión medida, pero se

debe tener cuidado de incluir una fila 0, que tiene, la presión inicial, a tiempo 0 y con tasa q_0:

Tabla 10: Conjunto de datos para una prueba PDD MultiTasa

0	;	4412	;	0
0.105	;	4332	;	180
0.151	;	4302	;	177
0.217	;	4264	;	174
0.313	;	4216	;	172
0.45	;	4160	;	169
0.648	;	4099	;	166
0.934	;	4039	;	163
1.34	;	3987	;	161
1.94	;	3952	;	158
2.79	;	3933	;	155
4.01	;	3926	;	152
5.78	;	3926	;	150
8.32	;	3927	;	147
9.99	;	3928	;	145
14.4	;	3931	;	143
20.7	;	3934	;	140
29.8	;	3937	;	137
43	;	3941	;	134
61.8	;	3944	;	132
74.2	;	3946	;	130
89.1	;	3948	;	129
107	;	3950	;	127
128	;	3952	;	126
154	;	3954	;	125

185	;	3956	;	123

La forma de cargar los datos se realiza con el botón *Archivo/Cargar Data*, luego de haber creado el archivo con los datos de la prueba. Luego, la interfaz le mostrará una ventana de cuadro de diálogo, en la cual el usuario *deberá* buscar el archivo previamente preparado con los datos de la prueba de presión y seleccionarlo para abrirlo.

Finalmente la interfaz PDD MultiTasa se mostrará nuevamente con los datos cargados en una tabla, junto con los datos calculados y las gráficas.

Para los datos de una prueba de caída de presión con múltiples tasas el software no aplica algoritmo de suavizado a los datos, debido a que el filtro puede eliminar datos relevantes.

2.6.3. VISUALIZACIÓN DE DATOS

Luego de la lectura de los datos de la prueba de presión (Sección 2.6.2), la interfaz principal del módulo PDD MultiTasa permite la visualización de una tabla con los datos de la prueba y con otros parámetros calculados para el análisis de la prueba además de mostrar de manera simultánea de las gráficas cartesiana, semilog y log-log; aunque también es posible ver cada grafica en una ventana independiente para su análisis, por medio de un botón en la parte inferior de cada grafica en la ventana principal. Cabe aclarar que el tiempo equivalente t_{eq} es función de la sumatoria, no está relacionado con el tiempo equivalente usado en prueba de ascenso de presión (PBU).

2.6.4. LECTURA DE PARÁMETROS ADICIONALES

Para realizar un análisis de la prueba de presión cargada a través del archivo de texto plano, es necesario introducir parámetros de la prueba, pozo y del yacimiento:

- Viscosidad del Fluido (cP).
- Factor Volumétrico de Formación del Aceite (Rb/STB).
- Espesor de la Formación (ft).
- Porosidad (Fracción).
- Compresibilidad Total (Psi^{-1}).
- Radio del Wellbore (ft).

Luego de introducir los datos requeridos, se oprime el botón de cargar datos, la herramienta se encargara de realizar una validación de los datos, que consiste en evaluar las cantidades introducidas como valores y verificara los datos mediante un algoritmo de excepciones expuesto previamente en el módulo de PDD una tasa.

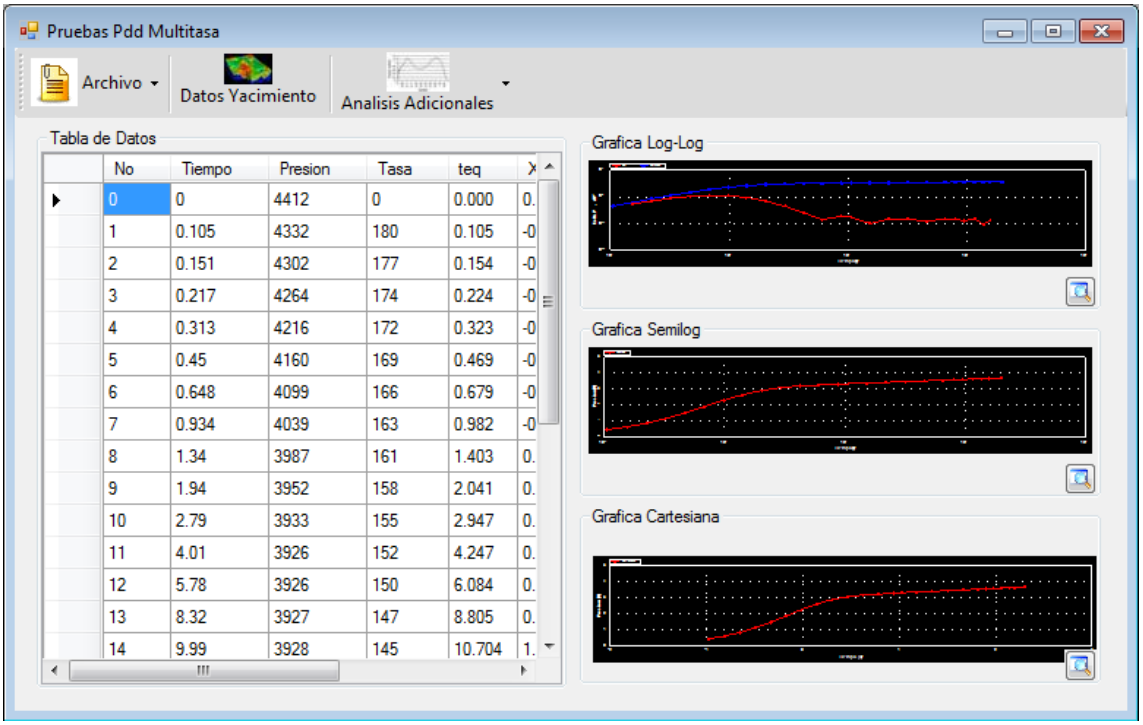
2.6.5. FLUJO DE TRABAJO PARA EL ANÁLISIS DE LA PRUEBA DE CAIDA DE PRESIÒN CON MULTIPLES TASAS

Luego de haber realizado la carga de datos de la prueba a través del archivo de texto plano (Sección 2.6.2) y de la introducción de los parámetros adicionales (Sección 2.6.4), el siguiente paso es el análisis de la prueba de presión. Con el fin de llevar a cabo este objetivo, se propone una serie de flujos de trabajo a seguir con esta herramienta, uno para análisis convencional y otro para el análisis moderno.

2.6.5.1. ANÁLISIS CONVENCIONAL

Uno de los enfoques y objetivos de la herramienta es permitir realizar un análisis de una prueba de presión de manera convencional, con la combinación de las gráficas *Cartesiana, Semilog, Log-Log*. Por medio de este análisis es posible hallar parámetros como Coeficiente de Almacenamiento, *Daño, Permeabilidad* y demás parámetros de interés.

Figura 50: Interfaz Modulo PDD MultiTasa



No	Tiempo	Presion	Tasa	teq
0	0	4412	0	0.000
1	0.105	4332	180	0.105
2	0.151	4302	177	0.154
3	0.217	4264	174	0.224
4	0.313	4216	172	0.323
5	0.45	4160	169	0.469
6	0.648	4099	166	0.679
7	0.934	4039	163	0.982
8	1.34	3987	161	1.403
9	1.94	3952	158	2.041
10	2.79	3933	155	2.947
11	4.01	3926	152	4.247
12	5.78	3926	150	6.084
13	8.32	3927	147	8.805
14	9.99	3928	145	10.704

1) RECTAS DE PRUEBA

La metodología para realizar las rectas prueba es análoga a la expuesta en la Sección 2.4.6.1.

2) IDENTIFICACIÓN DE REGÍMENES DE FLUJO

El procedimiento de identificar y definir las regiones de flujo es igual al descrito en la Sección 2.4.6.2.

3) ANÁLISIS GRÀFICA LOG-LOG

El análisis con la gráfica Log-Log se realiza inicialmente para definir los regímenes de flujo, pero en este caso se usa para calcular parámetros relacionados con el comportamiento en tiempos tempranos, tales como el *Almacenamiento*. Si en la prueba que el usuario está analizando, no se definió una región de flujo para tiempos tempranos, omita este numeral. Luego que se ha definido la región de flujo de tiempos tempranos (ETR), se ingresa un punto correspondiente a un tiempo en la línea de pendiente unitaria (tlpu). Finalmente se procede a presionar el botón *Cálculos Log-Log* de la ventana de la gráfica Log-Log, después de lo cual la herramienta arrojará en el panel de resultados, una estimación del valor del *Coeficiente de Almacenamiento.*

4) ANÁLISIS GRÀFICA SEMILOG

Por medio del análisis con la gráfica semilog es posible estimar valores de *Permeabilidad* y D*año*, junto con los demás parámetros derivados, tales como *Relación de Daño, Eficiencias de Flujo, etc*. Si en la prueba que el usuario está analizando, no se definió una región de flujo para tiempos medios, omita este numeral. El análisis por medio de la gráfica semilog consiste en oprimir el botón *Cálculos Semilog* de la ventana, después de lo cual, la herramienta arrojara en el panel de resultados, una estimación del valor de *Permeabilidad y Daño*, además de otras variables derivadas de interés.

5) ANÁLISIS GRÀFICA CARTESIANA

Por medio del análisis con la gráfica cartesiana es posible estimar valores de *Permeabilidad* y D*año*, junto con los demás parámetros derivados, tales como *Relación de Daño, Eficiencias de Flujo, etc*. Si en la prueba que el usuario está analizando, no se definió una región de flujo para tiempos medios, omita este numeral. . En análisis por medio de la gráfica cartesiana consiste en oprimir el botón *Cálculos* de la ventana, después de lo cual, la herramienta arrojara en el panel de resultados, una estimación del valor *Permeabilidad y Daño*, además de otras variables derivadas de interés.

2.6.5.2. ANÁLISIS MODERNO

A continuación se presentará un flujo de trabajo sugerido para un análisis de una prueba de caída de presión de múltiples tasas, por medio del método TDS.

1) ANALISIS CON RECTAS

El procedimiento de trazar rectas de prueba para identificar regiones de flujo es igual al descrito en la Sección 2.4.6.1.

2) IDENTIFICACIÓN DE REGÍMENES DE FLUJO

El procedimiento de identificar y definir las regiones de flujo es igual al descrito en la Sección 2.4.6.2.

3) IDENTIFICACION DE PUNTOS CARACTERISTICOS

Debido a que la metodología propuesta por Tiab utiliza solo la gráfica de la derivada para realizar los análisis, aquí se presentarán los resultados de todos los parámetros obtenidos en el método convencional en una sola gráfica.

El primer paso en el análisis moderno es la identificación de puntos característicos de la gráfica de la derivada en la gráfica de la derivada, estos puntos se enuncian a continuación.

- **tpu:** Es cualquier punto de tiempo sobre la línea de pendiente unitaria de la región ETR.
- **tr**: Se refiere a cualquier punto de tiempo en el flujo radial.

Finalmente, definidos los parámetros conocidos se oprime el botón de cálculos, y los resultados serán presentados en la tabla correspondiente a la gráfica Log-Log.

2.7. INTERFAZ MODULO PBU PRECEDIDO POR UNA TASA CONSTANTE

2.7.1. INTRODUCCION

La interfaz del módulo para el análisis de Pruebas de Restauración de Presión (PBU) para Una Tasa, permite la carga de los datos a través de un archivo de texto plano (*.txt), permite la visualización de los mismos en formato de tabla y en graficas de Horner y Log-Log. A través de esta, se pueden desplegar graficas individuales, sobre las cuales se puede dibujar rectas, calcular pendientes y factores de regresión, definir regiones de flujo (ETR, MTR y LTR), introducir valores característicos de las curvas y realizar cálculos de las propiedades físicas de interés (k y S) junto con estimaciones de la presión promedio del yacimiento.

2.7.2. LECTURA DE PARÁMETROS DE POZO Y YACIMIENTO

Para realizar un análisis de la prueba es necesario introducir parámetros de la prueba y del yacimiento que son adicionales antes de cargar los datos de la prueba, debido a que los valores que se van a graficar, dependen de parámetros de lectura previos tales como:

- Viscosidad del Fluido (cP).
- Factor Volumétrico de Formación del Aceite (Rb/STB).
- Espesor de la Formación (ft).
- Porosidad (Fracción).
- Compresibilidad Total (Psi^{-1}).
- Radio del Wellbore (ft).
- Ultima Tasa Antes del Cierre (STB).
- Np o Petróleo Producido Acumulado (STB).

Luego de introducir los datos requeridos, se oprime el botón de cargar datos, la herramienta se encargara de realizar una validación de los datos, que consiste en evaluar las cantidades introducidas como valores representativos, tales como porosidades positivas, menores que 0.42, etc.

2.7.3. LECTURA DE DATOS

La forma de ingresar los datos de la prueba de presión se debe realizar a través de un archivo de texto plano (*.txt), en el cual se ingresan los valores en 2 columnas sin encabezado y separadas por el símbolo (;). La primera columna debe corresponder al tiempo, la segunda a la presión de fondo medida al tiempo, pero se debe tener cuidado de incluir una fila, que sería la primera, con la presión inicial, a tiempo 0 y con tasa 0:

Tabla 11: Conjunto de datos para una prueba PBU Una Tasa

0	;	2840
1.1	;	3170
1.6	;	3199
2.5	;	3240
3.5	;	3278
5	;	3290
7	;	3302
9	;	3310
13	;	3320
20	;	3333
30	;	3343
40	;	3350
50	;	3363
70	;	3382
100	;	3400
150	;	3423
250	;	3450

La forma de cargar los datos se realiza a través del botón *Archivo/Cargar Data*, luego de haber creado el archivo con los datos de la prueba. Posteriormente, observará una ventana que le solicitará la elección de un factor de suavizado (el cual deberá estar entre 0 y 0.5), por medio del cual se eliminarán puntos que generen “ruido” en la gráfica de la derivada, para más información, remítase a *Smoothing* en el manual de referencia. Un factor recomendado es de 0.2, pero si desea no realizar suavizado, digite un valor de 0.

Luego, la interfaz le mostrará una ventana de cuadro de diálogo, en la cual el usuario *deberá* buscar el archivo previamente preparado con los datos de la

prueba de presión y seleccionarlo para abrirlo. Finalmente la interfaz PBU Una Tasa se mostrará nuevamente con los datos cargados en una tabla, junto con los datos calculados y las gráficas de los mismos.

2.7.4. VISUALIZACIÓN DE DATOS

Luego de la lectura de los datos de la prueba de presión (Sección 2.7.3), la interfaz principal del módulo PBU Una Tasa permite la visualización de una tabla con los datos de la prueba y con otros parámetros calculados para el análisis de la prueba además de mostrar de manera simultánea de las gráficas de Horner y Log-Log; aunque también es posible ver cada gráfica en una ventana independiente para su análisis, por medio de un botón en la parte inferior de cada gráfica en la ventana principal.

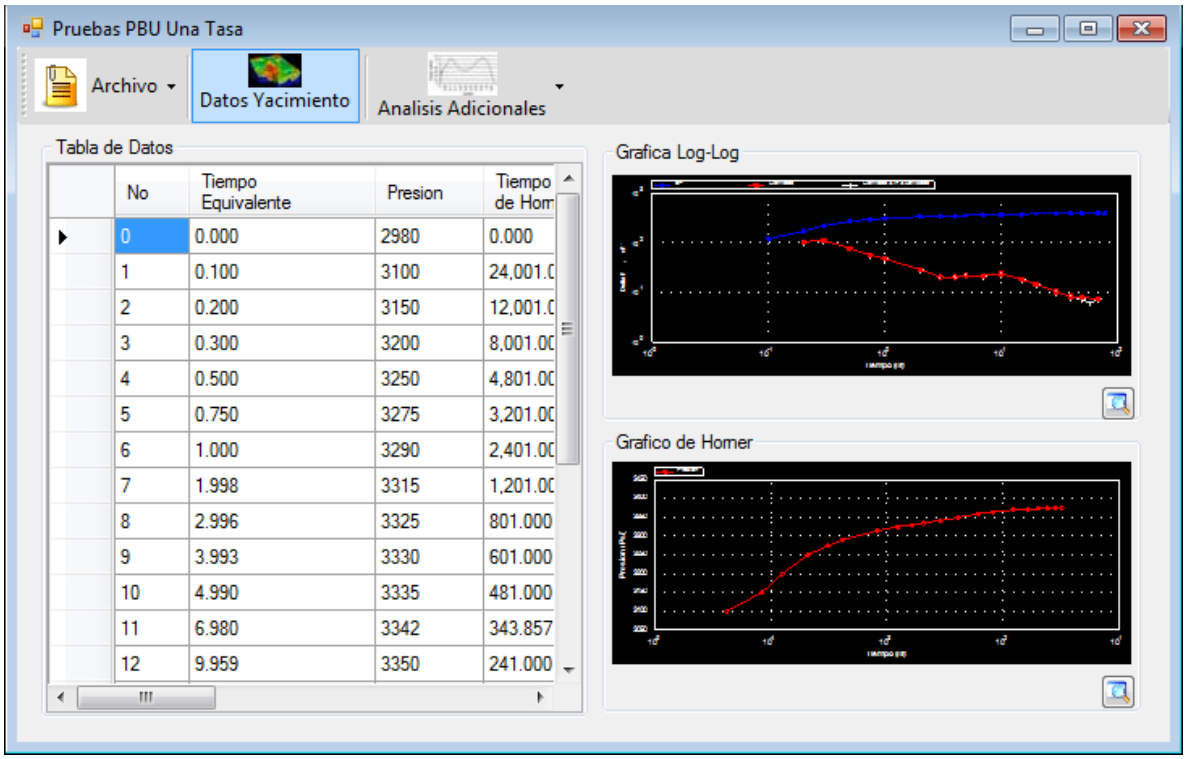
2.7.5. FLUJO DE TRABAJO PARA EL ANÁLISIS DE LA PRUEBA DE RESTAURACIÒN DE PRESIÒN PRECEDIDA POR UNA TASA CONSTANTE

Luego de haber realizado la carga de datos de la prueba a través del archivo de texto plano (Sección 2.7.3) y de la introducción de los parámetros adicionales (Sección 2.7.2), el siguiente paso es el análisis de la prueba de presión. Con el fin de llevar a cabo este objetivo, se propone una serie de flujos de trabajo a seguir con esta herramienta, uno para análisis convencional y otro para el análisis moderno.

2.7.5.1. ANÁLISIS CONVENCIONAL

Uno de los enfoques y objetivos de la herramienta es permitir realizar un análisis de una prueba de presión de manera convencional, con la combinación de las gráficas *de Horner, Log-Log.* Por medio de este análisis es posible hallar parámetros como Coeficiente de Almacenamiento, *Daño, Permeabilidad, Presión Promedio del Yacimiento* y demás parámetros de interés.

Figura 51: Interfaz PBU Una Tasa



1) RECTAS DE PRUEBA

La metodología para realizar las rectas prueba es análoga a la expuesta en la Sección 2.4.6.1.

2) IDENTIFICACIÓN DE REGÍMENES DE FLUJO

El procedimiento de identificar y definir las regiones de flujo es igual al descrito en la Sección 2.4.6.2.

3) ANÁLISIS GRAFICA LOG-LOG

El análisis con la gráfica Log-Log se realiza inicialmente para definir los regímenes de flujo, pero en este caso se usa para calcular parámetros relacionados con el

comportamiento en tiempos tempranos, tales como el *Almacenamiento*. Si en la prueba que el usuario está analizando, no se definió una región de flujo para tiempos tempranos, omita este numeral. Luego que se ha definido la región de flujo de tiempos temprano (ETR), debe ingresar un punto correspondiente a un tiempo en la línea de pendiente unitaria (tlpu). Finalmente procede a presionar el botón *Cálculos Log-Log* de la ventana de la gráfica Log-Log, después de lo cual la herramienta arrojara en el panel de resultados, una estimación del valor del *Coeficiente de Almacenamiento.* Para el cálculo de la derivada se realiza empleando el tiempo equivalente, ver manual de referencia.

4) ANÁLISIS GRAFICA DE HORNER

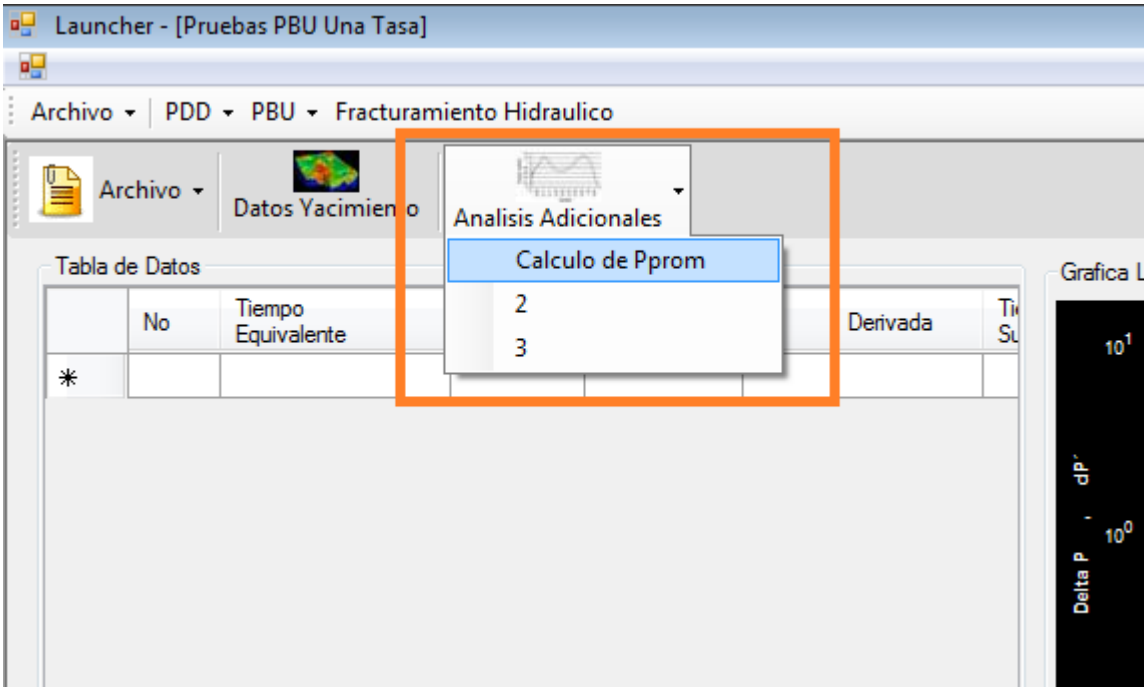
Por medio del análisis con la gráfica de Horner es posible estimar valores de *Permeabilidad*, D*año y Conductividad.* Si en la prueba que el usuario está analizando, no se definió una región de flujo para tiempos medios, omita este numeral. En análisis por medio de la gráfica semilog consiste en oprimir el botón *Cálculos Horner* de la ventana, después de lo cual, la herramienta arrojara en el panel de resultados, una estimación del valor de *Permeabilidad, Daño y Conductividad*.

5) ANÁLISIS GRAFICA MBH

Por medio del análisis con la gráfica de MBH es posible estimar valores de *Presión Promedio del Yacimiento, Índice de Productividad y otros parámetros de interés.* Si en la prueba que el usuario está analizando, no se definió una región de flujo para tiempos medios, omita este numeral.

Antes de realizar los cálculos de presión promedio con este método, es necesario que haya ya estimado un valor de la permeabilidad por medio del análisis convencional o el moderno.

Figura 52: Botones de Acceso a la Ventana de Cálculo de Pprom



La interfaz para el cálculo de la presión promedio se encuentra en la barra de tareas de la interfaz principal PBU Una Tasa, en el botón que dice *Análisis Adicionales/Calculo Pprom*.

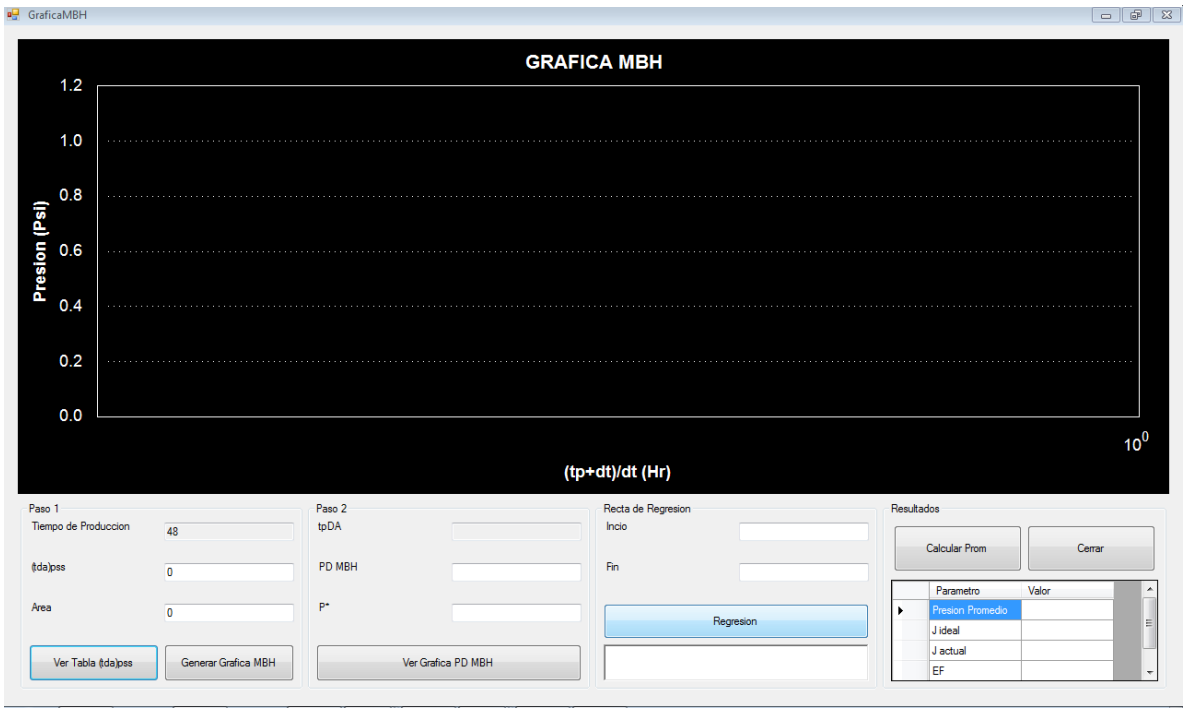
De esta manera se desplegará la interfaz para el cálculo de la presión promedio. De la cual se deben seguir una serie de pasos:

1. Debe conocer la forma aproximada del yacimiento, ya que deberá pulsar el botón de *Ver Tabla (tda)pss* para estimar un valor de *(tda)pss* según la forma de su yacimiento.
2. Conocido el *(tda)pss* y el Área aproximada (en Acres), pulse el botón *Generar Grafica MBH*, para que esta se genere y le arroje también un valor de tpDA en el cuadro de texto correspondiente.
3. Según su criterio, realice una regresión sobre los puntos que considere una recta, use la herramienta de la recta de regresión que le ofrece valores de

factor de regresión para escoger la definitiva. Cuando lo haya decidido, aparecerá una estimación de P^* sobre el cuadro de texto, cuyo valor usted puede modificar a su criterio o mantenerlo.

4. Ahora, con el valor de *tpDA* que se le proporcionó en el cuadro de texto, oprima el botón *Ver Grafica* P_{DMBH} para obtener una estimación de P_{DMBH}.
5. Finalmente, con el valor obtenido de P* y con el valor estimado por usted de PD MBH, oprima el botón *Calcular Pprom*, y en la tabla de resultados se mostrara una estimación de la *Presión Promedio del Yacimiento y otros parámetros de interés*.

Figura 53: Interfaz Gráfica MBH



2.7.5.2. ANÁLISIS MODERNO

A continuación se presentará un flujo de trabajo sugerido para un análisis de una prueba de restauración de presión de una tasa, por medio del método TDS.

1) RECTAS DE PRUEBA

El procedimiento de trazar rectas de prueba para identificar regiones de flujo es igual al descrito en la Sección 2.4.6.1.

2) IDENTIFICACIÓN DE REGÍMENES DE FLUJO

El procedimiento de identificar y definir las regiones de flujo es igual al descrito en la Sección 2.4.6.2.

3) IDENTIFICACIÒN DE PUNTOS CARACTERÌSTICOS

Debido a que la metodología propuesta por Tiab utiliza solo la gráfica de la derivada para realizar los análisis, aquí se presentarán los resultados de todos los parámetros obtenidos en el método convencional en una sola gráfica.

El primer paso en el análisis moderno es la identificación de puntos característicos de la gráfica de la derivada, estos puntos se enuncian a continuación.

- **tpu:** Es cualquier punto de tiempo sobre la línea de pendiente unitaria de la región ETR.
- **tr**: Se refiere a cualquier punto de tiempo en el flujo radial.
- **tinf**: Hace referencia a un punto de tiempo en el cual haya un cambio de concavidad en la curva (tiempo de inflexión), el cual se da cuando la onda de perturbación que recorre el yacimiento debido a la prueba, ha atravesado completamente un límite y está a punto de presentar de nuevo flujo radial infinito.

Finalmente, definidos los parámetros conocidos se oprime el botón de cálculos, y los resultados serán presentados en la tabla correspondiente a la gráfica Log-Log.

2.8. INTERFAZ MODULO PBU PRECEDIDO POR VARIAS TASAS

2.8.1. INTRODUCCION

La interfaz del módulo para el análisis de Pruebas de Restauración de Presión (PBU) precedido por múltiples tasa, permite la carga de los datos a través de un archivo de texto plano (*.txt), permite la visualización de los mismos en formato de tabla y en una gráfica cartesiana. A través de esta, se pueden desplegar gráficas individuales, sobre las cuales se puede dibujar rectas, calcular pendientes y factores de regresión, definir regiones de flujo (ETR, MTR y LTR), introducir valores característicos de las curvas y realizar cálculos de las propiedades físicas de interés (k y S).

2.8.2. LECTURA DE PARÁMETROS ADICIONALES

Para realizar un análisis de la prueba es necesario introducir parámetros de la prueba y del yacimiento que son adicionales antes de cargar los datos de la prueba, debido a que los valores que se van a graficar, dependen de parámetros de lectura previos tales como:

- Viscosidad del Fluido (cP).
- Factor Volumétrico de Formación del Aceite (Rb/STB).
- Espesor de la Formación (ft).
- Porosidad (Fracción).
- Compresibilidad Total (Psi^{-1}).
- Radio del Wellbore (ft).

Luego de introducir los datos requeridos, se oprime el botón de cargar datos, la herramienta se encargará de realizar una validación de los datos, que consiste en

evaluar las cantidades introducidas como valores representativos, tales como porosidades positivas, menores que 0.42, etc.

2.8.3. LECTURA DE DATOS

La forma de ingresar los datos de la prueba de presión se debe realizar a través de dos archivos de texto plano (*.txt), en el primero, debe incluir la producción histórica del pozo que se va a probar, este archivo debe contener 2 columnas sin encabezado y separadas por el símbolo (;). La primera columna debe corresponder al tiempo, la segunda a tasa medida en ese tiempo, pero se debe tener cuidado de incluir una fila, que sería la primera, con la presión inicial, a tiempo 0.

Tabla 12: Conjunto de datos de producción histórica para una prueba PBU MultiTasa

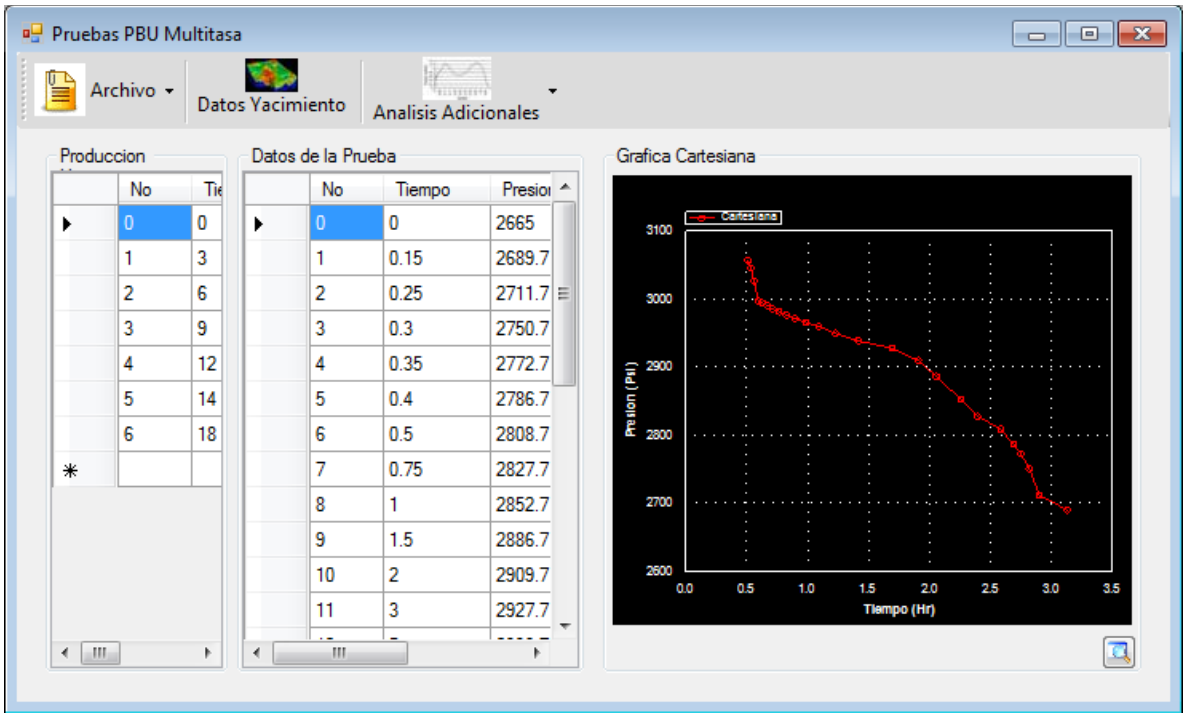
0	;	0
3	;	479
6	;	319
9	;	160
12	;	125
14	;	100
18	;	75

El segundo archivo se ingresan los valores en 2 columnas sin encabezado y separadas por el símbolo (;). La primera columna debe corresponder al tiempo, la segunda a la presión de fondo medida al tiempo, pero se debe tener cuidado de incluir una fila, que sería la primera, con la presión P_{wf} $\Delta t=0$, a tiempo 0:

Tabla 13: Conjunto de datos para una prueba PBU MultiTasa

0	;	2665
0.15	;	2689.7
0.25	;	2711.7
0.3	;	2750.7
0.35	;	2772.7
0.4	;	2786.7
0.5	;	2808.7
0.75	;	2827.7
1	;	2852.7
1.5	;	2886.7
2	;	2909.7
3	;	2927.7
5	;	2938.7
7	;	2949.7
9	;	2959.7
11	;	2965.7
13	;	2971.7
15	;	2976.7
17	;	2981.7
19	;	2985.7
21	;	2990.7
23	;	2994.7
25	;	2997.7
27	;	3026.7
29	;	3045.7
31	;	3056.7

Figura 54: Interfaz Módulo PBU MultiTasa



Se debe cargar primero la producción histórica y luego los datos de la prueba, debido a que se necesitan los primeros para generar parámetros a graficar.

La forma de cargar los datos se realiza a través del botón *Archivo/Cargar Producción Histórica y Archivo/Cargar Data*, luego de haber creado los archivo con los datos de producción y los de la prueba. Luego, la interfaz le mostrará una ventana de cuadro de diálogo, en la cual el usuario *deberá* buscar el archivo previamente preparado con los datos de la prueba de presión y seleccionarlo para abrirlo.

Finalmente la interfaz PBU MultiTasa se mostrará nuevamente con los datos cargados en una tabla, junto con los datos calculados y las gráficas de los mismos.

Para los datos de una prueba de restauración de presión con múltiples tasas no se le aplicó algoritmo de suavizado a los datos, debido a que el filtro puede eliminar datos relevantes, tales como los cambios de tasa.

2.8.4. VISUALIZACIÓN DE DATOS

Luego de la lectura de los datos de la prueba de presión (Sección 1.8.3), la interfaz principal del módulo PBU MultiTasa permite la visualización de una tabla con los datos de la prueba y con otros parámetros calculados para el análisis de la prueba además de mostrar una gráfica cartesiana.

2.8.5. FLUJO DE TRABAJO PARA EL ANÁLISIS DE LA PRUEBA DE RESTAURACION DE PRESION PARA MULTIPLES TASAS

Luego de haber realizado la carga de datos de la prueba a través de los archivo de texto plano (Sección 2.8.3) y de la introducción de los parámetros adicionales (Sección 2.8.2), el siguiente paso es el análisis de la prueba de presión. Con el fin de llevar a cabo este objetivo, se propone un flujo de trabajo a seguir.

2.8.5.1. ANÁLISIS CONVENCIONAL

Uno de los enfoques y objetivos de la herramienta es permitir realizar un análisis de una prueba de presión de manera convencional con ayuda de una gráfica cartesiana. Por medio de este análisis es posible hallar parámetros como Coeficiente de Almacenamiento, *Daño, Permeabilidad y Conductividad.*

1) RECTAS DE PRUEBA

La metodología para realizar las rectas prueba es análoga a la expuesta en la Sección 2.4.6.1.

2) IDENTIFICACIÓN DE REGÍMENES DE FLUJO

El procedimiento de identificar y definir las regiones de flujo es igual al descrito en la Sección 2.4.6.2.

2.8.5.2. ANÁLISIS GRAFICA CARTESIANA

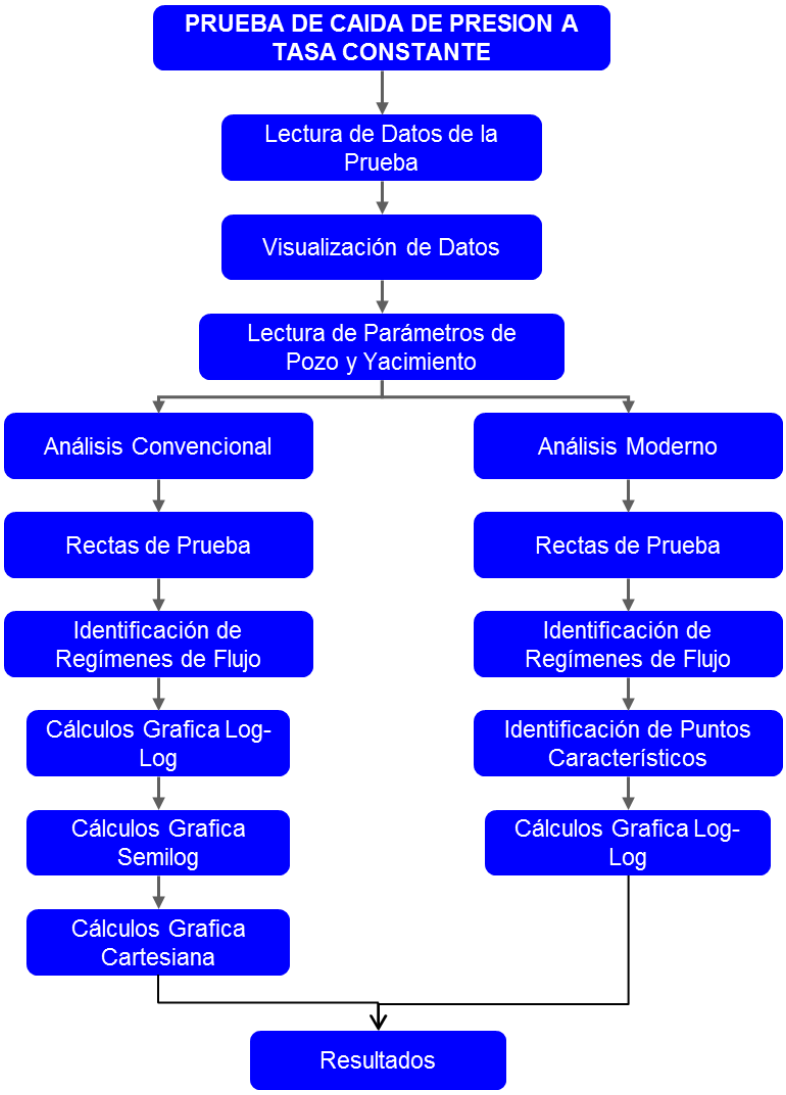
Por medio del análisis con la gráfica cartesiana graficando P_{ws} vs Función de Superposición X_n es posible estimar valores de *Permeabilidad*, D*año y Conductividad*. Si en la prueba que el usuario está analizando, no se definió una región de flujo para tiempos medios omita este numeral. . En análisis por medio de la gráfica cartesiana consiste en oprimir el botón *Cálculos Semilog* de la ventana, después de lo cual, la herramienta arrojara en el panel de resultados, una estimación del valor *Permeabilidad, Daño y Conductividad*.

2.9. DIAGRAMAS DE FLUJO DE TRABAJO

Se diseñaron una serie de diagramas de trabajo que resumen en forma de flujo lo enunciado en el manual de usuario, con el fin de dar un vistazo rápido a la manera de realizar los análisis de pruebas de presión para cada interfaz.

2.9.1. INTERFAZ PDD UNA TASA

Figura 55: Flujo de Trabajo Interfaz PDD Una Tasa



Pasos Detallados:

- ✓ Launcher:
  - Hacer click en el botón del módulo Pdd Una Tasa.
- ✓ Interfaz Pdd Una Tasa

- Dar Click en el botón Datos de Yacimiento sobre la interfaz Pdd Una Tasa.
- Ingresar los valores de Yacimiento, Pozo y Fluido en la ventana emergente.
- Presionar el botón Archivo/Cargar Data e ingresar el valor de factor de suavizado para los datos de la prueba.
- En la ventana emergente, buscar el archivo con los datos de la prueba y cargarlo.
- Hacer click en el botón con una Lupa, debajo de la vista previa de la Grafica Log-Log.

✓ Grafica Log-Log

- Introducir los valores del número del punto (que se puede visualizar al acercar el cursor al punto en observación) en los cuadros de texto correspondientes al cuadro de Rectas de Prueba, con el fin de calcular pendientes, cortes y factores de correlación de la serie de puntos escogida; y oprimir el botón Recta.
- Introducir los valores del número del punto correspondientes al inicio y fin de las regiones de flujo ETR, MTR y LTR en los cuadros de texto correspondientes; y oprimir el botón Intervalos.
- Definir puntos característicos en el cuadro Puntos Adicionales según su análisis.
- Oprimir el botón Calcular y visualizar los resultados en las tablas.

✓ Grafica Semilog

- Oprimir el botón Calcular y visualizar los resultados.

✓ Grafica Cartesiana

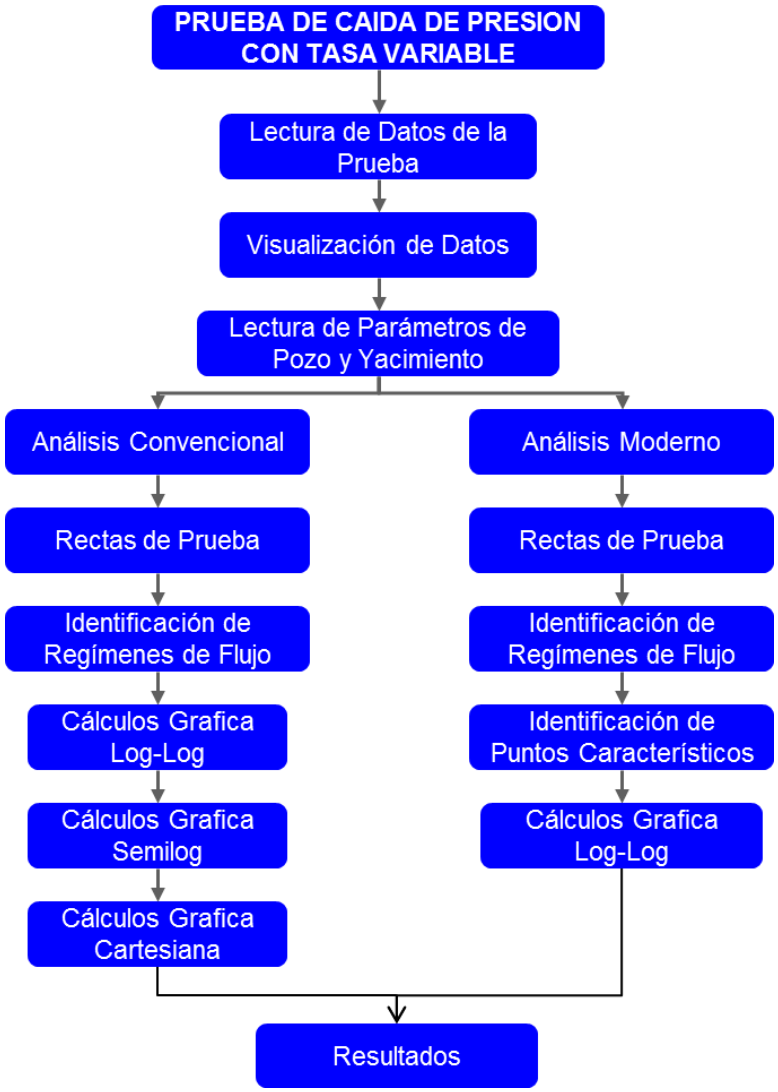
- Introducir los valores del número del punto del inicio y final de la recta correspondiente al estado pseudoestable y presionar el botón recta.
- En el cuadro de texto llamado Resultados de la Recta de Prueba copiar los valores de la pendiente y el intercepto y pegarlos en los cuadros de

texto correspondientes a los Puntos Adicionales de m* y Pint respectivamente.

- Oprimir el botón Calcular y visualizar los resultados.

2.9.2. INTERFAZ PDD MULTITASA

Figura 56: Flujo de Trabajo Interfaz PDD MultiTasa



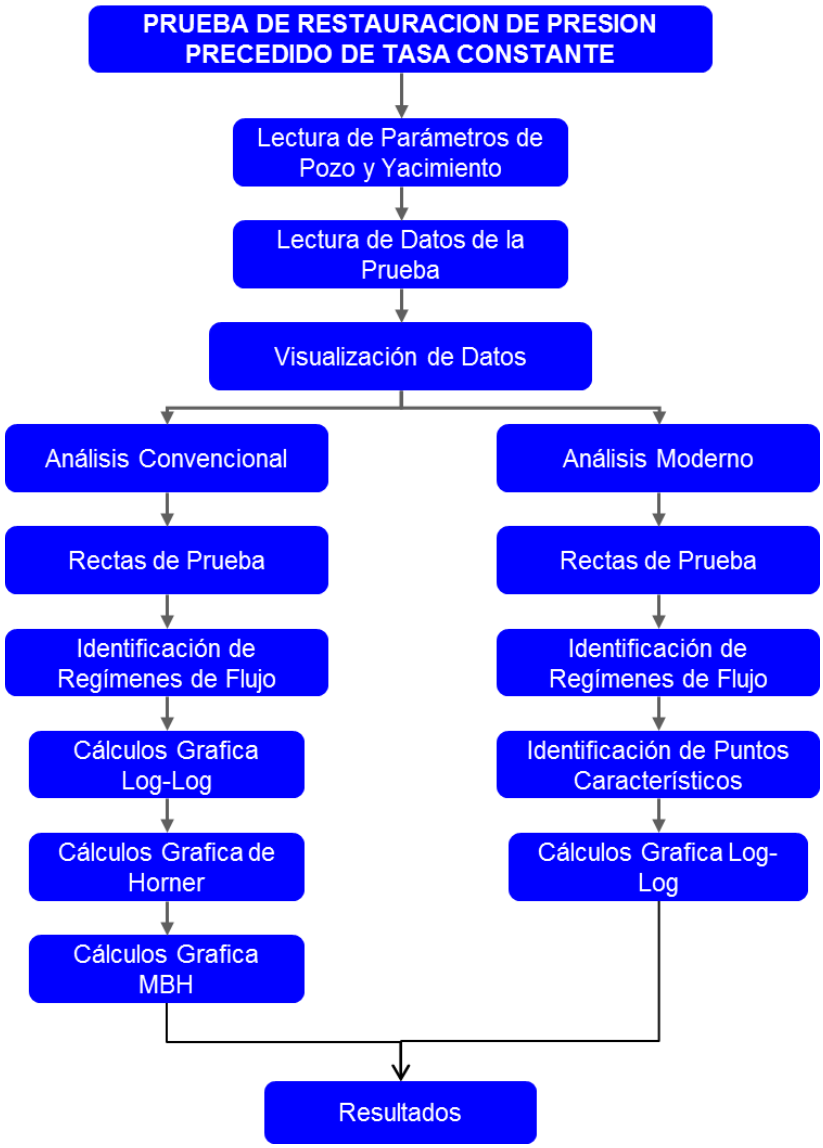
Pasos Detallados:

- ✓ Launcher:
  - Hacer click en el botón del módulo Pdd MultiTasa.
- ✓ Interfaz Pdd MultiTasa
  - Dar Click en el botón Datos de Yacimiento sobre la interfaz Pdd MultiTasa.
  - Ingresar los valores de Yacimiento, Pozo y Fluido en la ventana emergente.
  - Presionar el botón Archivo/Cargar Data y en la ventana emergente, buscar el archivo con los datos de la prueba y cargarlo.
  - Hacer click en el botón con una Lupa, debajo de la vista previa de la Grafica Log-Log.
- ✓ Grafica Log-Log
  - Introducir los valores del número del punto (que se puede visualizar al acercar el cursor al punto en observación) en los cuadros de texto correspondientes al cuadro de Rectas de Prueba, con el fin de calcular pendientes, cortes y factores de correlación de la serie de puntos escogida; y oprimir el botón Recta.
  - Introducir los valores del número del punto correspondientes al inicio y fin de las regiones de flujo ETR, MTR y LTR en los cuadros de texto correspondientes; y oprimir el botón Intervalos.
  - Definir puntos característicos en el cuadro Puntos Adicionales según su análisis.
  - Oprimir el botón Calcular y visualizar los resultados en las tablas.
- ✓ Grafica Semilog
  - Oprimir el botón Calcular y visualizar los resultados.
- ✓ Grafica Cartesiana
  - Introducir los valores del número del punto del inicio y final de la recta correspondiente al estado pseudoestable y presionar el botón recta.

- En el cuadro de texto llamado Resultados de la Recta de Prueba copiar los valores de la pendiente y el intercepto y pegarlos en los cuadros de texto correspondientes a los Puntos Adicionales de m* y Pint respectivamente.
- Oprimir el botón Calcular y visualizar los resultados.

2.9.3. INTERFAZ PBU UNA TASA

Figura 57: Flujo de Trabajo Interfaz PBU Una Tasa



Pasos Detallados:

- ✓ Launcher:
  - Hacer click en el botón del módulo interfaz Pbu Una Tasa.
- ✓ Interfaz Pbu Una Tasa
  - Dar Click en el botón Datos de Yacimiento sobre la interfaz Pbu Una Tasa.
  - Ingresar los valores de Yacimiento, Pozo y Fluido en la ventana emergente.
  - Presionar el botón Archivo/Cargar Data y en la ventana emergente, buscar el archivo con los datos de la prueba y cargarlo.
  - Hacer click en el botón con una Lupa, debajo de la vista previa de la Grafica Log-Log.
- ✓ Grafica Log-Log
  - Introducir los valores del número del punto (que se puede visualizar al acercar el cursor al punto en observación) en los cuadros de texto correspondientes al cuadro de Rectas de Prueba, con el fin de calcular pendientes, cortes y factores de correlación de la serie de puntos escogida; y oprimir el botón Recta.
  - Introducir los valores del número del punto correspondientes al inicio y fin de las regiones de flujo ETR, MTR y LTR en los cuadros de texto correspondientes; y oprimir el botón Intervalos.
  - Definir puntos característicos en el cuadro Puntos Adicionales según su análisis.
  - Oprimir el botón Calcular y visualizar los resultados en las tablas.
- ✓ Grafica de Horner
  - Introducir los valores del número del punto del inicio y final de la recta correspondiente a los tiempos tardíos y presionar el botón recta.
  - Observar en punto de inflexión en tiempos de Horner pequeños (hacia la izquierda) e introducir el número del punto en el cuadro de texto correspondiente.

- Oprimir el botón Calcular y visualizar los resultados.

2.9.4. INTERFAZ PBU MULTITASA

Figura 58: Flujo de Trabajo Interfaz PBU MultiTasa

- ✓ Launcher:
  - Hacer click en el botón del módulo Pbu MultiTasa.
- ✓ Interfaz Pbu MultiTasa
  - Dar Click en el botón Datos de Yacimiento sobre la interfaz Pdd Una Tasa.

- Ingresar los valores de Yacimiento, Pozo y Fluido en la ventana emergente.
- Presionar el botón Archivo/Cargar Producción Histórica y en la ventana emergente, buscar el archivo con los datos de la prueba y cargarlo.
- Presionar el botón Archivo/Cargar Data y en la ventana emergente, buscar el archivo con los datos de la prueba y cargarlo.
- Hacer click en el botón con una Lupa, debajo de la vista previa de la Grafica Cartesiana.

✓ Grafica Cartesiana

- Introducir los valores del número del punto (que se puede visualizar al acercar el cursor al punto en observación) en los cuadros de texto correspondientes al cuadro de Rectas de Prueba, con el fin de calcular pendientes, cortes y factores de correlación de la serie de puntos escogida; y oprimir el botón Recta.
- Introducir los valores del número del punto correspondientes al inicio y fin de las regiones de flujo ETR, MTR y LTR en los cuadros de texto correspondientes; y oprimir el botón Intervalos.
- Oprimir el botón Calcular y visualizar los resultados.

ANEXO C: DERIVADA Y ALGORITMO DE SUAVIZADO

DERIVADA LOGARITMICA

Calcular la derivada de presión requiere cuidado, ya que el proceso de diferenciación de los datos amplifica el ruido que podría estar presente. Una diferenciación numérica usando los puntos adyacentes producirá una derivada muy ruidosa.

Si los datos están distribuidos en una progresión geométrica (con la diferencia del tiempo convirtiéndose cada vez más grande de un punto a otro), entonces el ruido en la derivada puede ser reducido en algún grado usando una diferenciación numérica de la siguiente manera:

$$t\left(\frac{dp}{dt}\right)_i = \left(\frac{dp}{dlnt}\right)_i$$

$$\left(\frac{dp}{dlnt}\right)_i = \frac{\ln\left(\frac{t_i}{t_{i-1}}\right)\Delta p_{i+1}}{\ln\left(\frac{t_{i+1}}{t_i}\right)\ln\left(\frac{t_{i+1}}{t_{i-1}}\right)} + \frac{\ln\left(\frac{t_{i+1}t_{i-1}}{{t_i}^2}\right)\Delta p_i}{\ln\left(\frac{t_{i+1}}{t_i}\right)\ln\left(\frac{t_i}{t_{i-1}}\right)} - \frac{\ln\left(\frac{t_{i+1}}{t_i}\right)\Delta p_{i-1}}{\ln\left(\frac{t_i}{t_{i-1}}\right)\ln\left(\frac{t_{i+1}}{t_{i-1}}\right)}$$

Ecuación 59

ALGORITMO DE SUAVIZADO

A pesar de que derivada logarítmica reduce el ruido en el cálculo de la misma, incluso este enfoque es insuficiente y lleva a una derivada ruidosa. El mejor método para reducir el ruido es usar puntos del set de datos que estén separados como mínimo 0.2 ciclos logarítmicos, en comparación con puntos que están adyacentes inmediatamente. Entonces:

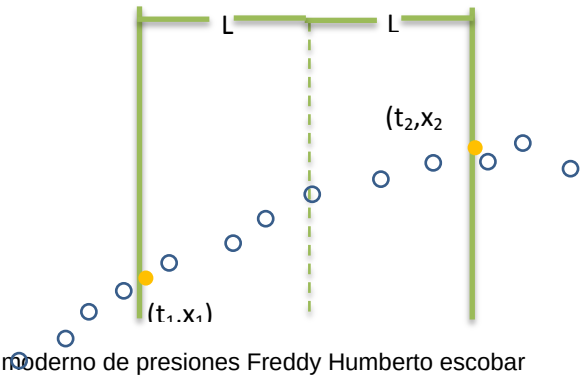
$$t\left(\frac{dp}{dt}\right)_i = \left(\frac{dp}{dlnt}\right)_i$$

$$\left(\frac{dp}{dlnt}\right)_i = \frac{\ln\left(\frac{t_i}{t_{i-1}}\right)\Delta p_{i+1}}{\ln\left(\frac{t_{i+1}}{t_i}\right)\ln\left(\frac{t_{i+1}}{t_{i-1}}\right)} + \frac{\ln\left(\frac{t_{i+1}t_{i-1}}{{t_i}^2}\right)\Delta p_i}{\ln\left(\frac{t_{i+1}}{t_i}\right)\ln\left(\frac{t_i}{t_{i-1}}\right)} - \frac{\ln\left(\frac{t_{i+1}}{t_i}\right)\Delta p_{i-1}}{\ln\left(\frac{t_i}{t_{i-1}}\right)\ln\left(\frac{t_{i+1}}{t_{i-1}}\right)}$$

$$lnt_{i+j} - lnt_i \geq 0.2$$

$$lnt_i - lnt_{i-k} \geq 0.2$$

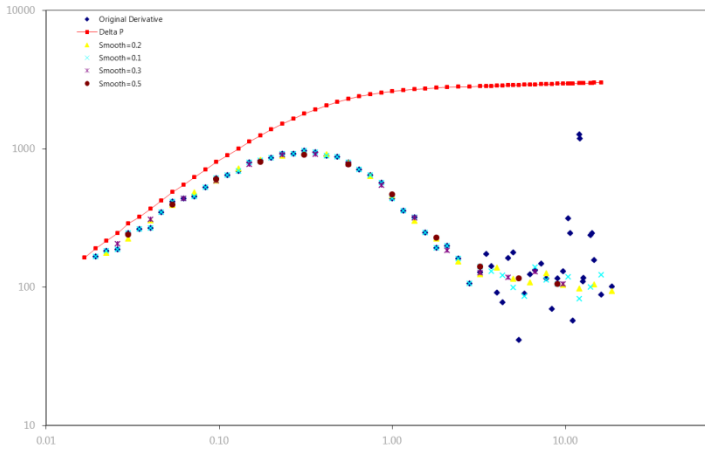
Figura 59: Ilustración del Proceso de Suavizado



Fuente: Editado de análisis moderno de presiones Freddy Humberto escobar

El valor de 0.2 (Conocido como el intervalo de diferenciación) puede ser reemplazado por una valor mayor o menor, este software ofrece la facilidad de escoger un rango que va entre (0 – 0.5)1 con diferentes consecuencias en el suavizado. Tenga en cuenta que si el intervalo es muy alto la forma de la derivada puede ser distorsionada, si no desea realizar suavizado use un valor de 0.

Figura 60: Sensibilidad usando diferentes valores de factor de suavizado



Fuente: autores

ANEXO D: UNIDADES Y NOMENCLATURA

El sistema de unidades usado por el software es Unidades de Campo el más extendido en la industria petrolera.

ENTRADA DE DATOS INPUT

Parámetros requeridos para la entrada de datos y sus correspondientes unidades.

q	Caudal o tasa de Producción	(STB/D)
t	tiempo de la Prueba	(Hr)
P	Presión de la Prueba	(Psi)
B	Factor Volumétrico de Formación	(bbl_{res}/ STB_{std})
H	Espesor de la formación	(ft)
μ	Viscosidad	(Cp)
Φ	Porosidad	(Adim.)
r_w	radio Del Wellbore	(ft)
Ct	Compresibilidad total	(Psi^{-1})
A	Área	(Acres)

SALIDA DE DATOS OUTPUT

C	Coeficiente de Almacenamiento	(STB/Psi)
K	Permeabilidad	(md)
K_{sp}	Permeabilidad Esférica	(md)

S	Daño	(Adim.)
S_{sp}	Daño esférico	(Adim.)
S_c	Daño por Completamiento Parcial	(Adim.)
C_A	Factor de Forma	(Adim.)
J	Índice de Productividad	(Bbl/Psi)
EF	Eficiencia de Flujo	(Adim.)
DR	Relación de Daño	(Adim.)
d_c	Distancia de la Falla más cercana	(ft)
d_f	Distancia de la Falla más lejana	(ft)
θ= Angulo entre fallas		(º)
x_f	Longitud media de la fractura	(ft)
ω	Coeficiente adimensional de almacenamiento en la fractura	(Adim.)
λ	Coeficiente de flujo interporoso	(Adim.)

CONVENCIONES

P_{wf}	Presión de fondo Fluyendo
P_{ws}	Presión de cierre
P_i	Presión Inicial
Lpu	Línea de Pendiente Unitaria
P_{wf1hr}	Presión de fondo fluyendo Extrapolada a 1 hora
P*	Presión extrapolada del gráfico de Horner que corta con el eje y.
ΔP_s	Caída de presión debida al skin
V_p	Volumen Poroso

P_{int}	Presión intersección
R_{weff}	Radio Efectivo del Pozo
$(t*\Delta P')$	Derivada
$(t*\Delta P')_r$	Derivada en Flujo radial
$(t*\Delta P')_i$	Derivada en intersección entre LPU y flujo radial
$(t*\Delta P')_{2HL}$	Derivada en la línea horizontal final
$(t*\Delta P')_{sp}$	Derivada en flujo esférico pendiente característica -1/2
$(t*\Delta P')_L$	Derivada en flujo lineal pendiente característica 1/2
$(t*\Delta P')_x$	Valor de la Derivada en la joroba
t_{inf}	Punto de inflexión en la transición entre la curva de acción infinita y la primera línea horizontal.
t_{2HLs}	Tiempo de inicio de la segunda Línea Horizontal
t_{pss}	Tiempo de estado Pseudo Estable (Pseudo Steady State)