

**UNIVERSIDAD DE SANTIAGO DE CHILE
FACULTAD DE INGENIERÍA
DEPARTAMENTO DE INGENIERÍA METALURGIA**



**DESARROLLO DE UN PROGRAMA CÓDIGO ABIERTO PARA LA
FUSIÓN DE CONCENTRADO EN EL CONVERTIDOR TENIENTE**

RODOLFO ALEJANDRO BERRÍOS ARCE

Profesor Guía: M.Cs. René Bustamante Moreno

Trabajo de Titulación presentado en conformidad
a los requisitos para obtener el Título de
Ingeniero de Ejecución en Metalurgia.

Santiago – Chile
2010

© RODOLFO ALEJANDRO BERRIOS ARCE

Se autoriza la reproducción parcial o total de esta obra, con fines académicos, por cualquier forma, medio o procedimiento, siempre y cuando se incluya la cita bibliográfica del documento.

AGRADECIMIENTOS

Quiero expresar los más sinceros agradecimientos al señor René Bustamante Moreno, quien con su conocimiento y experiencia me guio en este trabajo de titulación y así pude desarrollarlo de la mejor forma posible. Del mismo modo quiero agradecer al resto de los docentes del Departamento de Ingeniería Metalúrgica ya que contribuyeron enormemente en mi formación académica. Para todos ellos mis saludos y afectos.

TABLA DE CONTENIDOS

RESUMEN	XI
----------------------	----

CAPÍTULO I: INTRODUCCIÓN

1.1 Antecedentes generales.....	12
1.2 El Convertidor Teniente.....	13
1.3 Software y soluciones informáticas.....	15
1.3.1 Software propietario.....	16
1.3.2 Software Libre.....	18
1.3.3 Software Código Abierto.....	19
1.3.4 Ventajas y desventajas del Código Abierto.....	22
1.3.5 El Código Abierto en las empresas.....	25
1.3.6 Soluciones aplicables a los procesos pirometalúrgicos.....	27
1.4 Objetivos.....	29
1.5 Alcances y limitaciones.....	29

CAPÍTULO II: ANTECEDENTES TEORICOS

2.1 El proceso de fusión de concentrado de cobre.....	30
2.1.1 Productos del proceso.....	30
2.1.2 Efecto del fundente.....	32
2.1.3 Efecto del oxígeno.....	34
2.2 El proceso Teniente.....	35

2.2.1	Flujos de entrada.....	35
2.2.2	Flujos de salida.....	36
2.2.3	Descripción del proceso.....	37
2.3	Principales variables y parámetros del Convertidor Teniente.....	38
2.4	Balance de masa.....	39
2.5	Ajuste de balance de masa.....	41
2.6	Clasificación de programas informáticos.....	42
2.6.1	Programas de escritorio y programas de lado del servidor...	44
2.7	Modelo cliente servidor.....	47
2.8	Tipos de programación.....	48
2.9	Base de datos.....	50
2.9.1	Características de las bases de datos.....	51
2.9.2	Modelo de base de datos relacional.....	52

CAPÍTULO III: DESARROLLO DEL TEMA

3.1	Cálculo del balance de masa.....	53
3.1.1	Cálculo de la cantidad de Eje, Escoria y Fundente.....	53
3.1.2	Cálculo del coeficiente de oxígeno.....	56
3.1.3	Cálculo del oxígeno técnico.....	56
3.1.4	Cálculo del aire de proceso.....	57
3.2	Cálculo de los parámetros de ajuste.....	57
3.2.1	Parámetro característico para el fundente.....	58

3.2.2	Parámetro característico para el oxígeno.....	58
3.3	Planificación del programa.....	59
3.3.1	Características principales y específicas del programa.....	60
3.3.2	Elección de las tecnologías a emplear.....	60
3.3.3	Back-end y Front-end del programa.....	64
3.4	Desarrollo de la interfaz gráfica.....	68
3.4.1	Programas empleados.....	68
3.4.2	Secciones.....	69
3.4.3	Diseño del menú principal.....	69
3.4.4	Diseño de las notificaciones y alertas.....	70
3.4.5	Validación de la interfaz.....	72
3.5	Escritura del código fuente.....	72
3.6	Modelamiento de la base de datos.....	73
3.7	Ingreso de información y validación de datos.....	75
3.8	Comprobación de los resultados obtenidos.....	76
3.9	Recursos externos empleados.....	76

CAPÍTULO IV: RESULTADOS

4.1	Archivos de programa.....	79
4.2	Base de datos.....	80
4.3	Requerimientos para instalar y usar el programa.....	81
4.4	Distribución del programa.....	82

4.5 Capturas de pantalla.....	83
4.6 Esquema de funcionamiento.....	87
4.7 Tiempo y costo de desarrollo.....	88
CAPÍTULO V: DISCUSIONES.....	89
CAPÍTULO VI: CONCLUSIONES.....	95
CAPÍTULO VII: REFERENCIA BIBLIOGRAFICA.....	97
APÉNDICE	
8.1 APÉNDICE A: Cálculo de la cantidad de Eje, Escoria y Fundente..	99
8.1.1 Composición del Eje.....	99
8.1.2 Composición de la Escoria.....	101
8.2 APÉNDICE B: Balance de azufre y oxígeno.....	104
MATERIAL COMPLEMENTARIO	
Programa desarrollado.....	Material en CD-ROM
Manual de usuario.....	Material en CD-ROM

ÍNDICE DE TABLAS

Tablas del Capítulo 1

1.1	Típicos programas aplicables en procesos metalúrgicos.....	17
1.2	Principales proyectos código abierto.....	23
1.3	Programas (scripts) para tiendas en línea.....	26

Tablas del Capítulo 3

3.1	Índice TIOBE para agosto de 2010.....	62
3.2	Índice TIOBE histórico para agosto (2005-2010).....	63
3.3	Valor de los parámetros límite por defecto del programa.....	75

Tablas del Capítulo 4

4.1	Tipos de archivo del programa.....	79
4.2	Comparación de tamaños entre versiones del programa.....	82
4.3	Tiempo empleado en cada etapa de desarrollo.....	88

Tablas del Capítulo 8

8.1	Elementos que forman parte del Eje.....	99
8.2	Compuestos que forman parte del Eje.....	99
8.3	Elementos que forman parte de la Escoria.....	101
8.4	Compuestos que forman parte de la Escoria.....	101
8.5	Balance de azufre y oxígeno.....	104

ÍNDICE DE ILUSTRACIONES

Figuras del Capítulo 2

2.1	Equilibrio líquido-líquido para el sistema FeO—FeS—SiO ₂ a 1200°C.....	32
2.2	Sistema FeO—Fe ₂ O ₃ —SiO ₂ a 1200 y 1250°C.....	33
2.3	Diagrama de flujos de un proceso Teniente.....	35
2.4	Flujos másicos del Convertidor Teniente.....	41
2.5	Esquema simple del modelo cliente-servidor.....	46
2.6	Esquema resumido del modelo cliente-servidor.....	48

Figuras del Capítulo 3

3.1	Estructura de pestañas del menú principal.....	70
3.2	(a) Alertas por datos inválidos (b) Demostración del aviso de rango esperado por parámetro.....	71
3.3	Estructura del aviso de acción.....	71

Figuras del Capítulo 4

4.1	Estructura de las tablas “operaciones” y “operación_nueva”.....	80
4.2	Estructura de la tabla “parámetros_límite”.....	80
4.3	Vista completa de la pantalla principal.....	83
4.4	Captura de la sección “Diagrama y tablas”.....	84
4.5	Vista parcial de la sección “Carga actual”.....	84

4.6	Vista parcial principal del “Historial”.....	85
4.7	Vista de la sub-sección “Archivo completo” de la sección “Historial”.....	85
4.8	Captura parcial de sección “Parámetros límite”.....	85
4.9	Vista de mensaje de navegador incompatible.....	86
4.10	Vista de mensaje de JavaScript no habilitado.....	86
4.11	Esquema del funcionamiento del programa.....	87

RESUMEN

El presente trabajo de titulación pretende dar a conocer el concepto de desarrollo de software conocido como código abierto, lo cual se realizó mediante el desarrollo de un programa que calcula los parámetros de operación del Convertidor Teniente en la operación de fusión de concentrado de cobre.

Este desarrollo implicó la selección del conjunto de tecnologías informáticas que determinan el desarrollo de un programa, incluyendo el lenguaje informático, el sistema de base de datos y el tipo de programación empleado. Adicionalmente se tomaron en cuenta las limitaciones de las soluciones actuales y se buscó otorgar valor agregado al programa desarrollado mediante validaciones de datos y una simple e intuitiva interfaz de usuario.

La selección condujo al empleo de la programación de lado de servidor ya que además de ser más económica, ofrece características multiplataforma y de aplicación en red sin implicar mayores complicaciones.

Se obtuvo como resultado general que el desarrollo bajo el concepto de código abierto está apto para satisfacer este tipo de necesidades y que, mediante el conjunto de tecnologías informáticas empleadas, se logran ahorros en el costo de desarrollo de hasta el orden de un 37% respecto de otros lenguajes informáticos y el mismo concepto de desarrollo de software.

CAPÍTULO I: INTRODUCCIÓN

1.1. Antecedentes generales

El cobre es un elemento metálico que puede ser obtenido a partir de minerales sulfurados o bien de minerales oxidados, donde según el tipo de asociaciones que tenga el mineral, su tratamiento tomará una de las dos vías de obtención de cobre. Para los minerales de cobre asociados con oxígeno la obtención se realiza por la vía de la hidrometalurgia, es decir, reacciones químicas en soluciones acuosas. Para los minerales de cobre sulfurados, la obtención será por la vía de la pirometalurgia, la cual corresponde a procesos a elevadas temperaturas.

La abundancia de minerales sulfurados ha implicado que la pirometalurgia tenga un rol más que trascendental en la producción del mismo. De la producción total de cobre en el mundo, cerca de un 90% se obtiene por la vía de la pirometalurgia. La obtención de cobre metálico por esta vía se realiza en tres etapas:

1. Fusión a eje o mata.
2. Conversión del eje a cobre blíster.
3. Refinación a fuego.

El proceso de fusión consiste en la obtención una mata de cobre líquida a partir del concentrado sólido, el cual está compuesto por especies sulfuradas de cobre y hierro. En el proceso de fusión, se inyecta oxígeno para que reaccione con las especies mineralógicas del concentrado y además, se ingresa fundente (SiO_2) para que se forme escoria Fayalita ($2\text{FeO}\cdot\text{SiO}_2$). La fase óxido se denomina escoria y contiene FeO , Fe_3O_4 , $2\text{FeO}\cdot\text{SiO}_2$, Al_2O_3 , etc y la fase sulfuro se denomina Metal Blanco o Eje y contiene principalmente Cu_2S y FeS , ambas fases tienen distinta densidad y son separadas aprovechando esta

propiedad. El eje es posteriormente transformado a cobre blíster (98,5% Cu) en un horno de conversión y luego, en los hornos de refinación a fuego, el blíster se transforma en cobre anódico (99,5% Cu), el cual es llevado a electrorefinación para la obtención de cátodos de cobre (99,99% Cu).

1.2. El Convertidor Teniente

El Convertidor Teniente es un horno basculante de fusión en baño de concentrado desarrollado y patentado por la División El Teniente de CODELCO. Fue creado debido al fracaso de la tecnología *Oxygen Smelting*, que consiste en fundir concentrado por balance de calor y arrastre de concentrado en los gases. La idea principal bajo del desarrollo del Convertidor Teniente fue emplear el calor generado por oxidación de la carga para fundir concentrado en un convertidor convencional modificado.

Se originó en la década de 1970 en fundición Caletones producto de la investigación de Hermann Schwarze, quien con su equipo experimentó con un convertidor pequeño (2 m de diámetro por 3 m de largo) que había sido retirado de la fundición Chagres. En 1975 producto de la reparación de un convertidor Peirce-Smith, Schwarze (en esos tiempos superintendente general de la fundición Caletones) modifica la longitud del reactor y comienza a utilizarlo para fundir concentrado de cobre. Dado que la fundición trabajaba con una cantidad elevada de oxígeno, pudieron fundir una mayor cantidad de concentrado.

Según Hermann Shwarze^[1], este convertidor modificado (denominado “tarro” por él mismo) no fue creado con la finalidad de su comercialización sino que para solucionar específicamente los problemas de Caletones. Producto del éxito de esta tecnología, la cual involucraba menor uso de equipos y de

¹ http://www.sonami.cl/cgi-bin/procesa.pl?plantilla=/boletinmensual_detalle.html&id_art=78

combustible (coincidente con la crisis del petróleo de 1973), muchas fundiciones del mundo comenzaron a interesarse en la tecnología que estaba utilizando El Teniente, en ese entonces sociedad mixta El Teniente, conformada por el Estado de Chile y la norteamericana Kennecott Corporation.

En 1978 Chuquicamata manifiesta su interés en el convertidor modificado y le solicita a Shwarze el diseño de un reactor para esta división. Dado que el nombre informal de “tarro” no podía seguir siendo empleado para referirse a esta tecnología, Shwarze decide bautizar su reactor como Convertidor Modificado Tipo Teniente. Fue así como desde finales de los 70 el uso del Convertidor Teniente se extendió a lo largo de nuestro país. Actualmente existen 6 Convertidores Teniente operando en Chile (Chuquicamata, Paipote, Potrerillos, Ventanas y dos en Caletones), con una capacidad total combinada de 4.3 MM ton/año de concentrado, produciendo 1.2 MM ton/año de cobre^[2].

La tecnología Teniente de fundición ha sido comercializada por CODELCO en 3 convertidores en el resto mundo occidental ^[3], los que operan en NKANA (Zambia), ILO (Perú) y La Caridad (México). Existe además un Convertidor Teniente en Tailandia el cual está operando desde el año 1996^[4].

Una de las particularidades de este reactor consiste en que mantiene la posibilidad de llegar a operación autógena, es decir, sin requerir energía externa. Esto lo puede lograr con el enriquecimiento de oxígeno dependiendo del tipo de concentrado.

El Convertidor Teniente tiene el gran mérito de aumentar el rendimiento y además ahorrar energía, lo cual es doblemente importante ya que economizar es siempre una importante prioridad, especialmente en Chile por no ser un gran

² Carlos Caballero Deramond, “Chilean Copper Smelting and Refining Overview”, 2009

³ Álvaro González Letelier, “Riquezas Minerales de Chile a Nivel Mundial”, 2000.

⁴ http://www.bnamericas.com/news/mining/Teniente_Converter_Sells_For_US*15_MN

productor de petróleo, lo que significa ahorro de divisas. En el proceso de fusión tradicional, el petróleo es un ítem importante en el costo global, lo que se elimina en parte en esta etapa, que se conoce como “Proceso Teniente”.

1.3. Software y soluciones informáticas

Desde el momento en que las soluciones informáticas comenzaron a ayudar al hombre en las más diversas actividades, siempre ha existido un desarrollo constante de nuevas soluciones que buscan satisfacer las más variadas necesidades como editar una fotografía o controlar un proceso industrial. Soluciones que estrictamente se conocen como programas informáticos, los cuales corresponden a un conjunto de instrucciones, interpretadas en un computador y con el fin de lograr una tarea específica. De acuerdo a la función que cumplen estos programas, se los clasifica por software de sistema y software de aplicación, estos últimos corresponden a lo que comúnmente conocemos como programas, por ejemplo un procesador de texto como Microsoft Word.

Es indudable que cada vez tenemos a nuestra disposición las más variadas alternativas de programas para una tarea específica, esto se debe a que los programas se van adaptando a los constantes cambios por ejemplo, la disponibilidad de computadores con mayores capacidades de procesamiento, lo cual permite realizar tareas o poseer características que en un pasado no tan lejano solo vivían en la imaginación. Gracias a esto, hemos visto como cada vez hay mejores programas con características más amigables, con un carácter más específico y sobretodo confiables. Obviamente que esto, desde un punto de vista técnico, ha significado la creación y desarrollo de nuevos lenguajes de programación, modelos de desarrollo y entornos. Desarrollo que no tan sólo abarca temas puramente técnicos sino que también temas de carácter ético

como el respeto de las libertades de los usuarios sobre el producto adquirido, lo cual ha implicado una fuerte distinción entre tipos de programas.

En el concierto actual se puede definir un programa de acuerdo a diversas características que lo componen, como por ejemplo el lenguaje en que está escrito, plataformas en las que puede desempeñarse (sistemas operativos), licencia, etc. Lo cual puede hacer engorroso esta segmentación, sin embargo, la clasificación de acuerdo al modelo de desarrollo nos permite identificar claramente tres tipos de software, los que corresponden al Software propietario, Software libre y finalmente Software código abierto.

1.3.1. Software propietario

Se define como software propietario a aquel programa informático donde el usuario tiene el derecho de usar el programa pero no puede modificarlo en lo que respecta al código fuente, ya que este código no está disponible o existen trabas legales que no lo permiten. Esto significa que el usuario no puede realizar mejoras o cambios sobre el software, lo cual es análogo a comprar cualquier equipo o suministro y no poder modificarlo para ser usado a medida y de esta manera obtener el máximo beneficio del mismo. La principal ventaja del software propietario es que está respaldado por el proveedor, generalmente una importante empresa, quien continuamente mejora y extiende las capacidades de sus soluciones. Programas de este tipo son, por ejemplo, AutoCAD, Microsoft Office, METSIM, Adobe Photoshop, HSC, etc.

El software propietario es el que más se conoce y que el mayor aplicación tiene tanto en particulares como en empresas. Los programas de este tipo suelen ser bastante robustos y con un amplio número de funciones y características. Como se mencionó anteriormente, su código cerrado y por lo tanto también su desarrollo y extensión. Adicionalmente, las empresas detrás

de estos programas entienden el software como una respuesta a un problema particular y no como una herramienta extensible, lo cual obliga a las mismas a desarrollar extensiones y funciones bastante generales y que, en algunos casos, un determinado usuario jamás va a necesitar. Esto también crea una exclusiva y excesiva dependencia con la empresa desarrolladora del programa, lo cual otorga como principal desventaja la fijación precios bastante excesivos. Es interesante analizar el costo de estos programas y la real aplicación que se le darán a los mismos, en pocas palabras, no se debería justificar el empleo de programas tan potentes si sólo se usará para realizar una o pocas tareas particulares. En la tabla 1.1 se exponen los programas más representativos y su costo base.

Tabla 1.1. Típicos programas aplicables en procesos metalúrgicos.

Programa	Desarrollador	Costo base*, dólares
METSIM ^[5]	John Bartlett	\$12,000
HSC Chemistry ^[6]	Outotec	\$1,750
Simulink ^[7]	The MathWorks	\$6,200

(*) Información válida al 5 de Marzo de 2010.

El costo base, representado en la tabla 1.1, se encarece considerablemente si se toma en cuenta que en la mayoría de los casos se deben comprar módulos adicionales. Sólo por dar un ejemplo, en el caso de METSIM el valor indicado corresponde al costo base (8,000 dólares) más el módulo de balance de calor (4,000 dólares). A esto además hay que sumarle el

⁵ <http://www.metsim.com/>

⁶ <http://www.hsc-chemistry.net/>

⁷ <http://www.mathworks.com/products/simulink/>

costo de capacitación que se requiere para usar estas poderosas pero complejas herramientas. Hablamos de costos bastante elevados y sin libertades esenciales si se planea maximizar la función de este software en la aplicación particular de la planta, la única libertad es el extenso uso que se le puede dar y que lamentablemente está limitado a la habilidad de las personas que pueden dominar estos programas. Esto lleva a pensar en alguna solución al respecto que permita la libre extensión y no limite el poder que se puede obtener con la herramienta adquirida.

1.3.2. Software Libre

Se define como software libre a aquel programa informático que respeta extensamente la libertad de los usuarios sobre el producto obtenido en todo nivel imaginable, es decir, puede ser usado, copiado, estudiado, modificado y compartido libremente. El concepto o la denominación fue introducida en la primera mitad de la década del 1980 por Richard Stallman^[8].

Su principal ventaja radica en las libertades mencionadas y la disposición en toda extensión de su código fuente, pero también en que su respaldo o red de desarrolladores es de carácter global y operan sólo con la finalidad de mejorar y colaborar en estas iniciativas, razón por la cual existen muchos proyectos bajo el concepto de software libre. Su principal aplicación se manifiesta en programas gratuitos y libres como por ejemplo, el sistema operativo Linux. Sin embargo, esta filosofía o manera de ver el software no es realmente atractiva desde un punto de vista comercial ya que las libertades en que se basa tienen que estar garantizadas y en ningún caso se aceptan restricciones intermedias de ningún tipo. Esto es observado por la Fundación para el Software libre ya que, como se ha esbozado, el software libre es

⁸ Richard M. Stallman, "Visión general del Sistema GNU", <http://www.gnu.org/gnu/gnu-history.es.html>

impulsado por la búsqueda de conocimiento y no tiene como finalidad o incluso arista, los beneficios comerciales que esto puede implicar. Respecto de este punto y como antecedente adicional, ahora último está tomando mucha fuerza la suite ofimática OpenOffice.org, competencia de Microsoft Office y que es desarrollada por Sun Microsystems en asociación con su comunidad de desarrolladores.

1.3.3. Software Código Abierto

Se define como el software que provee el código fuente del programa y fue derivado del concepto de software libre, esto es porque toma el concepto de libertad del código fuente del software libre pero sin implicar los temas éticos de la libertad de usuario, es decir, es un modelo que sólo toca la libertad de un punto de vista técnico.

Fue así como en el final de la década de 1990 se introdujo el concepto de código abierto como modelo de desarrollo, el cual busca dejar de lado la sensación de anti comercialismo que representa el software libre y además establecer flexibilidades a los programadores que no ofrecen software libre pero que disponen del código fuente de los programas para su revisión o modificación. Esta libertad permite al usuario modificar libremente el programa para ser empleado de manera más extensa y personalizada, sin que esto lo amarre a quien le ha ofrecido la solución original en lo absoluto. Esto es, sin lugar a dudas, el atractivo más importante del código abierto ya que en el caso de una empresa, ésta puede usar el programa como base para el desarrollo de una solución a medida sin incurrir en el alto costo de realizar un programa desde cero. A diferencia del software libre, no existe ningún compromiso ético respecto de dar a conocer estos avances y mejoras, mucho menos distribuir éstos de manera gratuita, lo que permite a la empresa mantener la propiedad sobre sus desarrollos y permite hacer un producto atractivo desde un punto de

vista comercial ya que se pueden establecer variados niveles de estrategias y modelos comerciales.

Cuando un programa código abierto se realiza empleando como base recursos informáticos de software libre, tiene las mismas ventajas y desventajas mencionadas en el modelo de desarrollo del software libre. Sin embargo, cuando no está basado en componentes de este tipo, tiene el mismo robusto soporte que el software propietario respecto a sus cimientos. Esto hace del código abierto un modelo de desarrollo muy atractivo ya que es el único modelo que presenta términos intermedios, incluso con la posibilidad de contar con licencias dobles, es decir, se puede aplicar una licencia libre y una propietaria sin ningún problema. En resumen, desde un punto de vista técnico, el atractivo de este modelo de desarrollo radica en que se pueden ofrecer soluciones con una libertad no tan excesiva y con un beneficio económico importante para el desarrollador, traducible en una solución más económica para el usuario.

Si bien es cierto cualquier desarrollo de software se puede hacer bajo el concepto de código abierto, probablemente resulte ser más interesante hacerlo empleando componentes y recursos gratuitos ya que se abaratan los costos de producción, lo cual obviamente hace aún más atractivo todo el concepto ya que además de libertad se está hablando de algo que es más competitivo desde el punto de vista de la inversión. Esto es tomando en cuenta que los recursos o componentes de software más representativos tienen como respaldo una comunidad global de programadores que velan por el constante desarrollo, sin otro motor más que el deseo de mejorar la herramienta base. Desde luego que existen componentes de baja calidad, escaso respaldo y cuya comunidad de usuarios es bastante reducida. Es fundamental realizar la elección en perspectiva de los recursos a emplear ya que una mala decisión significaría realizar un producto con una fecha de vencimiento corta.

Como se ha mencionado, la característica fundamental del código abierto radica en que el programa puede ser inspeccionado, estudiado, mejorado y conocido exhaustivamente. Esto es vital para la optimización y el conocimiento de lo que estamos usando, como está funcionando o que está haciendo realmente. Por dar un ejemplo, esto permite determinar rápidamente si existe alguna violación no autorizada de propiedad o datos privados producto de un desconocimiento total de la famosa letra chica. Conocer realmente qué hace un programa es obviamente un beneficio importante en todo sentido. Otra ventaja derivada de esta característica principal es que esta apertura permite que la calidad del producto sea alta ya que el código fuente está a la vista y sujeto a inspección. Fácilmente se puede determinar la calidad del mismo.

Como todos sabemos, todo está sujeto a sufrir cambios. Es ideal que el software también lo esté y si es necesario modificar algo esto se haga lo más rápidamente posible y que no suponga meses de espera o retrasos producto que el departamento de informática de la empresa está atado de manos. Esto no es un problema en el código abierto y representa un importante argumento, el código fuente puede ser rápidamente inspeccionado y modificado por cualquier persona con conocimientos informáticos adecuados, sin la necesidad de estar íntimamente ligado a un proveedor determinado.

Un punto interesante a tener en cuenta es que el código abierto implica la realización de estrategias comerciales distintas, que escapan a simplemente ofrecer una licencia por un tiempo determinado. Se pueden determinar al menos cuatro estrategias:

1. Un modelo de licencia dual donde el código fuente es publicado bajo los términos tradicionales del código abierto y simultáneamente con una licencia comercial. Generalmente se cobra por una licencia de carácter perpetuo o por un tiempo determinado, incluyendo en este

costo la facultad de usar legalmente el programa y el acceso a soporte, como también a actualizaciones.

2. Como modelo de servicio donde no se cobra por el software sino por los servicios prestados por el mismo. Se suele hospedar remotamente parte o la totalidad del software.
3. No cobrar por el software pero si por el soporte, entrenamiento y servicios de consultoría.
4. Como encapsulación funcional, donde el producto comercial se distribuye separado del código abierto pero emplea a este ultimo como la base de su funcionamiento. Generalmente no se ofrece soporte ni servicios adicionales, se distribuye el producto como un entorno de desarrollo.

Es importante mencionar que estas estrategias son sólo generalidades y corresponden a lo mayoritariamente realizado por las empresas desarrolladoras.

1.3.4. Ventajas y desventajas del Código Abierto

Sin lugar a dudas el argumento principal a favor del código abierto es la economía que representa, habitualmente se habla que proyectos maduros entregan el 80% de las funcionalidades de una solución propietaria a tan solo un 10% del costo^[9]. Siendo específicos, las ventajas fundamentales del código abierto corresponden a:

1. Sentido de propiedad del software, es decir, quien adquiere una solución no está amarrado con el desarrollador original y por ende permite mayor flexibilidad desde el punto de vista del cliente.

⁹ Sun Microsystems, Inc., "Open Source in the enterprise: Fulfilling the promise", 2009, pág. 5

2. Mayores estándares de programación ya que el código está a la vista y puede ser inspeccionado para determinar la calidad del mismo.

El sentido de propiedad del software permite que se puedan extender las capacidades del programa para las necesidades específicas del usuario, es decir, el usuario puede alterar y extender el programa de acuerdo a su necesidad particular, ya sea para por ejemplo mejorar la interfaz, cambiar el algoritmo que calcula un determinado valor o incluso incorporar nuevas características y funciones. Se podría decir que el programa adquirido es un bien editable y a la vez mejorable, el cual puede ser visto como la base para un posterior desarrollo completamente a medida. Si comparamos esta realidad con lo que por otro lado ofrece el software propietario, encontraríamos que sería similar a comprar una herramienta y no poder mejorarla para que se adapte completamente a una necesidad específica y tener que resignarse a trabajar con software envasado. El sentido de propiedad se traduce en independencia tecnológica, lo cual permite dejar en el pasado la dependencia de centros tecnológicos extranjeros.

La disposición del código fuente permite su inmediata inspección, lo cual incita al desarrollador a trabajar con estándares de programación comúnmente aceptados para poder así ofrecer un código más trabajable y entendible. Esto motiva a los desarrolladores a crear código y por ende programas de mayor calidad, lo cual se traduce en un beneficio directo para el usuario.

Al momento de buscar desventajas del código abierto, podemos contemplar que siempre se habla de su falta de soporte y su poca madurez. Si bien es cierto existen innumerables foros y grupos de usuarios dispuestos a ayudarse mutuamente, un programa con responsabilidades críticas necesitará siempre un soporte de primer nivel. Sin embargo, hay que entender que este problema de soporte está más estrechamente relacionado con los creadores del

programa determinado que con el concepto en sí, un programa mal hecho y realizado con pésimos recursos siempre será malo. Sobre madurez de proyectos código abierto se puede decir bastante. En la tabla 1.2^[10] se muestra la edad de diversos proyectos código abierto y la categoría a la cual pertenecen.

Tabla 1.2. Principales proyectos código abierto.

Categoría	Proyectos	Madurez, años
Base de datos	MySQL, PostgreSQL	12
Plataforma web	Apache, JBoss, GlassFish	10
Herramientas de desarrollo	NetBeans, Eclipse	12
Sistemas operativos	Linux, OpenSolaris, BSD	25
Virtualización	Xen, OpenxVM	6
Navegadores	Mozilla	15
Ofimática	OpenOffice.org	20

Claramente hay bastante madurez en los proyectos mencionados, quienes se han fortalecido enormemente con los años, sin embargo, siempre hay iniciativas nuevas que tratan de mejorar lo existente desde interesantes y novedosos puntos de vista, con el consiguiente problema de inmadurez. Desde luego que también hay iniciativas que no han pasado del año. Sin lugar a dudas, es bastante amplia la realidad del código abierto, pero la recomendación universal es siempre optar por los proyectos, soluciones y herramientas consolidados y dejar madurar aquellos que son relativamente recientes.

¹⁰ Sun Microsystems, Inc., "Open Source in the enterprise: Fulfilling the promise", 2009, pág. 15

1.3.5. El Código Abierto en las empresas

Según Gartner^[11], 85% de las compañías en Asia, Norteamérica y Europa están usando código abierto en algún nivel, donde el principal motivo de su uso es la reducción de costos asociados a las licencias. Si bien es cierto este alto porcentaje es bastante atractivo, hay que tener en cuenta que sólo representa el uso de código abierto en cualquier nivel, es decir, desde un sistema de gestión hasta un programa para enviar correo electrónico. Desde luego el porcentaje mayoritario se lo lleva el software libre.

Según una encuesta realizada en el año 2007 por la IOUG^[12] (Grupo independiente de usuarios Oracle), un 13% de las compañías están usando mayoritariamente código abierto, lo cual representa un aumento del 225% respecto del año 2006, lo cual confirma su condición como una de las tecnologías con más crecimiento. Esta misma encuesta da cuenta que las razones principales para la aplicación del código abierto en una gran empresa (más de 5,000 empleados) corresponden al ahorro de costos (65%) y la libertad respecto del desarrollador original (26%). También da a conocer las limitaciones del código abierto en la empresa, donde el problema fundamental radica en que estas soluciones no proveen el mismo y robusto soporte que una solución propietaria. En efecto, basta con que se realice una analogía entre soluciones propietarias y código abierto de uso diario, no son muy extensos los ejemplos donde el código abierto resulta superior a la solución del tipo propietaria.

Si bien es cierto los antecedentes disponibles para poder realizar una comparación, desde un punto de vista económico, entre software código abierto comercial y software propietario son pocos, se puede realizar una analogía

¹¹ http://news.zdnet.com/2100-9595_22-249842.html

¹² Joe McKendrick, "Open Source in the Enterprise: New Software Disrupts the Technology Stack", 2007.

entre soluciones aplicables a tiendas en líneas (comercio electrónico), lo cual se muestra en la tabla 1.3 y que además tiene la particularidad de considerar solamente soluciones altamente comparables, es decir, todas comparten el mismo lenguaje de programación, base de datos, funciones, características, etc.

Tabla 1.3. Programas (scripts) para tiendas en línea.

Producto	Modelo	Costo anual*, dólares
CS-Cart ^[13]	Propietario	\$285
Interspire Shopping cart ^[14]	Propietario	\$295 - \$1795
SunShop ^[15]	Código abierto (parcial)	\$249
XCART ^[16]	Código abierto	\$115 - \$431

(*) Información válida al 5 de Marzo de 2010.

Los datos de la tabla 1.3 dan cuenta de ahorros significativos, lo cual no siempre será como tal si la solución, sea cual sea su modelo, tiene un amplio número de características y funciones que encarezcan su costo de desarrollo. Respecto de soluciones de características similares, el código abierto es notablemente más económico.

Si bien es cierto el código abierto es atractivo por su bajo costo de desarrollo y por su consecuente menor costo de inversión, su poder o características siempre tienden a poseer un desarrollo poco maduro, lo cual explica la principal traba de su aplicación en la empresa. Pero más allá de esta inmadurez, está el desconocimiento por completo del tema, es decir, las empresas no lo exigen. Afortunadamente el código abierto se está convirtiendo

¹³ <http://www.cs-cart.com/>

¹⁴ <http://www.interspire.com/shoppingcart/>

¹⁵ <http://www.turnkeywebtools.com/sunshop/>

¹⁶ <http://www.x-cart.com/>

en una tendencia y será muy interesante ver su evolución en los siguientes años.

Con estos antecedentes es válido pensar que es posible realizar una solución código abierto, más económica que las disponibles (ver tabla 1.1) para tareas de carácter cotidiano y básico, como por ejemplo, la predicción o balance de un proceso pirometalúrgico, sin implicar mayores funcionalidades y por ende costos relacionados al desarrollo, en otras palabras, tomar las ventajas de costos y apertura del código abierto sin involucrarse en características que aún representan carencias para él y de este modo obtener un programa aplicable a la pirometalurgia del cobre, específicamente, en lo que respecta a los parámetros de operación de un Convertidor Teniente.

1.3.6. Soluciones aplicables a los procesos pirometalúrgicos

Si bien es cierto existen diversas programas aplicables en los procesos pirometalúrgicos (ver tabla 1.1), una de las alternativas más usadas en el proceso de fusión es Microsoft Excel, el cual se emplea principalmente para determinar la carga del horno tomando en cuenta una serie de factores metalúrgicos y realizando los balances pertinentes. El empleo de Excel se debe a que es un cálculo corriente que se puede escribir con relativa facilidad y resulta más económico que alternativas de simulación más robustas ya que raramente se emplean todas las funciones y herramientas que poseen. Obviamente el cálculo será cada vez más complejo de acuerdo a los parámetros y situaciones de simulación que se tomen en cuenta, pero en general es mucho más económico que emplear una simulación con un programa como METSIM.

Las soluciones descritas tienden a tener con problemas de interactividad con el usuario y son complicadas tanto de usar como de entender,

probablemente porque los conceptos de interfaz amigable no son tomados realmente en cuenta para los programas de este tipo y se deja su aplicación para los programas de uso cotidiano y común por todo el mundo, sin embargo, desde la creación del primer computador personal ha existido un interés por este apartado^[17]. El problema de interfaz acarrea un costo constante de capacitación de personal nuevo y también un tiempo excesivo en la determinación de la información porque al operador le resulta laborioso emplear la herramienta en cuestión.

En el caso específico de Excel el problema es mayor ya que entre versiones de Excel la retro compatibilidad tiende a ser problemática en algunos casos y la integridad del cálculo como de la herramienta en sí se puede ver comprometida por cosas tan simples y cotidianas como el empleo de un carácter erróneo o una referencia circular. Adicionalmente, Excel es un programa relativamente caro de actualizar en una empresa ya que Microsoft no ofrece la posibilidad de comprar Excel en volumen (licencias masivas) sino que toda la suite Office^[18], esto hace que la empresa esté pagando por programas que no serán utilizados extensamente. Además, el descuento por licencias masivas solamente existe si adicionalmente se contratan servicios posteriores y renovaciones anticipadas. Tomando en cuenta que Microsoft Office 2010 profesional tiene un valor de 499 dólares, comprar Excel para unos 200 equipos tiene un valor aproximado de 68 millones de pesos (estimación realizada en mayo de 2010).

Según lo descrito anteriormente, una nueva alternativa para estos procesos tiene campo tanto en la relación directa con el usuario u operador y con las nuevas tecnologías y conceptos explicados extensamente en párrafos

¹⁷ <http://www.guidebookgallery.org/articles/ofmiceandmenus>

¹⁸ <http://www.microsoft.com/licensing/mla/summary.aspx>

anteriores. Quizás el código abierto no esté preparado para simulaciones robustas y complejas, pero si para un cálculo simple análogo a lo realizado actualmente por un documento en Excel y con el valor agregado de las consideraciones de usabilidad, interactividad y además con las ventajas del código abierto.

1.4. Objetivos

El objetivo de este trabajo consiste en dar a conocer el concepto de desarrollo de código abierto y su utilización para el proceso de fusión en el Convertidor Teniente. Objetivo que se llevará a cabo mediante el desarrollo de un programa código abierto que, mediante un balance de masa, determine las siguientes condiciones de operación:

- Cantidad de fundente necesario
- Escoria y Eje producidos
- Flujo de Aire de proceso y Oxígeno técnico

La realización del objetivo mencionado implicará integración de disciplinas que van más allá de la metalurgia convencional. Dicho de otra manera, este trabajo además buscará demostrar cómo es posible y aplicable la integración de conocimientos de programación informática a las capacidades del ingeniero metalúrgico.

1.5. Alcances y limitaciones

El alcance de este proyecto corresponderá a la operación de fusión de concentrado en un Convertidor Teniente, donde se realizará sólo el balance de masa para la determinación de parámetros operacionales. El programa desarrollado será de tipo demostrativo, con funciones limitadas y sólo contemplará parte de los requerimientos de un proceso real de planta.

CAPÍTULO II: ANTECEDENTES TEORICOS

2.1. El proceso de fusión de concentrado de cobre

El objetivo de la fusión es llevar a estado líquido el concentrado de cobre sólido y producir en este estado la separación de fase sulfuradas (metal blanco o eje) y fase oxidada (escoria). La fase de sulfuros en lo posible debe contener todo el cobre alimentado mientras que la escoria debe estar, en lo posible, exenta de cobre.

El proceso de fusión ocurre a temperaturas del orden de 1200°C, en un sistema fundido, con suspensión de partículas sólidas en el baño, correspondiente a compuestos de alto punto de fusión (sílice, magnetita, etc). El proceso se puede resumir como producción de Metal blanco o Eje más escoria y gases debido a la reacción de concentrado más fundente y energía.

2.1.1. Productos del proceso

El proceso da como productos principales tres fases:

➤ Escoria

Corresponde a la fase más liviana del material fundido, formada por óxidos de hierro y componentes del fundente agregado. Consiste en una mezcla líquida que contiene principalmente en Fayalita (Fe_2SiO_4), Magnetita (Fe_3O_4) y Sílice (SiO_2). Contiene otros compuestos en menores cantidades como Alúmina (Al_2O_3), Cuprita (Cu_2O) y Calcosina (Cu_2S).

➤ Metal Blanco o Eje (mata)

Corresponde a la fase más densa del material fundido, por lo cual se ubica en la parte inferior del baño y está compuesto principalmente por Cu_2S y

FeS. La cantidad de eje producido está determinada por contenido de azufre en la carga del horno ya que el cobre es el primero que toma el azufre necesario para formar Cu_2S . La proporción es aproximadamente una cuarta parte de azufre por unidad de cobre según la fórmula del Cu_2S . Luego de que el cobre presente se haya combinado con el azufre, el resto del azufre se combinará con el hierro para formar FeS (1,75 unidades de Fe por 1 unidad de S).

➤ **Gases de salida**

Los gases de salida contienen SO_2 generado por las reacciones de oxidación, N_2 proveniente del aire empleado para oxidar el concentrado y pequeñas cantidades de CO_2 , H_2O y compuestos volátiles. El gas tiene un contenido de SO_2 del orden del 10 a 60% y está determinado por la cantidad de aire soplado, el tipo de concentrado y el grado de mata producido. En los últimos años el volumen de SO_2 producido por reactores de fusión ha aumentado debido al uso de aire con mayor cantidad de enriquecimiento, lo cual reduce la cantidad de nitrógeno y quemado de combustible en el horno.

Los gases de salida también suelen contener cantidades substanciales de polvo (hasta 0.3 Kg/Nm^3). El polvo proviene de (1) pequeñas partículas de concentrado que no reaccionó, (2) gotas de mata/escoria que no sedimentaron y (3) elementos volátiles contenidos en el concentrado, como arsénico, antimonio, bismuto y plomo, los que se solidificaron mientras el gas se enfriaba o reaccionaron para formar compuestos no volátiles. El polvo normalmente contiene entre 20 y 40% en peso de cobre, haciéndolo potencialmente valioso. Los gases de salida son normalmente tratados para la recuperación de calor, captación del SO_2 y recuperación de sólidos.

2.1.2. Efecto del fundente

La separación natural a alta temperatura, entre las fases del sistema, puede alterarse positivamente con la adición de fundente. Esto se puede apreciar si se observa el diagrama ternario de la figura 2.1.

De la figura 2.1 se puede observar el efecto de la sílice (SiO_2) en la separación de mata sulfurada y escoria oxidada. A 1200°C y en ausencia de SiO_2 no existiría una separación real entre la fase sulfurada y oxidada, sino una mezcla líquida homogénea.

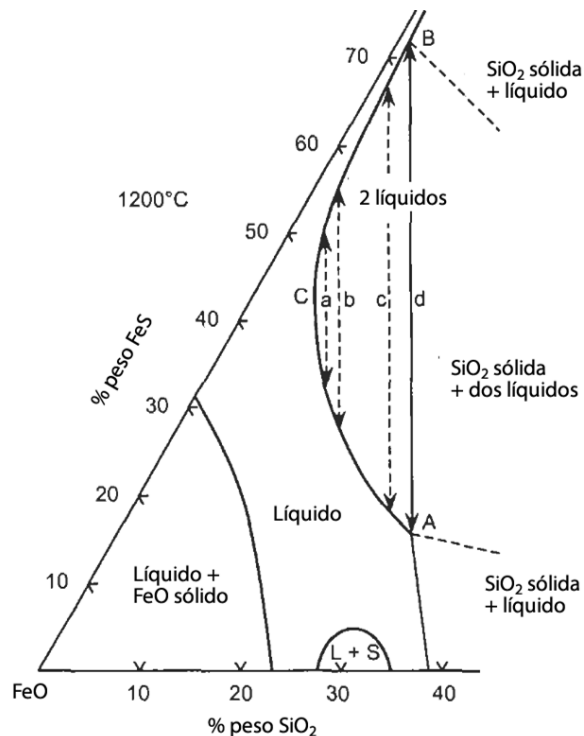


Figura 2.1. Equilibrio líquido-líquido para el sistema FeO—FeS—SiO₂ a 1200°C .

Al adicionar SiO_2 , aparece una zona de inmiscibilidad con una separación entre dos líquidos: Uno rico en FeS y otro rico en FeO. A medida que el contenido de SiO_2 aumenta, también aumenta el grado de separación,

llegándose a un valor máximo con las composiciones más alejadas entre la fase oxidada y sulfurada, para contenidos de SiO₂ del orden de 35 a 40% en peso. A partir de ese momento, cualquier otra adición de fundente involucrara la aparición de una fase sólida rica en sílice. La sílice en la escoria se muestra en el ternario FeO—Fe₂O₃—SiO₂ de la figura 2.2.

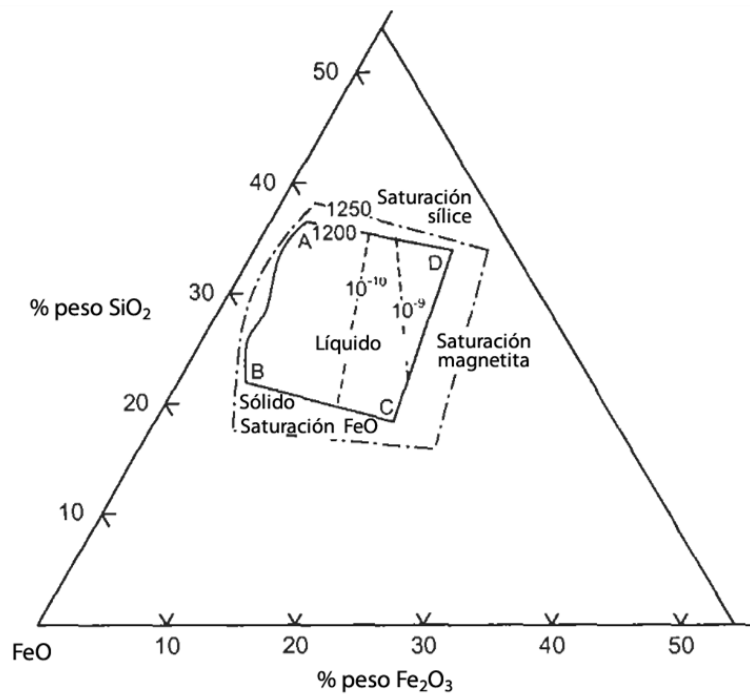


Figura 2.2. Sistema FeO—Fe₂O₃—SiO₂ a 1200 y 1250°C.

Del diagrama de la figura 2.2 se puede observar como existe una pequeña región líquida delimitada en sus costados por tres regiones saturadas: Sílice (SiO₂), magnetita (Fe₃O₄) y wustita (FeO). El proceso de fusión de concentrado de cobre opera típicamente cerca de la saturación con magnetita (línea CD).

Existen pequeñas solubilidades de sílice y oxígeno en la mata, pero se ha demostrado que el incremento de Cu₂S en la mata disminuye estas

solubilidades dramáticamente. Como resultado de esto, las matas industriales contienen cerca de un 1% de oxígeno.

Agregar sílice implica un mayor costo energético además del costo en sí de adicionar este flujo. Por otro lado, la viscosidad de la escoria aumenta con el contenido de sílice. Esto hace que la escoria sea más difícil de manejar y reduce la tasa con la cual las partículas de mata sedimentan a través de la capa de escoria. Si las partículas de mata no pueden decantar rápidamente, éstas quedarán contenidas en la escoria. Lo cual aumentará las pérdidas de cobre.

2.1.3. Efecto del oxígeno

El proceso de fusión se basa en fundir el concentrado gracias a la oxidación de la carga de minerales sulfurados. El oxígeno es comúnmente adicionado al sistema como un flujo de aire enriquecido.

Inyectar grandes cantidades de O_2 oxidará una mayor cantidad del hierro presente en el concentrado, lo cual implicará menor cantidad de sulfuro de hierro en la mata, generando una mata de mayor grado. Sin embargo, usar demasiado oxígeno favorece la oxidación del cobre, el óxido de cobre generado por esta situación se disuelve de manera indeseable en la escoria. Como resultado, agregar la cantidad de O_2 correcta es fundamental para producir un grado aceptable de mata sin generar una escoria alta en cobre.

Las reacciones de oxidación que se verifican en el sistema son exotérmicas, es decir, liberan energía en forma de calor. Esto ayuda a disminuir el consumo de combustible necesario para lograr la elevada temperatura de fusión (~1200 a 1250°C) y para mantener la temperatura del proceso.

2.2. El proceso Teniente

La Figura 2.3 muestra un diagrama con el flujo de materiales del proceso.

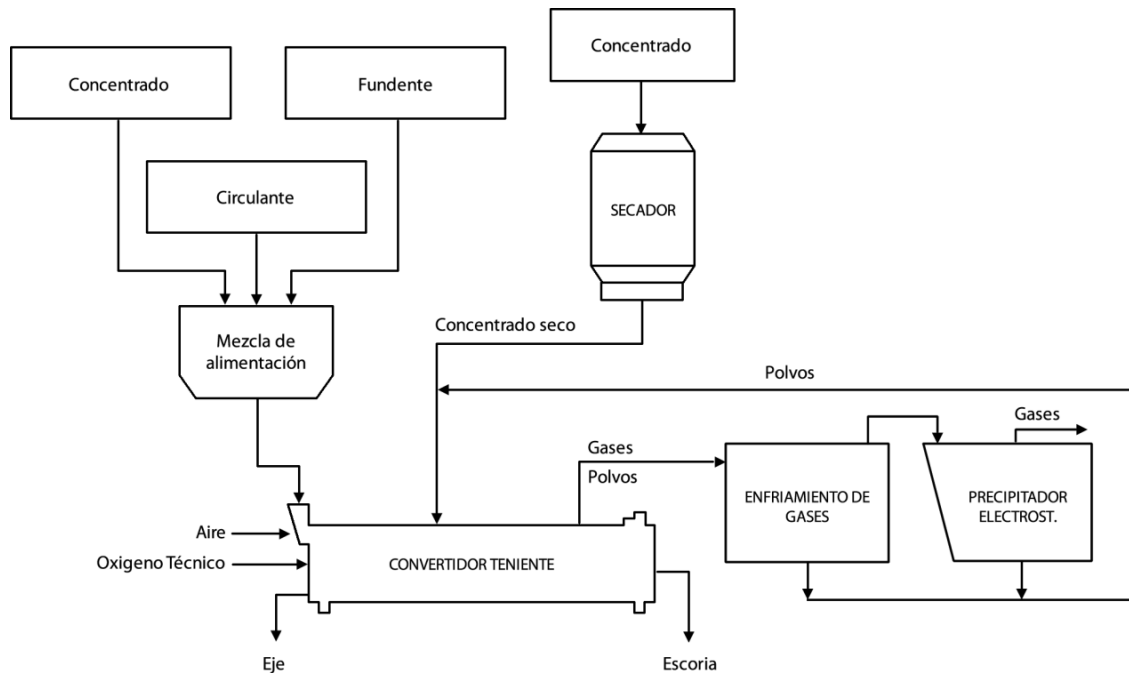


Figura 2.3. Diagrama de flujos de un proceso Teniente.

2.2.1. Flujos de entrada

➤ Concentrado

Proviene de las plantas de beneficio (concentradoras) y se acopia en tolvas de almacenamiento. Es clasificado de acuerdo a su lugar de origen ya que cada concentrado tiene una mineralogía asociada, por lo cual su efecto en el proceso es ligeramente distinto (aporte energético). La mineralogía típica del concentrado contiene Calcopirita (CuFeS_2), Calcosina (Cu_2S), Bornita (Cu_5FeS_4) y Pirita (FeS_2) como sus componentes principales.

➤ **Fundente**

Proviene del cuarzo de mina o de arena de playa y del mismo modo que el concentrado, se almacena en una tolva exclusiva.

➤ **Circulante**

Proviene de restos de material que queda adherido en el fondo de las ollas de eje y escoria. Este material es enviado al área de procesamiento de minerales para su chancado, en donde es triturado y molido para luego ser enviado a una tolva de almacenamiento.

➤ **Aire de proceso**

Es el aire inyectado al Convertidor Teniente, el cual tiene un cierto porcentaje de enriquecimiento en oxígeno. La mezcla Aire-Oxígeno es controlada en función del flujo y características del concentrado.

2.2.2. Flujos de salida

➤ **Metal blanco o Eje**

Consiste en una mezcla sulfuros de cobre y hierro, los cuales salen en forma discontinua del Convertidor Teniente. Esta mezcla es posteriormente enviada al proceso de conversión, el cual se realiza en los convertidores Peirce-Smith.

➤ **Escoria**

La escoria corresponde a la fase que contiene los óxidos producidos por las reacciones de fusión y es inmiscible con el Eje del baño fundido. Al igual que el eje, la escoria es retirada del Convertidor Teniente de forma discontinua a una temperatura aproximada entre 1200 y 1250°C. La escoria es recirculada a

los hornos de limpieza de escorias y/o hornos de reverbero para recuperar el contenido de cobre (5 a 8%).

➤ **Polvos oxidados**

Corresponden a uno de los productos del proceso de oxidación, los cuales son captados y tratados del mismo modo que los polvos sulfurados. Los polvos oxidados recuperados son retornados al Convertidor Teniente.

➤ **Gases**

Los gases están compuestos tanto por los gases producidos por las reacciones de fusión como también por los gases inertes del aire insuflado al Convertidor Teniente. Los componentes principales de los gases de salida son N_2 , SO_2 (8 a 10%), CO , CO_2 , S_2 , O_2 y arrastre de sólidos finos (polveros oxidados) producto de la oxidación de la carga. Los gases son captados por una campana para posteriormente ser tratados para la recuperación de polvos y captación de SO_2 .

2.2.3. Descripción del proceso

El proceso de fusión en el Convertidor Teniente consiste en carga continua de concentrado de cobre y fundente por el inyector de carga sólida o Garr-Gun. El concentrado seco es inyectado al convertidor junto con aire enriquecido con oxígeno a través de una línea de toberas, a una temperatura de aproximadamente $100^{\circ}C$. La inyección de aire enriquecido con oxígeno permite la oxidación del hierro y del azufre presentes en los minerales que constituyen el concentrado. El mecanismo de fusión corresponde a calor generado por oxidación de carga.

Se agrega fundente (sílice) con el objeto de captar el hierro contenido en los minerales sulfurados fundidos y concentrarlos en la parte más liviana de la mezcla fundida (ver sección 2.1.1).

El hierro forma magnetita (Fe_3O_4), la cual se concentra en la escoria y el azufre forma gases los cuales junto a otros gases son evacuados a través de una campana. Los gases producidos en la fusión son utilizados en las calderas para producir vapor, el cual se emplea para generar energía eléctrica a través de turbogeneradores. Los gases de fusión tienen un contenido del orden de 8 a 10% de SO_2 , el cual es captado en gran parte para posteriormente producir ácido sulfúrico (H_2SO_4).

El Convertidor Teniente produce Metal Blanco o Eje con un contenido del orden del 72 a 75% de cobre y una escoria con un contenido de cobre del orden de 4 a 8%.

2.3. Principales variables y parámetros del Convertidor Teniente

➤ Enriquecimiento

El enriquecimiento regula la capacidad de procesamiento y eficiencia energética. A mayor enriquecimiento se tendrá una mayor capacidad de procesamiento y una mayor eficiencia energética. La cantidad de oxígeno es fuertemente controlada para obtener la cantidad deseada de Fe y S oxidados.

➤ Ley del Eje

La ley del eje se ajusta con el flujo de oxígeno alimentado y la entrada de concentrado. Disminuir la ley del Eje implica más tiempo de soplado en el proceso de conversión posterior.

➤ **Inmiscibilidad Eje/escoria**

Corresponde a la separación entre estas fases y se controla con el flujo de fundente y concentrado alimentado.

➤ **Humedad del concentrado**

A menor humedad del concentrado por toberas se tendrá mayor eficiencia energética y capacidad de procesamiento.

➤ **Temperatura de operación**

El desgaste del refractario aumenta drásticamente con la temperatura. La temperatura del eje y la escoria se ajusta con el quemado de combustible.

➤ **Tiempo de soplado**

Determina capacidad de procesamiento, aumentar el tiempo de soplado da como resultado un incremento en la capacidad de procesamiento del Convertidor Teniente.

2.4. Balance de masa

Un balance de masa es una confrontación cuantitativa entre el material que es alimentado al sistema y el material que sale de él. El balance de masa se basa en una de las leyes fundamentales de la ciencia conocida como la “Ley de la conservación de la masa”, la cual fue elaborada por Mijaíl Lomonósov en 1745 y por Antoine Lavoisier en 1785 y establece que la materia no puede ser creada ni destruida en un sistema dado ^[19].

¹⁹ Alan Fine – Gordon Geiger, “Handbook on Material and Energy Balance Calculations in Metallurgical Processes”, TMS 1979

En general mediante un balance de masa se tendrá como resultado una visión general del sistema en lo que respecta del flujo de materiales. Su aplicación es fundamental para las operaciones metalúrgicas y los objetivos del mismo en un proceso metalúrgico dado son variados y dependerán de la etapa en la que se encuentre el proceso. Bajo este concepto los objetivos se pueden clasificar en dos casos:

➤ **Proceso en etapa de diseño**

En este caso el objetivo del balance de masa es establecer las condiciones generales en las que se desenvolverá el proceso, es decir, se trata de determinar los parámetros de operación.

➤ **Proceso en etapa de operación**

Corresponde cuando el proceso ya se está llevando a cabo. En este caso el objetivo del balance de masa es planificar y controlar el proceso evitando problemas operacionales y ayudando en la toma de decisiones.

La figura 2.4 muestra en forma simplificada los flujos másicos que entran y salen del Convertidor Teniente, en donde la carga que ingresa corresponde al concentrado, circulantes, fundente, aire y oxígeno técnico. Por otro lado, los flujos que salen del horno corresponden a Eje, Escoria, polvos oxidados y gases.

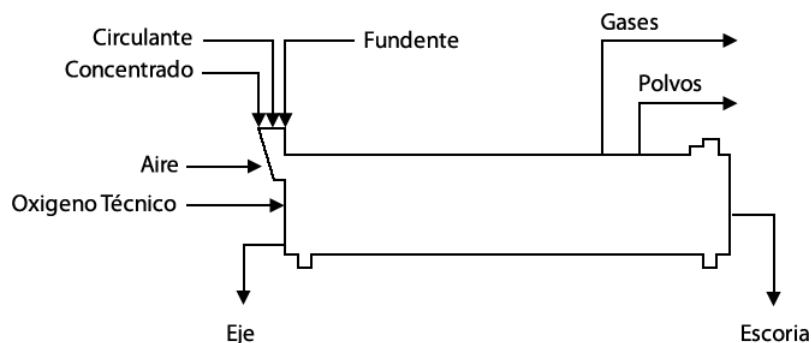


Figura 2.4. Flujos másicos del Convertidor Teniente.

La ecuación general para un proceso continuo es la siguiente:

$$\frac{\text{Masa}_{\text{ENTRA}}}{\text{U. Tiempo}} = \frac{\text{Masa}_{\text{SALE}}}{\text{U. Tiempo}} + \text{Variación de inventario} \quad (\text{Ec. 2.6})$$

Los flujos másicos en unidad de tiempo que entran y salen del sistema se pueden apreciar en la figura 2.4. La variación de inventario corresponde a los cambios de Eje y Escoria en el horno por unidad de tiempo, los flujos de entrada son todos continuos mientras que sólo el gas lo es en los flujos de salida, el Eje y la Escoria varían dependiendo de la unidad de tiempo.

Los cálculos en el proceso de fusión en el Convertidor Teniente se realizan considerando un “Estado estacionario”, el cual se define como el estado de un proceso en el que no hay cambio con respecto al tiempo de alguna condición del proceso, lo cual incluye la cantidad y composición del material que ingresa al sistema.

2.5. Ajuste de balance de masa

Dado que el proceso está sujeto a variaciones operacionales que afectan la predictibilidad del balance, ya sea por problemas en la línea de alimentación, incoherencias en las composiciones, fallas del reactor y en general una serie de

condiciones de operación, es más que importante tener en cuenta estas variaciones y presentar una solución que busque minimizar los efectos adversos a la predictibilidad del balance. De este modo aparece en el horizonte el ajuste de balance de masa a partir de mediciones durante la operación que permitan proponer los nuevos valores para el proceso.

Como se mencionó anteriormente, existen dos instancias a la hora de analizar un balance:

- a) El proceso se va a llevar a cabo (etapa de diseño).
- b) El proceso se está llevando a cabo (etapa de operación).

En el primer caso (a), se trata de determinar los parámetros de operación, mientras que en el segundo caso (b) se cuenta con mediciones directas de las variables, las que se emplean para determinar los parámetros de corrección que luego serán aplicados en la instancia (a) con el objeto de determinar los nuevos parámetros de operación del proceso.

De este modo, a partir de los valores reales de operación, se puede entregar una mejor predicción de las variables objetivo, estableciendo parámetros de ajuste para estos balances.

2.6. Clasificación de programas informáticos

Al momento de realizar la clasificación de programas informáticos encontraremos que existen variados criterios de segmentación, los cuales son abordados según el punto de vista particular de interés, por ejemplo:

- El o los sistemas operativos compatibles.
- La metodología de desarrollo implementada.
- La función que realiza el programa.

- El precio y condiciones de licencia.
- El lenguaje de programación empleado.
- El modo de ejecución del programa.

Cada una de estas divisiones o clasificaciones otorgan una amplia cantidad de subdivisiones, por ejemplo si nos vamos al primer punto de la lista encontraremos que esta clasificación se dividiría en los sistemas operativos Windows, Linux, OS X, Solaris, Free BSD, etc. Análogamente ocurriría lo mismo para cada punto en la lista salvo para el modo de ejecución del programa ya que corresponde a una clasificación bastante más universal dado que separa los programas de acuerdo a modo de ejecución, es decir, la manera mediante cual el programa logra funcionar. Mediante esta clasificación podemos distinguir dos tipos de programas: (1) Los que se ejecutan de manera compilada y (2) Los que son interpretados.

Los programas que se ejecutan compilados son aquellos que están traducidos a lenguaje de máquina a partir del código escrito del programa (código fuente) y son empleados mediante un archivo ejecutable. Ejemplos de este tipo de programa es la suite Office de Microsoft, Photoshop, HSC Chemistry, Firefox, AutoCAD, METSIM y un largo etcétera. Prácticamente todos los programas que son vastamente conocidos por los usuarios comunes y corrientes son programas que se ejecutan de manera compilada y que reciben el nombre de programas de escritorio.

Por otro lado, los programas interpretados emplean otro programa que va adaptando las instrucciones conforme son solicitadas. Proceso que se denomina *interpretar* y a los programas que lo hacen se los conoce como intérpretes. Un clásico ejemplo es bash, el interpretador de comandos para estaciones Unix. Otros ejemplos de este tipo de programas los podemos percibir a diario ya que la mayor parte o la totalidad de las páginas y servicios

web están hechos con programas que operan de este modo mediante el modelo de cliente-servidor y se denominan programas de lado del servidor.

2.6.1. Programas de escritorio y programas de lado del servidor

Un programa de escritorio es aquel que se instala y/o ejecuta de manera compilada y el programa es un todo en uno. Los archivos y librerías empleados son localizados en la carpeta del programa y emplea librerías (conjunto de recursos) globales del sistema operativo. Por otro lado, los programas de lado de servidor no son compilados sino que son interpretados al momento de realizar una solicitud determinada. Es decir, requieren de un interpretador que lee el código fuente, permitiendo la ejecución del mismo, la interpretación de los comandos y su representación como resultado.

Es importante tener en cuenta que la dependencia de librerías del sistema en la programación de escritorio determina gran parte de las trabas de portabilidad o aplicación en más de un sistema operativo ya que cada sistema tiene su forma particular de funcionamiento y por ende sus propias librerías. Incluso programas que son multiplataforma tienen leves diferencias en cuanto a rendimiento debido a que desde un punto de vista técnico, se podría hablar de dos programas ligeramente diferentes ya que las librerías del sistema son distintas y por ende el código fuente del programa es distinto. La programación de lado de servidor también emplea librerías de sistema pero afortunadamente el impacto es mínimo o nulo, ya que no depende tanto de las librerías sino que del interpretador en particular. De este modo, un interpretador multiplataforma permitiría olvidar la dificultad de portabilidad en diversos sistemas operativos.

Sobre ventajas y desventajas de cada tipo de programación, es destacable que cuando se trata de programación de escritorio, hablamos de programas que pueden contar con una enorme cantidad de recursos a su

disposición si así lo requieren, como disco duro, memoria disponible y procesador. De esta manera podemos ver programas como editores multimedia, juegos, procesadores de texto, etc. Por otro lado, los programas de lado de servidor tienen la ventaja de operar con un consumo pequeñísimo de recursos y generalmente realizan tareas que no demandan alto uso de recursos de sistema. Lógicamente que existen desventajas en cada tipo de programación. En la programación de escritorio la portabilidad multiplataforma encarece notablemente el tiempo y costo de desarrollo en la mayoría de lenguajes informáticos, mientras que en la programación de lado de servidor, se tiene que la puesta en marcha puede ser engorrosa y la velocidad con la cual opera el programa es afectada por la conexión de la red cuando se está operando como servidor en una serie de equipos cliente.

Una importante ventaja de la programación de lado de servidor radica en que estos programas pueden ser interpretados en un único computador cliente-servidor, análogo a usar un programa de escritorio, o ser instalados en un servidor y disponer del programa a computadores clientes, de esta manera el programa puede ser usado en red ya sea de manera local o remota (servidor en internet). Esto determina la diferencia fundamental entre ambos tipos de programación ya que establece maneras distintas de usar los programas. El programa de escritorio se ejecuta y corre como proceso en el computador donde está instalado mientras que el programa de lado de servidor es solicitado, usualmente vía navegador web, en el servidor donde se encuentra alojado/interpretado y el servidor interactúa con cada cliente centralizando la información y ejecución del programa. De esta manera, un número ilimitado de computadores puede tener acceso al programa con tan solo instalarlo en un servidor. La figura 2.5 ilustra los pasos que efectúa el cliente-servidor en un programa de este tipo.



Figura 2.5. Esquema simple del modelo cliente-servidor.

Si bien es cierto los programas de lado del servidor están hechos para trabajar en red, esto no los imposibilita para funcionar como un programa de escritorio aún teniendo en cuenta que trabajar en red tiene la ventaja de centralizar la información y es más eficiente cuando más de una persona va a emplear el programa, además, se puede confiar de datos que no se pueden manipular ya que están en un servidor donde solamente hay acceso a para usar el programa y no editar sus archivos, copiar la carpeta que lo contiene, alterarlo, etc.

Por otro lado también puede ser dispuesto para trabajar a través de internet, lo cual le da un importante punto a favor. Por dar un ejemplo, el jefe de turno puede ver desde su casa como operó el horno en un determinado momento o leer los comentarios de una operación mientras está de viaje. Si bien es cierto trabajar a través de internet tiene interesantes ventajas, hay que tener en cuenta que este programa realiza una operación fundamental y no se

puede interrumpir su labor por problemas de conectividad o acceso a internet. Un antecedente al respecto es lo que ocurrió el día 27 de febrero de 2010 cuando producto de un gran terremoto el país perdió prácticamente por completo su conectividad por varios días, afectando el sistema interconectado central, internet, la telefonía móvil y fija, etc. Por lo tanto, el programa estará orientado a su aplicación sin dependencia de internet.

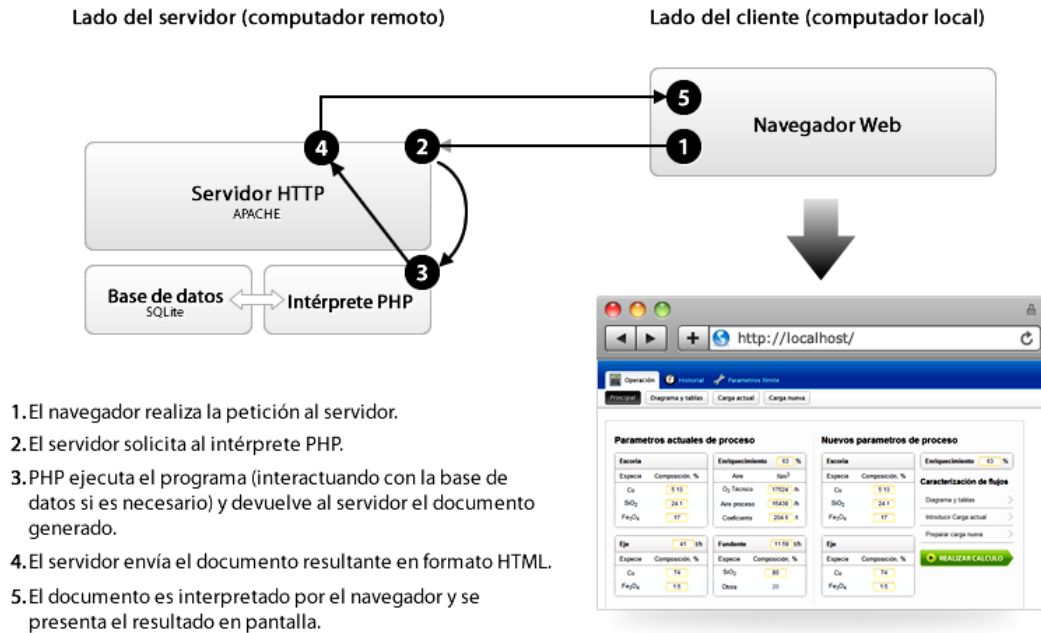
2.7. Modelo cliente servidor

Corresponde a la estructura que rige el funcionamiento de un programa de lado del servidor y consiste básicamente en un programa cliente que realiza peticiones a otro programa ubicado en el servidor.

Cuando se emplea el programa, se establece un proceso en el cual participan dos partes. Por un lado el usuario quien ejecuta un programa en el computador local, conocido como *programa cliente*, el cual se encarga de contactar al computador remoto para solicitar el servicio deseado. El computador remoto por su parte responderá a lo solicitado mediante la ejecución de otro programa, denominado *programa de servidor*. Los términos cliente y servidor se emplean tanto para referirse a los programas que cumplen estas funciones como para denominar a los computadores donde son ejecutados los programas. Respecto del programa cliente este realiza dos funciones distintas:

1. Se encarga de gestionar la comunicación con el servidor y recibir los datos enviados por este.
2. Presenta al usuario los datos en pantalla y le ofrece los comandos necesarios para utilizar las prestaciones que ofrece el servidor, es decir, dispone de la interfaz.

La figura 2.6 muestra un esquema resumido y macro del modelo cliente servidor con cada uno de los pasos involucrados.



- 1.El navegador realiza la petición al servidor.
- 2.El servidor solicita al intérprete PHP.
- 3.PHP ejecuta el programa (interactuando con la base de datos si es necesario) y devuelve al servidor el documento generado.
- 4.El servidor envía el documento resultante en formato HTML.
- 5.El documento es interpretado por el navegador y se presenta el resultado en pantalla.

Figura 2.6. Esquema resumido del modelo cliente-servidor.

2.8. Tipos de programación

Existe una amplia diversidad de maneras de programar, las que se diferencian por el enfoque particular o filosofía de construcción de un programa, lo cual se conoce como paradigma de programación. Ningún paradigma o tipo de programación es mejor que otro, cada uno tiene ventajas y desventajas de acuerdo a la situación en la que se está siendo empleado y la mayoría de las veces se puede mezclar el tipo de programación en un programa determinado. Los tipos de programación se pueden clasificar en:

➤ Programación no estructurada

Se basa en secuencias de instrucciones en donde el programa no sigue ningún orden de ejecución particular o claramente legible. El código fuente

resulta ser cada vez más complicado de entender a medida que se incorporan instrucciones, lo cual hace más complicado el desarrollo. Fue el primer tipo de programación que apareció, pero con el tiempo fue remplazado por la programación estructurada ya que cada vez los requerimientos de funciones o tareas a realizar eran mayores.

➤ **Programación estructurada**

Se originó a finales de la década de 1960 como una manera de solucionar los problemas derivados de la programación no estructurada. Este tipo de programación organiza el código fuente en una serie de estructuras, indicando claramente un inicio y fin para cada una de ellas. Esto hizo más fácil escribir el código fuente y la vez lo hizo más legible.

➤ **Programación procedural o por procedimientos**

Es un tipo de programación estructurada en donde el código se divide en procedimientos llamados “funciones”. Cada función realiza una instrucción determinada donde el programador puede establecer variables para cada caso. De este modo, mediante la misma función, se pueden ejecutar varias veces procedimientos distintos de acuerdo a las variables ingresadas. El empleo de estas funciones permite que el código no sea repetitivo lo cual facilita tanto su comprensión, edición y reduce el peso o espacio que ocupa el programa. La principal ventaja de este tipo de programación resulta ser la facilidad con la cual se puede entender el código y la flexibilidad que entregan las funciones.

➤ **Programación orientada a objetos**

Es un paradigma que usa objetos y sus interacciones. Se introdujo en la década de 1970 pero su popularización ocurrió sólo a finales de la década de 1990. Es un tipo de programación muy similar a la procedural con la diferencia

en que las variables y las funciones son agrupadas en “clases” quienes definen un “objeto”. De este modo se busca minimizar lo más posible la reiteración del código fuente. La principal desventaja de este paradigma es que es complicado de entender y trabajar, lo cual implica el empleo de software de entornos de desarrollo en lugar de simples editores de código fuente. Sin embargo, es la elección más conveniente a la hora de planear un programa de alta demanda y de gran envergadura ya que el código fuente está muy bien organizado.

2.9. Base de datos

En informática, una base de datos es un conjunto de datos organizados en un mismo contexto, almacenados sistemáticamente y dispuestos de manera digital. El concepto se puede entender como un “almacén” en donde es posible guardar grandes cantidades de información de forma organizada, lo que permite tanto la búsqueda como la inserción de datos. De este modo se puede tener por un lado el programa con todas sus funciones e instrucciones y por otro lado la base de datos con los resultados, preferencias, etc.

Las bases de datos se emplean en conjunto con un Sistema de Gestión de Base de Datos (en inglés *Database Management System*) el cual es un software dedicado a servir de interfaz entre la base de datos, el usuario y los programas que la utilizan. El Sistema de Gestión de base de Datos (SGDB) permite la definición, manipulación y consulta de datos. El conjunto base de datos y el SGDB conforman el sistema de base de datos, el cual es empleado finalmente por el programador para la realización del programa.

2.9.1. Características de las bases de datos

➤ **Independencia**

Una de las características fundamentales del sistema de base de datos es la independencia de datos, lo cual implica que la información no está relacionada estrictamente a un programa en particular y puede ser solicitada, exportada y modificada de acuerdo al problema particular. De este modo se puede tener un sistema de base de datos compartido, el cual es empleado para más de un propósito particular.

➤ **Acceso simultáneo**

El sistema de base de datos permite que la información esté disponible y trabajable de manera concurrente, es decir, más de un usuario puede estar solicitando la información al mismo tiempo.

➤ **Integridad de datos**

La integridad de la base de datos se refiere a la validez y la consistencia de los datos almacenados. Normalmente la integridad se expresa mediante restricciones o reglas que no se pueden violar. El SGBD se encarga de mantener la integridad de datos, evitando almacenamiento o edición errónea de los datos.

➤ **Respaldos**

El SGBD permite la generación de respaldos de la base de datos, ya sea por tarea programada o solicitud en demanda. Los respaldos permiten que la información se encuentre más segura ya que se podría contar con una copia de los datos, minimizando así pérdidas derivadas de un problema de *hardware*.

2.9.2. Modelo de base de datos relacional

Dado que la naturaleza de los datos puede ser muy variable, se han desarrollado distintos tipos y modelos de base de datos. Cada tipo de base de datos tiene una finalidad particular y definen distintas maneras de administrar los datos, ya sea privilegiando el alto rendimiento, la velocidad de transferencia, la cantidad de información disponible, etc.

En la actualidad el modelo de base de datos más empleado es el relacional, el cual se basa en el empleo de interconexiones (relaciones) a la hora de manejar los datos. En este modelo, el lugar y la forma con que se almacenan los datos no tiene relevancia, ya que la información se relaciona mutuamente sin jerarquías. Esto le permite ser un modelo más entendible de trabajar y por ende altamente popular.

Una base de datos relacional está compuesta por “tablas” las que a su vez contienen “filas” con los registros o datos. Mediante este modelo es posible relacionar los datos de una tabla en otra. De este modo, es muy simple realizar consultas aparentemente complejas y minimiza la redundancia de datos ya que no es necesario duplicar registros sino simplemente relacionarlos.

El lenguaje más habitual para construir las bases de datos relacionales es SQL (*Structured Query Language* o Lenguaje Estructurado de Consultas), el cual es un estándar implementado por los principales motores o sistemas de gestión de bases de datos relacionales.

CAPÍTULO III: DESARROLLO DEL TEMA

3.1. Cálculo del balance de masa

El objetivo del balance de masa es determinar ciertos parámetros de operación con los cuales debe trabajar el proceso: (1) Cantidad de fundente necesario, (2) Aire de proceso y (3) Oxígeno técnico. Estos parámetros son calculados a partir de un porcentaje conocido de enriquecimiento, ley del eje y cantidad de sílice en la escoria, al fundir una cierta cantidad de concentrado.

Primeramente se determinará la cantidad de eje, escoria y fundente. Para ello se consideran correlaciones empíricas en función de la ley del eje, contenido de sílice, fierro y otros.

3.1.1. Cálculo de la cantidad de Eje, Escoria y Fundente

Dada una ley de Eje, composición de la escoria, los flujos máxicos de Concentrado, Circulante y conociendo además el porcentaje de la alimentación total que se va en los polvos oxidados (remoción de polvos) y la composición del fundente, se procede a determinar la cantidad producida de Eje y Escoria junto con el fundente necesario para el proceso. Para ello se realizan los siguientes balances:

➤ **Balance de SiO₂**

$$\text{SiO}_{2\text{ CONC}} + \text{SiO}_{2\text{ CIRC}} + \text{SiO}_{2\text{ FUND}} + \text{SiO}_{2\text{ CARBON}} = \text{SiO}_{2\text{ ESC}} + \text{SiO}_{2\text{ POLVOS}} \quad (\text{Ec. 3.1})$$

➤ **Balance de Otros**

$$\text{Ot}_{\text{CONC}} + \text{Ot}_{\text{CIRC}} + \text{Ot}_{\text{FUND}} + \text{Ot}_{\text{CARBON}} = \text{Ot}_{\text{ESC}} + \text{Ot}_{\text{POLVOS}} \quad (\text{Ec. 3.2})$$

➤ **Balance de Cobre**

$$\text{Cu}_{\text{CONC}} + \text{Cu}_{\text{CIRC}} = \text{Cu}_{\text{ESC}} + \text{Cu}_{\text{EJE}} + \text{Cu}_{\text{POLVOS}} \quad (\text{Ec. 3.3})$$

➤ **Balance de Hierro**

$$\text{Fe}_{\text{CONC}} + \text{Fe}_{\text{CIRC}} = \text{Fe}_{\text{ESC}} + \text{Fe}_{\text{EJE}} + \text{Fe}_{\text{POLVOS}} \quad (\text{Ec. 3.4})$$

La resolución de las ecuaciones anteriores entrega las siguientes expresiones, donde las incógnitas a determinar aparecen encerradas en líneas segmentadas:

$$\text{SiO}_2_{\text{CONC}} + \text{SiO}_2_{\text{CIRC}} + \text{SiO}_2_{\text{CARBON}} - \text{SiO}_2_{\text{POLVOS}} = [\% \text{SiO}_2_{\text{ESC}}] \cdot \boxed{\text{ESC}} - [\% \text{SiO}_2_{\text{FUND}}] \cdot \boxed{\text{FUND}} \quad (\text{Ec. 3.5})$$

$$\text{Ot}_{\text{CONC}} + \text{Ot}_{\text{CIRC}} + \text{Ot}_{\text{CARBON}} - \text{Ot}_{\text{POLVOS}} = [\% \text{Ot}_{\text{ESC}}] \cdot \boxed{\text{ESC}} - [\% \text{Ot}_{\text{FUND}}] \cdot \boxed{\text{FUND}} \quad (\text{Ec. 3.6})$$

$$\text{Cu}_{\text{CONC}} + \text{Cu}_{\text{CIRC}} - \text{Cu}_{\text{POLVOS}} = [\% \text{Cu}_{\text{ESC}}] \cdot \boxed{\text{ESC}} + [\% \text{Cu}_{\text{EJE}}] \cdot \boxed{\text{EJE}} \quad (\text{Ec. 3.7})$$

$$\text{Fe}_{\text{CONC}} + \text{Fe}_{\text{CIRC}} - \text{Fe}_{\text{POLVOS}} = [\% \text{Fe}_{\text{ESC}}] \cdot \boxed{\text{ESC}} + [\% \text{Fe}_{\text{EJE}}] \cdot \boxed{\text{EJE}} \quad (\text{Ec. 3.8})$$

$\% \text{Ot}_{\text{ESC}}$: Determinado en ecuación 8.8 del Apéndice A.

$\% \text{Ot}_{\text{FUND}}$: Determinado a partir del $\% \text{SiO}_2_{\text{FUND}}$

$\% \text{Fe}_{\text{ESC}}$: Determinado en ecuación 8.9 del Apéndice A.

$\% \text{Fe}_{\text{EJE}}$: Determinado en ecuación 8.3 del Apéndice A.

Los “otros” indicados en las ecuaciones 3.6 y 3.7 corresponden a los óxidos inertes al proceso.

Las incógnitas que aparecen encerradas en las ecuaciones 3.5, 3.6, 3.7 y 3.8, se determinan mediante un sistema de ecuaciones de 4x4, el que se resuelve en forma matricial de la siguiente manera:

$$\begin{array}{cccc}
 & \text{Eje} & \text{Fund.} & \text{Esc.} & \text{Ot. Esc.} \\
 A = & \begin{bmatrix} A_1 & 0 & A_2 & 0 \\ 0 & A_3 & A_4 & 0 \\ 0 & A_5 & 0 & A_6 \\ A_7 & 0 & A_8 & A_9 \end{bmatrix} & & B = & \begin{bmatrix} B_1 \\ B_2 \\ B_3 \\ B_4 \end{bmatrix} \\
 & & & & \text{(Ec. 3.9)}
 \end{array}$$

Cada uno de los elementos de la de la matriz A corresponden a:

$$\begin{array}{lll}
 A_1 = \%Cu_{\text{EJE}} & A_2 = \%Cu_{\text{ESC}} & A_3 = - \%SiO_2_{\text{FUND}} \\
 A_4 = \%SiO_2_{\text{ESC}} & A_5 = - \%Otros_{\text{FUND}} & A_6 = \%Otros_{\text{ESC}} \\
 A_7 = \%Fe_{\text{EJE}} & A_8 = \%Fe_{\text{ESC}} & A_9 = - \%Fe_{\text{FeO}}
 \end{array}$$

Por otro lado los elementos de la matriz de constantes B corresponden a:

$$\begin{array}{ll}
 B_1 = \frac{Cu_{\text{CONC}} + Cu_{\text{CIRC}}}{1 + \%Polvos} & B_2 = \frac{SiO_2_{\text{CONC}} + SiO_2_{\text{CIRC}}}{1 + \%Polvos} + SiO_2_{\text{CARBON}} \\
 B_3 = \frac{Ot._{\text{CONC}} + Ot._{\text{CIRC}}}{1 + \%Polvos} + Ot._{\text{CARBON}} & B_4 = \frac{Fe_{\text{CONC}} + Fe_{\text{CIRC}}}{1 + \%Polvos}
 \end{array}$$

Al multiplicar la matriz inversa de A por la matriz B se obtiene la cantidad de Eje y Escoria producida como también la cantidad de fundente necesario:

$$X = A^{-1} \cdot B \quad (\text{Ec. 3.10})$$

3.1.2. Cálculo del coeficiente de oxígeno

Conocida la cantidad de Eje y Escoria se procede a realizar el balance de azufre y oxígeno para calcular el Coeficiente de oxígeno, el cual se define como la cantidad de oxígeno necesario para producir, a partir de un concentrado de características dadas, un Eje y Escoria de una determinada ley. El coeficiente de oxígeno se obtiene mediante la siguiente expresión:

$$\text{Coeficiente de Oxígeno} \left[\frac{\text{Nm}^3}{\text{ton}} \right] = \frac{\text{Demanda de oxígeno} \left[\frac{\text{Nm}^3}{\text{h}} \right]}{\text{Concentrado} \left[\frac{\text{ton}}{\text{h}} \right]} \quad (\text{Ec. 3.11})$$

La demanda de oxígeno (determinada en Ec. 8.12 del Apéndice B) se calcula mediante el balance previamente mencionado, el cual toma en cuenta la entrada de estos elementos en el concentrado y circulante, mientras que por otro lado toma la salida de los mismos en el eje, escoria y polvos.

3.1.3. Cálculo del oxígeno técnico

Conocido el coeficiente de oxígeno se determina el flujo de oxígeno técnico en Nm^3/ton mediante la siguiente expresión:

$$\text{Oxígeno Téc.} \left[\frac{\text{Nm}^3}{\text{ton}} \right] = \frac{\text{Coef. Ox.} \left[\frac{\text{Nm}^3}{\text{ton}} \right]}{\%N_{2 \text{ AIRE}}} \times \left(1 - \frac{\%O_{2 \text{ AIRE}}}{\% \text{Enriquecimiento}} \right) \quad (\text{Ec. 3.12})$$

La expresión anterior emplea el porcentaje de enriquecimiento, el cual es un parámetro de operación con el cual se desea operar el horno.

Considerando que el aire está compuesto por 21% de Oxígeno, 79% de Nitrógeno y 1% de otros, el programa mostrará el flujo de oxígeno técnico en Nm³/h, para lo cual se emplea la siguiente expresión:

$$\text{Oxígeno Téc.} \left[\frac{\text{Nm}^3}{\text{h}} \right] = \text{Oxígeno Téc.} \left[\frac{\text{Nm}^3}{\text{ton}} \right] \times \text{Concentrado} \left[\frac{\text{ton}}{\text{h}} \right] \quad (\text{Ec. 3.13})$$

3.1.4. Cálculo del aire de proceso

Conocido el coeficiente de oxígeno, se determina el flujo de aire de proceso mediante la siguiente expresión:

$$\text{Aire proceso} \left[\frac{\text{Nm}^3}{\text{ton}} \right] = \frac{\text{Coef. Oxígeno} \left[\frac{\text{Nm}^3}{\text{h}} \right] - \text{Oxígeno Téc.} \left[\frac{\text{Nm}^3}{\text{h}} \right]}{\%O_2 \text{ AIRE}} \quad (\text{Ec. 3.14})$$

Al igual que el Oxígeno técnico, el programa mostrara el flujo de aire de proceso en Nm³/h, lo cual se realiza mediante el siguiente cálculo:

$$\text{Aire proceso} \left[\frac{\text{Nm}^3}{\text{h}} \right] = \text{Aire proceso} \left[\frac{\text{Nm}^3}{\text{ton}} \right] \times \text{Concentrado} \left[\frac{\text{ton}}{\text{h}} \right] \quad (\text{Ec. 3.15})$$

3.2. Cálculo de los parámetros de ajuste

Como se mencionó anteriormente (sección 2.4) una operación de esta naturaleza está sujeta a imprecisiones en la predictibilidad del balance, razón por la cual se introduce el concepto de parámetro característico de ajuste y su objetivo es mejorar la consistencia de la operación. Los parámetros característicos serán adicionados algebraicamente para la determinación de los nuevos parámetros de operación con los cuales trabajará el horno.

3.2.1. Parámetro característico para el fundente

Conocidos los flujos máxicos con los cuales se está alimentando el horno y además conociendo las condiciones reales con las cuales ocurrió la operación, se puede establecer un parámetro de ajuste para el flujo máxico de fundente.

Se hace un balance de sílice (SiO_2) con los datos de la operación actual, obteniendo el flujo máxico de fundente necesario para obtener los resultados deseados, es decir, se obtiene un flujo teórico de fundente. Estos resultados se obtienen mediante el sistema de ecuaciones descrito en la sección 3.1.1 (ecuación 3.9).

Posteriormente mediante la comparación de los flujos de fundente (real y calculado) se determina el parámetro característico de ajuste mediante la siguiente expresión:

$$\text{Parám. de Ajuste Fund. } \left[\frac{\text{ton}}{\text{h}} \right] = \text{Flujo Fundente (Calculado - Real)} \quad (\text{Ec. 3.16})$$

Donde el flujo real de fundente se obtiene mediante medición.

3.2.2. Parámetro característico para el oxígeno

Del mismo modo que el fundente, se determina un parámetro característico de ajuste para el oxígeno. Se tendrá que la diferencia entre el flujo volumétrico de oxígeno teórico con el real que se está usando será el parámetro característico de ajuste. El cálculo de este parámetro se realiza mediante la siguiente expresión:

$$\text{Parám. de Ajuste Oxígeno } \left[\frac{\text{Nm}^3}{\text{h}} \right] = \text{Flujo Oxígeno (Teórico - Real)} \quad (\text{Ec. 3.17})$$

El valor del parámetro característico de ajuste para el oxígeno se emplea para corregir el coeficiente de oxígeno, el cual es el que se debe usar para operar el horno. El coeficiente de oxígeno ajustado (Nm³/ton) será:

$$\text{Coef. Ajust.} = \frac{\text{Parám. Ajuste Ox.} \left[\frac{\text{Nm}^3}{\text{h}} \right] + \text{Demanda Ox.} \left[\frac{\text{Nm}^3}{\text{h}} \right]}{\text{Concentrado} \left[\frac{\text{ton}}{\text{h}} \right]} \quad (\text{Ec. 3.18})$$

Conocido el Coeficiente de oxígeno ajustado, se determina el aire de proceso ajustado mediante las ecuaciones 3.14 y 3.15.

El ajuste de oxígeno permite calcular una nueva ley del Eje para los nuevos parámetros de operación. Si existe diferencia entre los valores del coeficiente de oxígeno, el programa realizará iteraciones variando la ley propuesta y realizando el balance de la ecuación 3.9 a modo de obtener un nuevo coeficiente. Esta iteración terminará cuando ambos coeficientes sean iguales o se superen los límites preestablecidos de ley del Eje.

3.3. Planificación del programa

Conocido el balance de masa, la determinación del conjunto de herramientas y tecnologías de software dependerá de las necesidades particulares del programa. Necesidades que tienen como base fundamental el balance de masa descrito previamente, pero para hacer un desarrollo más atractivo, se establecerá como objetivo determinar funciones y características que le otorguen al programa un valor agregado, sin perder del horizonte que además debe ser económicamente más atractivo. Esto le proporcionará una mayor competitividad respecto del resto de soluciones disponibles.

3.3.1. Características principales y específicas del programa

1. Realizar el balance de masa básico de un Convertidor Teniente para una operación teórica propuesta.
2. Ser desarrollado bajo el concepto de código abierto y empleando recursos gratuitos y/o libres.
3. Presentar una pulida interfaz de usuario que permita un uso intuitivo y simple del programa.
4. Poseer sistemas de validación de datos de entrada para evitar problemas derivados de la integridad del cálculo.
5. Usar un sistema de base de datos para poder contar con un historial de operaciones.
6. Posibilidad de configurar ciertos valores y límites esperados de los parámetros.
7. Retroalimentación mediante comparación de valores reales versus calculados.

Sobre la retroalimentación indicada en el punto número 7, cabe mencionar que esta característica inserta al programa en el marco de control de gestión operativa ya que se tienen elementos de planificación (cálculo de los parámetros a controlar) y posterior a esto el control de gestión (revisión de los resultados obtenidos). Al evaluar o comparar estos resultados el programa es capaz de corregir las desviaciones a modo de lograr el resultado deseado y como consecuencia ayudar en el ciclo de toma de decisiones.

3.3.2. Elección de las tecnologías a emplear

Existen innumerables alternativas a la hora de seleccionar las tecnologías o el conjunto de herramientas que dan forma a un programa, incluso si se toma en cuenta que debe ser desarrollado bajo el concepto de

código abierto y con recursos gratuitos y/o libres. La cantidad de herramientas o proyectos disponibles es francamente enorme, sin embargo, esta amplia oferta de recursos debe ser acotada en privilegio de la que represente mejor en conjunto las siguientes características:

1. Solida madurez y proyectos consolidados.
2. Extensa comunidad de desarrolladores.
3. Amplio número de recursos disponibles.
4. Portabilidad simple o instantánea con diversas plataformas (sistemas operativos).

Las características previamente mencionadas tienen como raíz el lenguaje de programación en cuestión, obviamente lenguajes más consolidados tienen una mayor comunidad de desarrollo, recursos, etc.

Para realizar el análisis de lenguajes de programación respecto de popularidad existe el Índice Comunitario de Programación TIOBE ^[20], el cual es el ranking de los lenguajes de programación más relevantes. Esta lista es elaborada mensualmente según la frecuencia de búsqueda de los términos clave en todos los buscadores importantes como Google, Bing, Yahoo! y Wikipedia. En la tabla 3.1 hay una versión resumida del índice TIOBE para agosto de 2010.

²⁰ http://www.tiobe.com/index.php/content/paperinfo/tpci/tpci_definition.htm

Tabla 3.1. Índice TIOBE para agosto de 2010.

Lenguaje	Posición (Ago. 2009)	Posición (Ago. 2010)
Java	1	1
C	2	2
C++	3	3
PHP	4	4
Visual Basic	5	5
C#	7	6
Python	6	7
Perl	8	8
Objective-C	19	9
Delphi	11	10

De la tabla 3.1 se puede observar la presencia de lenguajes consolidados como Java o C con una inmovilidad de los primeros puestos, lo que da cuenta de una solidez a través del periodo (2009-2010).

Destacable es que en los primeros puestos aparezca el lenguaje PHP^[21] (*PHP Hypertext Pre-processor*), el cual curiosamente cae en una categoría muy particular ya que es un lenguaje interpretado de lado de servidor, empleado para la creación de páginas webs dinámicas, mientras que la mayoría del resto es empleado para la creación de programas de escritorio. Esto habla de la expansión de internet y de la consolidación de PHP como el lenguaje más importante en el área web en los últimos años. El avance y consolidación de PHP es aún más notorio si nos remontamos a datos históricos (tabla 3.2) del índice TIOBE.

²¹ <http://php.net/>

Tabla 3.2. Índice TIOBE histórico para agosto (2005-2010).

Lenguaje	Posición (2005)	Posición (2009)	Posición (2010)
Java	1	1	1
C	2	2	2
C++	3	3	3
PHP	5	4	4
Visual Basic	6	5	5
C#	7	7	6
Python	8	6	7
Perl	4	8	8
Objective-C	43	19	9
Delphi	10	11	10

De la tabla 3.2 se puede ratificar la inmovilidad de Java, C y C++ de los primeros puestos a lo largo de un periodo de 5 años. Como dato, el fuerte avance de Objective-C se debe a que es el lenguaje empleado en la creación de programas para el teléfono móvil iPhone de Apple.

Considerando la simpleza del cálculo necesario y las características ya mencionadas, la elección correcta parece ser PHP ya que se destaca por su simpleza, tener una enorme comunidad de desarrolladores, contar con incontables recursos disponibles y estar en alza constantemente. Elección que tiene en cuenta que lenguajes como C++ son poderosos al lado de PHP y permiten hacer programas extremadamente avanzados, pero también suponen un dominio extenso de lenguajes complejos y su portabilidad multiplataforma debe ser desarrollada en cada caso y no es tan rápida de aplicar como en PHP.

Es muy cierto que PHP no es un lenguaje que permita realizar funciones que son propias de lenguajes mayores pero para efectos del cálculo necesario, cualquier lenguaje podría satisfacer las necesidades establecidas, pero el costo de desarrollar en Java o C++ es mucho mayor que PHP ya que son lenguajes más complejos y justamente encarecen el costo de desarrollo.

3.3.3. Back-end y Front-end del programa

Se procederá a detallar el funcionamiento del programa desde el punto de vista macro, es decir, el *Back-end* y el *Front-end*. En este contexto el término *Back-end* se refiere a la serie de componentes que procesan los resultados o salidas de un programa, mientras que el *Front-end* se refiere a la parte del programa que interactúa directamente con el usuario, también conocida como interfaz gráfica.

➤ Back-end

Como PHP corresponde a programación de lado de servidor, los programas escritos en PHP (denominados Scripts) requieren de un interpretador corriendo en el computador donde se instalara el mismo, cuya labor consiste en darle sentido a las instrucciones escritas en el código fuente. Si es necesario un ejemplo, es análogo a la necesidad de instalar Java Runtime Environment o .NET Framework para poder usar un programa que necesite esos componentes de software.

Los interpretadores usualmente se pueden obtener como módulos o agregados de servidores web HTTP^[22], lo cual implica que se necesita un servidor web con soporte PHP. Afortunadamente los servidores web están

²² <http://www.w3.org/Protocols/>

disponibles en variadas y múltiples opciones para todos los sistemas operativos y su consumo de recursos es mínimo. Según Netcraft^[23], la tecnología de servidor HTTP con mayor uso y por ende popularidad es Apache^[24] con un 52%. Teniendo en cuenta que su competidor más cercano (Microsoft ISS) tiene tan sólo un 17%, Apache es la mejor opción en este apartado tomando en cuenta que además es código abierto. Es importante recalcar que el dominio de Apache se debe a que la puesta en marcha de un servidor de este tipo es totalmente gratuita, pero sin importar el servidor HTTP, todas las soluciones son análogas entre si y realizan la misma tarea. Para ponerlo simple, la elección de uno u otro servidor no afecta al programa escrito en PHP.

Tomando en cuenta que una de las funciones consiste en almacenar las operaciones y mantener un historial de las mismas, es necesario contar con un sistema que permita guardar los resultados generados por el programa. Esto se traduce en sistema de base de datos para almacenar esta información y tenerla conectada con el programa en PHP. Tomando en cuenta los antecedentes mencionados en la sección 2.9 y 2.9.2, la selección tiene que tener en cuenta como principio fundamental un sistema basado en el modelo relacional de base de datos.

La diversidad de sistemas de base de datos relacionales es amplia y abarca todo tipo de necesidades. No obstante, actualmente se puede definir un más que claro predominio por parte de MySQL^[25] en lo que respecta a aplicaciones web. Tanto así que este proyecto comunitario, código abierto, fue adquirido por Sun Microsystems en 2008 por US\$ 1000 millones^[26], quienes sucesivamente fueron adquiridos por Oracle Corporation^[27] en enero del año

²³ http://news.netcraft.com/archives/2010/01/07/january_2010_web_server_survey.html

²⁴ <http://apache.org/>

²⁵ <http://mysql.com/>

²⁶ <http://www.mysql.com/news-and-events/sun-to-acquire-mysql.html>

²⁷ <http://www.oracle.com/us/sun/index.htm>

2010 por US\$ 7.4 billones^[28]. Antecedentes que hacen pensar en MySQL como una alternativa más que obvia, pero en el horizonte aparecen alternativas que claramente pueden ser una mejor aproximación para este programa en particular. Como por ejemplo SQLite^[29], un sistema de base de datos relacional de dominio público que tiene como objetivo fundamental el mínimo uso de recursos.

Es de este modo como SQLite resulta ser una alternativa atractiva ya cumple con los requerimientos básicos y tiene un funcionamiento de consulta muy similar a MySQL ya que comparten el uso del lenguaje SQL de base de datos, lo que implica que la migración de un sistema a otro se puede realizar rápidamente y sin mayor problema. Bajo los conceptos propuestos, SQLite parece ser una solución superior a MySQL en lo que respecta a las necesidades de este programa ya que no se necesita una base de datos de gran tamaño y tampoco necesita un uso de la misma en alta demanda. Sin embargo, por lejos MySQL es la opción ideal para proyectos de una mayor envergadura y compromiso de datos.

➤ **Front-end**

El servidor web con el módulo de PHP y el sistema de gestión de base de datos SQLite son requerimientos necesarios para poder interpretar, generar los resultados y guardar la información, pero esto es sólo una parte del programa ya que se requiere de un apartado fundamental: La interfaz que interactúa con el usuario.

Desde un punto de vista técnico, la interfaz de un programa de este tipo es una representación compuesta de una serie de elementos:

²⁸ http://en.wikipedia.org/wiki/Sun_acquisition_by_Oracle

²⁹ <http://www.sqlite.org/>

1. Lenguaje de Marcado de Hipertexto HTML^[30] (en inglés *HyperText Markup Language*). Es usado para describir la estructura y el contenido de un documento HTML.
2. Hojas de estilo en cascada CSS^[31] (en inglés *Cascading Style Sheets*). Permiten definir el estilo y la forma de la estructura de un documento HTML.
3. JavaScript^[32]. Permite mejorar la interfaz de usuario en diversos niveles, afecta al conjunto HTML y CSS a modo de crear interfaces más avanzadas.

La visualización de esta interfaz tiene un único requerimiento y corresponde a un navegador web a gusto del usuario, esto no es ninguna complejidad ya que todos los computadores personales vienen con un navegador instalado y además existe una amplia y actualizada oferta de soluciones gratuitas disponibles en internet.

Una importante consideración en lo que respecta a la interfaz, es que como puede ser visualizada en cualquier navegador web, tiene que estar hecha y programada de tal manera que indistintamente del navegador web que sea empleado (tomando en cuenta salvedades respecto de versiones muy antiguas) se podrá ver y usar el programa de la misma manera. Esto radica en que cada navegador web interpreta el código HTML con leves diferencias y por ende, si no se tiene la precaución o la experiencia necesaria, se pueden tener importantes diferencias en lo que respecta a la interfaz de acuerdo a estos distintos navegadores. Como antecedente, la tendencia actual es hacia los estándares web, los cuales se definen como una serie de reglas o normas a la hora tanto de escribir el código HTML como a la interpretación que él navegador

³⁰ <http://www.w3.org/html/>

³¹ <http://www.w3.org/Style/CSS/>

³² <https://developer.mozilla.org/en/JavaScript/>

web da a este código fuente, de este modo, navegadores web más recientes satisfacen de mejor manera los estándares que sus versiones predecesoras de hace algunos años. La misión, desde el punto de vista programación, es que esta interfaz se vea idéntica o con pérdidas muy menores a modo de privilegiar la opción de navegador web del usuario y no la del programador, pero también, que el programa detecte el uso de un navegador web muy antiguo y permita ofrecer opciones más recientes para un óptimo desempeño del programa.

3.4. Desarrollo de la interfaz gráfica

Conocidas las funciones y características del programa, se procede con el diseño de la interfaz gráfica del mismo. Para ello es recomendado el uso de un editor gráfico y posteriormente un editor de código fuente en el cual se procederá a transcribir el diseño en lenguaje HTML.

3.4.1. Programas empleados

Respecto al diseño de la interfaz, el uso de un editor gráfico se debe a que se desea crear una interfaz atractiva, clara y simple. Para ello se empleó el programa inkscape^[33] de código abierto y gimp^[34], el cual es software libre. Ambos programas no tienen costo alguno.

Por otro lado, la transcripción del diseño a código HTML y CSS se realiza empleando Notepad++^[35], programa software libre y gratuito para la edición de código fuente para diversos lenguajes de programación y disponible para Microsoft Windows.

³³ <http://www.inkscape.org/>

³⁴ <http://www.gimp.org/>

³⁵ <http://notepad-plus-plus.org/>

3.4.2. Secciones

Cada sección del programa corresponde a una tarea particular que debe efectuar el mismo, desde realizar una operación hasta guardar las preferencias del mismo. Tomando en cuenta la sección 3.3.1, las secciones y sub-secciones que definen al programa son las siguientes:

1. Principal
 - Realizar operación
 - Diagrama y tablas (representación imprimible)
 - Carga nueva (ingreso de la carga nueva)
 - Carga actual (ingreso de la carga actual)
2. Historial
 - Recientes (operaciones recientes)
 - Archivo completo (separado por año y posteriormente mes)
3. Configuración de los parámetros límite

3.4.3. Diseño del menú principal

El menú principal tiene como objetivo fundamental agrupar o condensar de la mejor manera posible las diferentes secciones que tiene el programa. El diseño se centrará en el empleo de menú con pestañas, el cual tiene como característica principal la clara identificación de la sección que se está empleado en un determinado momento.

De este modo, se cuenta con un menú superior el cual contiene las tres pestañas principales y en el interior de cada una de ellas las sub-secciones. La figura 3.1 ilustra de mejor manera la estructura del menú principal.

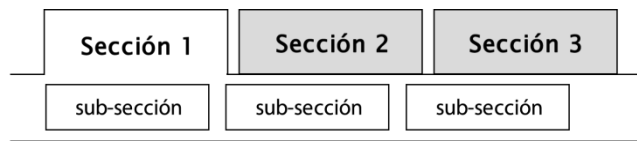


Figura 3.1. Estructura de pestañas del menú principal. En este ejemplo se muestra una sección y sus sub-secciones.

3.4.4. Diseño de las notificaciones y alertas

Una parte fundamental del programa es la interfaz de usuario amigable, la cual, aparte de tener en cuenta consideraciones en lo que respecta al diseño de menús y demás elementos gráficos, toma en cuenta una pseudocomunicación entre el programa y el usuario. Estos elementos de notificación y alertas están diseñados de tal manera de ser una pequeña ayuda al usuario tanto para captar su atención ante un error o indicar la acción que ha realizado el programa.

El primer tipo de alertas corresponde a las que son arrojadas a la hora de ingresar datos de manera errónea (caracteres no permitidos). El programa devolverá, antes de enviar los datos, una alerta indicando el o los campos donde se está cometiendo el error. La figura 3.2a muestra este tipo de avisos para dos condiciones dadas.

El segundo tipo de avisos involucra directamente a los parámetros de operación. Tomando en cuenta los parámetros límite (valores mínimos y máximos que el propio usuario define) el programa indica, al momento de ingresar la información, el rango esperado. La figura 3.2b muestra la situación descrita anteriormente. Cabe señalar que este aviso es sólo una notificación y permite que el usuario ingrese valores que estén fuera de estos límites.

Finalmente, el tercer tipo de avisos corresponde a una notificación de acción. Cuando el programa guarda datos o realiza cualquier tarea de este tipo,

muestra una notificación en el extremo superior derecho indicando el estado de esta acción, ya sea exitoso o no. Esto permite que el usuario sepa qué está pasando y si se está generando o no, por ejemplo, una nueva operación. La figura 3.3 ilustra de mejor manera este tipo de avisos.

Escoria	
Especie	Composición, %
Cu	<input type="text" value="200"/>
SiO ₂	<input type="text" value="abc"/>
Fe ₃ O ₄	<input type="text" value="17"/>

3.2a

Escoria	
Especie	Composición, %
Cu	<input type="text" value="5.13"/>
SiO ₂	<input type="text" value="24.1"/>
Fe ₃ O ₄	<input type="text" value="17"/>

3.2b

Figura 3.2a. Alertas por datos inválidos. Se aprecia el error por valores no esperados y por ingreso de caracteres inválidos. **Figura 3.2b.** Demostración del aviso de rango esperado por parámetro. En este caso, el valor debería estar en el rango de 1 a 6% para el Cobre en la Escoria.

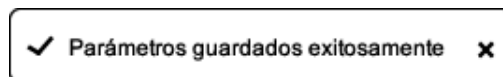


Figura 3.3. Estructura del aviso de acción. En este caso, indica que se almacenaron los parámetros límite en la base de datos. El botón de la derecha permite cerrar este aviso y no entorpecer la visión del programa.

Las alertas mostradas en las figuras 3.2a y 3.2b corresponden a notificaciones presentadas mientras el usuario escribe, de este modo se busca que el usuario no pierda su foco de atención y apenas cometa un error el programa le advierte. Por otro lado, el aviso de la figura 3.3 aparece una vez que el usuario realiza la acción determinada.

3.4.5. Validación de la interfaz

Como se mencionó en la sección 3.3.4 respecto del *Front-end*, el requerimiento para poder visualizar la interfaz es básicamente un navegador web, el cual permite interpretar el HTML y CSS como una interfaz gráfica.

Para evitar cualquier problema derivado de un navegador muy antiguo, principalmente respecto de pérdidas de integridad visual, el programa cuenta con un validador de navegador web. De este modo, navegadores muy antiguos devolverán una pantalla indicándole al usuario que debe actualizar su navegador o bajar alguna alternativa más reciente.

Como la interfaz incorpora elementos de notificación y alertas en JavaScript, es necesario que el navegador empleado también tenga habilitada la capacidad de interpretar el código JavaScript. Es así como también existe un validador para esta situación, entonces si el navegador empleado no tiene JavaScript habilitado, el programa mostrará una pantalla con este problema y dará las instrucciones paso a paso para activar JavaScript, dependiendo del navegador empleado ya que en cada cual la configuración es distinta.

3.5. Escritura del código fuente

Como se explicó en la sección 2.8, existen diversos tipos de programación mediante los cuales es posible escribir el código fuente. Para el desarrollo de este programa se empleó una combinación de dos paradigmas o tipos de programación.

El grueso del código fuente está hecho con programación procedural, es decir, el programa corre linealmente de principio a fin solicitando porciones de código según sea necesario. Se simplificó la carga del sistema separando mediante directorios, funciones y archivos cada tarea particular. De este modo,

existe un archivo distinto que contiene el código empleado en las operaciones, historial y parámetros límite respectivamente.

Por otro lado, una mínima parte del código fuente se escribió en programación orientada a objetos, lo cual fue necesario debido a que PHP no trae incorporadas operaciones matemáticas con matrices. Afortunadamente está disponible una alternativa código abierto que realiza estas operaciones y justamente está disponible para ser empleada como “objeto”.

Finalmente, el código fuente también se escribió con el programa Notepad++, el cual análogamente fue empleado para la interfaz gráfica (HTML y CSS).

3.6. Modelamiento de la base de datos

La base de datos, escrita en lenguaje SQL y empleada mediante el sistema relacional de SQLite tiene la siguiente estructura:

A. Tabla “operaciones” conteniendo en 15 campos los siguientes datos:

- Identificadores de operación
 1. Id de operación
 2. Fecha de operación
- Parámetros propuestos y reales de operación.
 1. Fundente
 2. Eje
 3. Escoria
 4. Enriquecimiento
- Flujos del sistema
 1. Concentrado
 2. Circulante

3. Flotación
4. Carbón
5. Salida de polvos

B. Tabla “operación_nueva” conteniendo lo mismo de la tabla ‘operaciones’ pero con los datos de la nueva operación propuesta.

C. Tabla “parámetros_límite” conteniendo en 4 campos los siguientes rangos porcentuales de valores límites:

- Cobre en el Eje y la Escoria
- Fe_3O_4 en el Eje y Escoria
- SiO_2 en la Escoria
- Enriquecimiento y Oxígeno técnico

La tabla “operaciones” se emplea para almacenar las operaciones en el historial y permitir el cálculo de los parámetros de ajuste con la “operación_nueva”. Por otro lado, la tabla “parámetros_límite” contiene los valores máximos y mínimos de los parámetros principales.

El sistema SQLite tiene la particularidad de estar integrado en PHP, es decir, permite que el archivo de base de datos (denominado “db_programa.sqlite”) esté disponible en la raíz del programa. Esto es importante ya que tiene la gran ventaja de hacer más simple la ejecución del programa como una aplicación de escritorio.

Sobre los parámetros, la tabla 3.3 muestra los valores límites que vendrán por defecto en el programa. Estos límites fueron confeccionados tomando en cuenta los valores operacionales usuales del Convertidor Teniente.

Tabla 3.3. Valor de los parámetros límite por defecto del programa.

	Parámetro	Mínimo, %	Máximo, %
<i>Escoria</i>	Cu	1	6
	SiO ₂	20	35
	Fe ₃ O ₄	8	25
<i>Eje</i>	Cu	60	75
	Fe ₃ O ₄	1	5
<i>Aire de proceso</i>	Enriquecimiento	21	40
	O ₂ Técnico	100	100

3.7. Ingreso de información y validación de datos

Dado que es común errar al momento de insertar datos manualmente, la validación de datos corresponde a una de las características de valor agregado en el programa desarrollado. Los tipos de errores considerados son dos:

1. Caracteres no admitidos, los cuales corresponden a cualquier ingreso de dato que no sea numérico.
2. Ingreso de valores porcentuales sobre 100%.

Gran parte de la validación de datos comienza por corregir al usuario antes de solicitar el envío de esta información, de este modo el programa cuenta con alertas mientras el usuario escribe (figura 3.2a), lo cual evita que el usuario envíe datos con caracteres no admitidos y cuando se ingresan porcentajes sobre 100%.

Adicionalmente se tomó en cuenta el problema del símbolo decimal, el programa tiene la capacidad de entender de la misma manera el uso de punto o coma para los decimales, lo cual es un dolor de cabeza menos para el operador. El resultado es una validación que sirve tanto para ayudar al operador como para que el programa no comprometa la integridad del cálculo realizado.

3.8. Comprobación de los resultados obtenidos

Para poder verificar la validez del cálculo realizado por el programa, se procedió con el balance de masa en una hoja de Microsoft Excel y se compararon los resultados obtenidos con los que entregó el programa realizado. De este modo, variando los datos de entrada se encontró que el programa efectivamente estaba realizando correctamente los cálculos necesarios ya que los resultados obtenidos eran exactamente los mismos.

3.9. Recursos externos empleados

A continuación se presentan los recursos externos empleados, indicando el tipo de recurso y su uso en el programa. Estos recursos se incluyen en el material complementario entregado en el soporte digital de esta memoria.

➤ **matriz.php**^[36]

Es un archivo PHP que contiene las instrucciones necesarias para que PHP realice operaciones matemáticas con matrices.

³⁶ <http://www.phpclasses.org/package/2859-PHP-Perform-operations-with-matrices.html>

➤ **browser_class_inc.php**^[37]

Es un archivo PHP que contiene las instrucciones necesarias para que PHP reconozca el navegador web empleado con el usuario, de este modo el programa puede determinar si el navegador usado es compatible.

➤ **css_browser_selector.js**^[38]

Corresponde a un archivo JavaScript que permite alterar el CSS según el navegador empleado. Es necesario debido a que el HTML y CSS tienen leves diferencias de interpretación entre un navegador y otro.

➤ **Plugins para jQuery**^[39]

jQuery es un *framework* (entorno de desarrollo) que permite una mayor facilidad para escribir instrucciones en JavaScript. jQuery se usó en conjunto con *plugins* (complementos) los cuales se listan a continuación:

1. jquery.validationEngine.js permite validar al instante los datos proporcionados por el usuario (Fig. 3.2a).
2. jquery.calculation.js realiza un cálculo al instante para las composiciones, permite que el usuario pueda ver al instante la cantidad porcentual de composiciones calculables como la diferencia para obtener 100%.
3. jquery.bt.js permite mostrar un pequeño elemento emergente al costado de los parámetros de nueva operación (Fig. 3.2b).
4. jquery.jgrowl.js permite mostrar los avisos de acción (Fig. 3.3).

³⁷ <http://phpclasses.nlared.com/package/2827-PHP-Detects-the-user-browser-type-and-version.html>

³⁸ http://rafael.adm.br/css_browser_selector

³⁹ <http://jquery.com/>

➤ **Recursos gráficos menores**

Se emplearon íconos de dominio público para el menú principal. También se emplearon los íconos de cada navegador para el mensaje de alerta por navegador no compatible.

CAPÍTULO IV: RESULTADOS

4.1. Archivos de programa

El programa realizado se puede resumir como 63 archivos distribuidos en 6 carpetas y con un tamaño combinado de 377 Kilobits. La tabla 4.1 contiene un detalle con estos distintos tipos de archivos.

Tabla 4.1. Tipos de archivo del programa.

Tipo de archivo	Cantidad de archivos	Tamaño combinado, Kilobits
Imagen	30	53
PHP Script	21	130
JavaScript	7	172
Documento CSS	2	20
Icono	1	1.4
.htaccess	1	1
Base de datos SQLite	1	1

Los archivos del tipo “Imagen” son usados en conjunto con los archivos “Documento CSS” y “JavaScript” para la representación gráfica de la interfaz.

Los archivos “PHP Script” son los que realizan el cálculo, reciben las entradas y generan los resultados. Junto con esto, es PHP mediante una librería quien accede a la base de datos contenida en el archivo SQLite.

Finalmente, el archivo “.htaccess” contiene reglas de redirección que permite que el programa tenga URL amigables o semánticas^[40]. De este modo,

⁴⁰ http://en.wikipedia.org/wiki/Semantic_URL

en la barra de navegación del usuario se muestra amigablemente la ruta de acceso a una determinada sección.

4.2. Base de datos

La base de datos cuenta con 3 tablas (descritas en la sección 3.6). La estructura de la tabla “operaciones” y “operación_nueva” se muestra en la figura 4.1 mientras que la estructura de la tabla “parámetros_límite” se muestra en la figura 4.2.

Campo	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Extra
<u>id</u>	int(11)			No	None	AUTO_INCREMENT
fecha_tiempo	datetime			No	None	
concentrado	longtext	utf8_general_ci		No	None	
circulante	longtext	utf8_general_ci		No	None	
flotacion	longtext	utf8_general_ci		No	None	
carbon	longtext	utf8_general_ci		No	None	
polvos	longtext	utf8_general_ci		No	None	
par_fundente	longtext	utf8_general_ci		No	None	
real_fundente	longtext	utf8_general_ci		No	None	
par_escoria	longtext	utf8_general_ci		No	None	
real_escoria	longtext	utf8_general_ci		No	None	
par_eje	longtext	utf8_general_ci		No	None	
real_eje	longtext	utf8_general_ci		No	None	
par_aire	longtext	utf8_general_ci		No	None	
real_aire	longtext	utf8_general_ci		No	None	

Figura 4.1. Estructura de las tablas “operaciones” y “operación_nueva”. En la nomenclatura se ha ignorado el empleo de tildes por conveniencia.

Campo	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Extra
<u>id_parametro</u>	int(11)			No	None	AUTO_INCREMENT
parametro	text	utf8_general_ci		No	None	
min	float			No	None	
max	float			No	None	

Figura 4.2. Estructura de la tabla “parámetros_límite”. En la nomenclatura se ha ignorado el empleo de tildes por conveniencia.

Sobre tamaño, la base de datos cuenta con un mínimo de 2.2 Kilobits que corresponden a los registros de los parámetros límite. El tamaño de la base de datos dependerá desde luego de la cantidad de información que almacene, la cual estará relacionada con cada nueva operación realizada. Cada operación le significa cerca de 1 kilobyte adicional a la base de datos. Tomando esto en cuenta, se necesitarían 1024 operaciones para sumar 1 MB en la base de datos.

4.3. Requerimientos para instalar y usar el programa

En orden de poder emplear el programa se necesita cumplir con un pequeño número de condiciones. Los siguientes son los requerimientos mínimos para poder poner en marcha el programa:

1. Servidor HTTP Apache 1.3 (o algún servidor con soporte PHP)
2. PHP versión 5.0.0
3. SQLite versión 2.0 (incluido en PHP 5)
4. 70 MB de espacio disponible en disco.

Estos requerimientos son bastante simples de concretar ya que cualquier computador de incluso hace 10 años atrás los podría satisfacer. Los requerimientos son tan simples que hasta es posible llevar el programa en un disco USB o *pendrive* y ejecutarlo desde ahí.

Estrictamente el programa necesita tan sólo 391 kilobits (incluyendo la base de datos y archivos de imágenes) pero en rigor el requerimiento asociado tanto a Apache como a PHP hace que el espacio en disco requerido sea del orden de 70 Megabits, lo cual aún es poco.

4.4. Distribución del programa

El programa estará distribuido de tal manera de poder ser empleado en tres entornos o condiciones:

- A. Instalable en el computador del usuario (Windows)
- B. Ejecutable sin instalación desde un disco USB (Windows)
- C. Script montable en un servidor HTTP PHP (Multiplataforma)

La facultad de ser instalado mediante un asistente o ser ejecutado desde un disco USB es posible gracias al empleo del programa gratuito Server2Go^[41], el cual permite ejecutar los requerimientos fundamentales del programa (Apache y PHP) y además contar con un navegador web incorporado. Por otro lado, el instalador es nada más que la versión portable con un asistente que permite la instalación del programa en el disco duro, permitiendo seleccionar la carpeta de destino, la inclusión de accesos directos y un desinstalador. La creación del instalador es posible gracias al programa código abierto InstallJammer^[42]. La tabla 4.2 contiene un cuadro comparativo con los tipos de versiones y su tamaño.

Tabla 4.2. Comparación de tamaños entre versiones del programa.

Versión	Tamaño distribuido	Tamaño en disco
Script montable	391 KB	391 KB
Portable (disco USB)	99.4 MB	99.4 MB
Instalable	42.8 MB	99.4 MB

⁴¹ <http://www.server2go-web.de/>

⁴² <http://www.installjammer.com/>

Cabe mencionar que las distintas versiones del programa sólo se refieren al paquete distribuido, el programa en sí es el mismo para cada versión. La diferencia entre versiones radica en la disposición de los requerimientos señalados en la sección 4.3.

Como nota adicional, los requerimientos tanto de Servidor HTTP (Apache y PHP) suman aproximadamente 70 MB, el espacio adicional requerido para las versiones portable e instalable se debe a que estas versiones traen incorporado un navegador web portable (Firefox Portable), el cual corre directamente sin instalación y ocupa el espacio restante para sumar 99.4 MB.

4.5. Capturas de pantalla

A continuación se presentan las capturas de pantalla del programa, obtenidas mediante la versión “Script montable” instalada en un computador remoto con Red Hat Linux y accedido mediante Firefox en Windows 7.



Figura 4.3. Vista de la pantalla principal. La parte superior muestra el menú de pestañas, en el centro se aprecia la inserción de los parámetros de proceso.

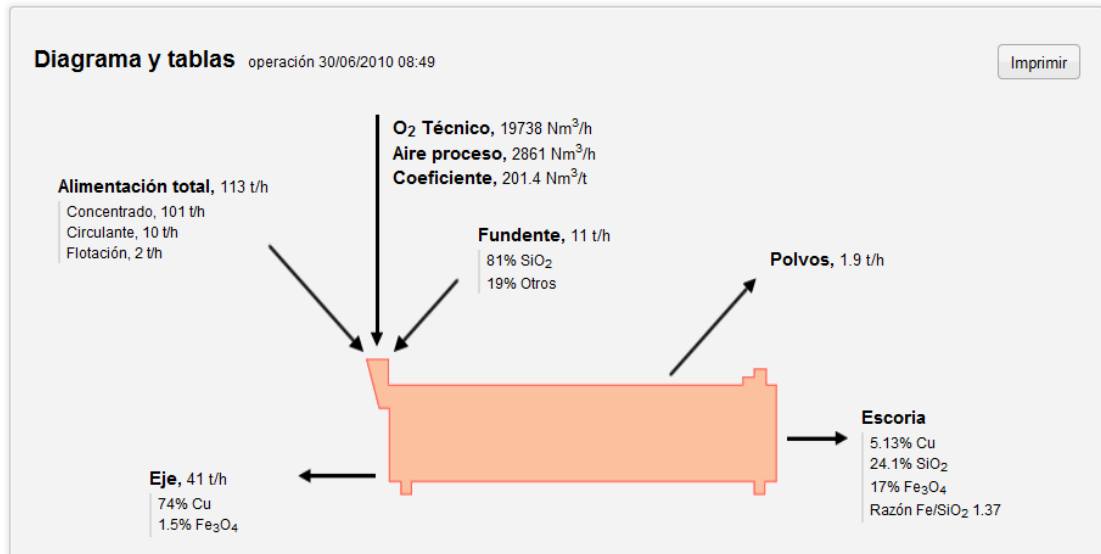


Figura 4.4. Captura de la sección “Diagrama y tablas”. Se aprecia un diagrama simplificado y la opción de imprimir.

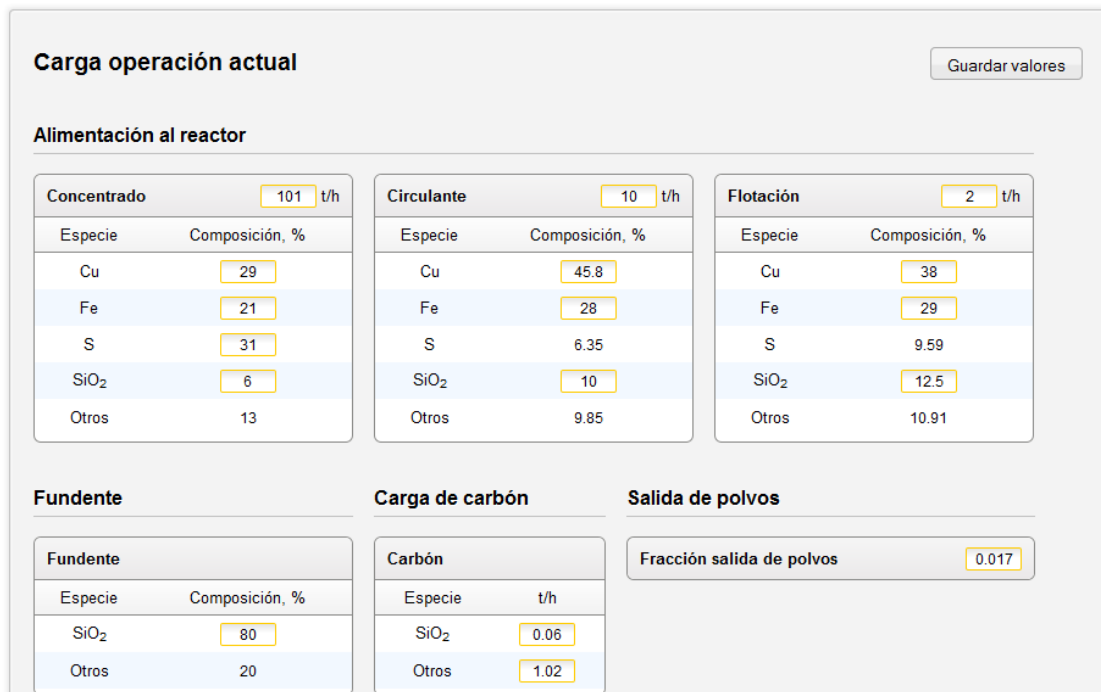


Figura 4.5. Vista parcial de la sección “Carga actual”. Se aprecia la caracterización de flujos y la opción de guardar. Vista válida también para “Carga nueva”.

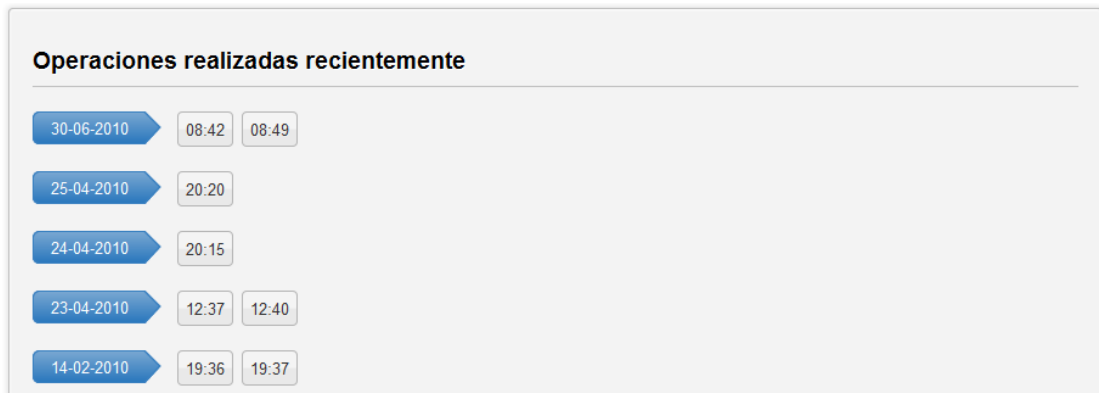


Figura 4.6. Vista parcial principal del “Historial”. Se aprecia un listado organizado por fecha y hora de las operaciones realizadas recientemente.

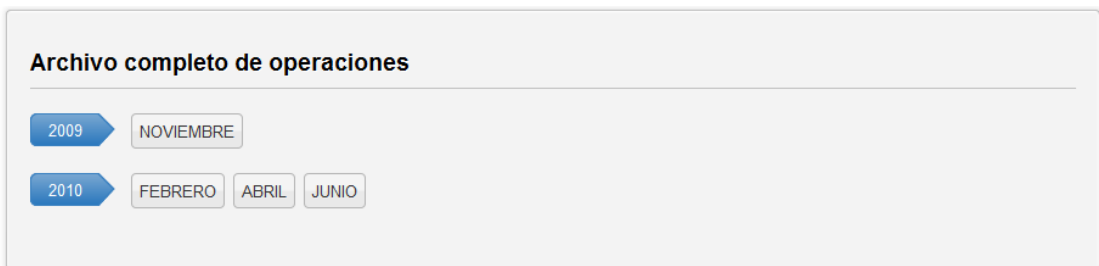


Figura 4.7. Vista de la sub-sección “Archivo completo” de la sección “Historial”. Se aprecia una división anual y mensual de las operaciones.

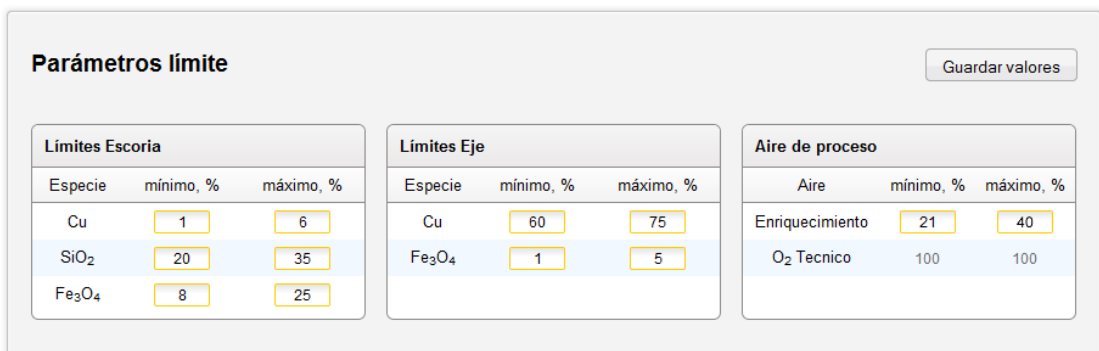


Figura 4.8. Captura parcial de sección “Parámetros límite”. Se aprecia una categorización de los parámetros de operación y la opción de guardar cambios.

NAVEGADOR INCOMPATIBLE

El navegador **Internet Explorer 6.0** es incompatible con este programa, actualice su navegador a una versión más reciente o puede bajar gratuitamente uno de los siguientes navegadores:



Firefox

Disponible para Windows, Mac OS X y Linux
<http://www.mozilla.com/en-US/firefox/firefox.html>



Internet Explorer

Disponible solamente para Windows
<http://windows.microsoft.com/es-es/windows/downloads>



Google Chrome

Disponible para Windows, Mac OS x y Linux
<http://www.google.com/chrome/>

Figura 4.9. Vista de mensaje de navegador incompatible. El programa indica el navegador empleado y ofrece cinco alternativas (Firefox, Internet Explorer, Google Chrome, Safari y Opera)

JAVASCRIPT NO HABILITADO EN NAVEGADOR

El programa necesita JavaScript para poder operar correctamente. Siga estas instrucciones para habilitar JavaScript en **Firefox 3.6.6** Puede buscar [documentacion en linea](#) al respecto.



Instrucciones para habilitar JavaScript en Firefox 3.6.6

1. En la barra de menú, haga click en **Herramientas** luego en **Opciones**.
2. En Opciones, seleccione la pestaña **Contenido**.
3. Seleccione **Habilitar Javascript**.
4. Haga click en **OK** o **Aceptar**.
5. Cierre y vuelva a abrir Firefox.

Figura 4.10. Vista de mensaje de JavaScript no habilitado. En este caso indica las instrucciones para habilitarlo en Firefox 3.6.6

4.6. Esquema de funcionamiento

La figura 4.11 muestra de manera resumida el esquema de funcionamiento del programa realizado.

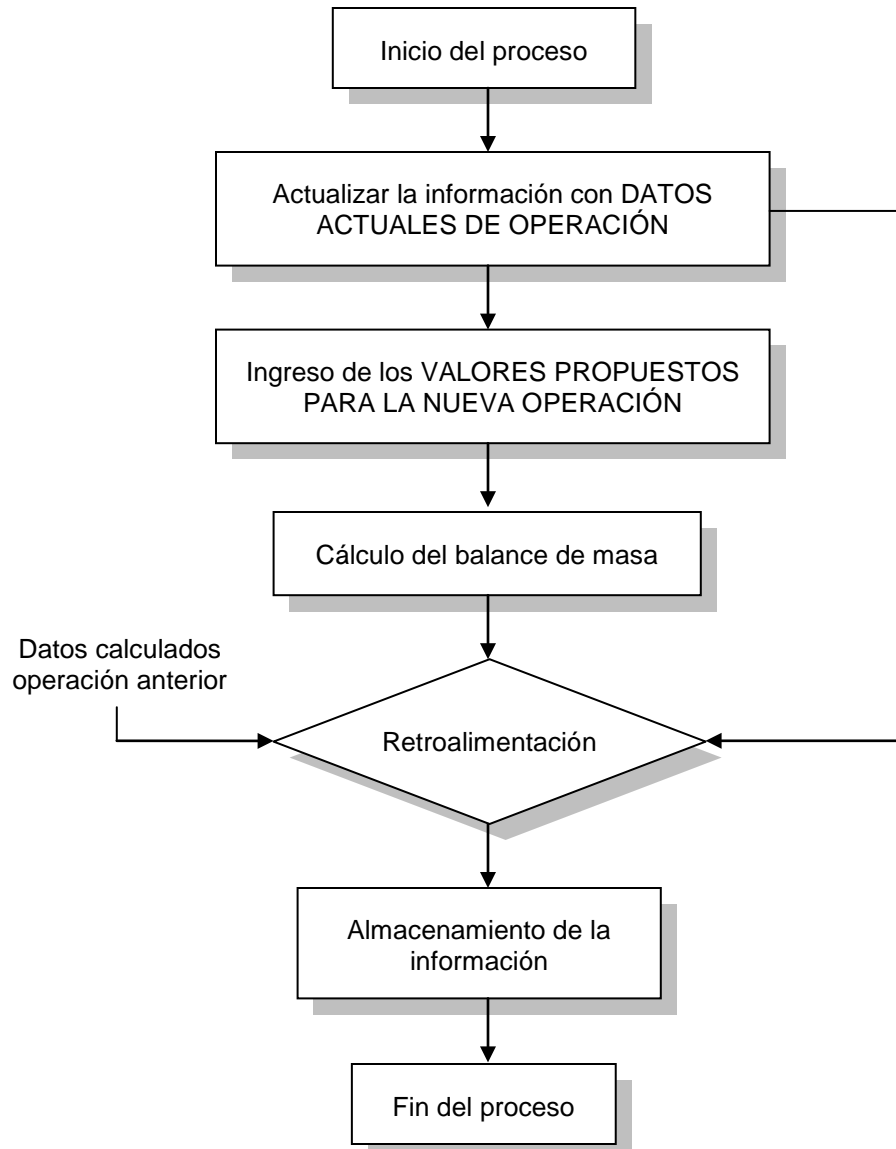


Figura 4.11. Esquema del funcionamiento del programa. Se aprecia como el programa toma los datos operacionales para efectuar la retroalimentación para ayudar en la toma de decisiones.

4.7. Tiempo y costo de desarrollo

El tiempo total de desarrollo fue de 8 semanas, incluyendo la etapa de selección de tecnologías. La tabla 4.1 muestra un detalle con la cantidad de semanas empleadas en cada etapa.

Tabla 4.3. Tiempo empleado en cada etapa de desarrollo.

Etapa de desarrollo	Tiempo, semanas
Selección de tecnologías	1
Diseño y programación de interfaz	1
Programación del código fuente	5
Pruebas y creación del instalador	1

Tomando en cuenta el tiempo de desarrollo de siete semanas al excluir la semana empleada en la selección de tecnologías y que el costo de emplear un programador PHP con un año de experiencia es de aproximadamente \$350.000.- pesos, el costo de desarrollo sería de unos \$612.500.- pesos considerando el Back-end y pruebas de funcionamiento. Por otro lado, el costo asociado por diseñar y aplicar una interfaz de este tipo es del orden de \$50.000.- o incluso menos ya que HTML, CSS y JavaScript son lenguajes simples y altamente extendidos. Finalmente, el costo final de este programa es de unos \$662.500.- pesos.

CAPÍTULO V: DISCUSIONES

Si bien es cierto uno de los puntos débiles del modelo de desarrollo conocido como código abierto corresponde a que la mayoría de las soluciones de este tipo carecen de las extensas características presentes en sus contrapartes y por lo tanto, describen un desarrollo incompleto o inmaduro, hay que tener en cuenta que existen proyectos código abierto con un enorme respaldo y años de experiencia, que incluso los hicieron atractivos para su posterior millonaria adquisición por otras empresas, por ejemplo la adquisición de MySQL por Sun Microsystems^[43]. Teniendo en cuenta que la selección de tecnologías implica un amplio espectro de proyectos código abierto en la forma de lenguajes de programación, sistemas de base de datos, etc., la selección no tan sólo debe tomar en cuenta los requerimientos del cálculo ni la rapidez del desarrollo, sino que tanto más importante es la madurez y extensión de dichos recursos ya que esta consideración facilita enormemente el desarrollo dado que se confía en proyectos que tienen mayor experiencia, mayor respaldo y por sobre todo mayor seriedad. Esto deja como antecedente y punto de partida un claro norte: Buscar las tecnologías que ofrezcan una probada madurez ya que esto afecta enormemente la posteridad o vida útil del programa a desarrollar.

Tomando en cuenta que la selección privilegia la madurez y proyectos altamente extendidos, se tiene que este filtro reduce el espectro de lenguajes informáticos a unos 15 candidatos. La posterior selección comienza a considerar factores económicos ya que en esta altura se podría decir que todos los candidatos tienen excelentes antecedentes de madurez y desarrollo. Considerando someramente el salario promedio mensual de un programador especializado en cada lenguaje, se llega a que la mayor economía la

⁴³ Cfr. Capítulo 3, Sección 3.3.3. p. 65

representan los lenguajes de lado de servidor ya que para este tipo de programación se habla de costos del orden de \$350.000.- a \$450.000.- versus la programación de escritorio donde los costos son del orden de \$600.000.- a \$750.000.- De este modo, considerando los lenguajes de lado de servidor se reducen los candidatos a tan sólo 4 lenguajes de programación, los cuales si bien es cierto son equivalentes entre sí, resulta claramente notable la superior extensión de PHP en términos de desarrollo, actividad y comunidad de desarrolladores. Factores derivados principalmente de la potencia y simpleza de este lenguaje de programación, lo cual se debe principalmente a tres razones:

1. PHP soporta más de un tipo de programación, permitiendo combinar distintos algoritmos y una mayor flexibilidad para programar.
2. La extensión de PHP es alta, lo cual implica una mayor cantidad de recursos y de colaboradores.
3. Su aplicación multiplataforma depende sólo del interpretador, lo cual permite su instantánea aplicación en diversos sistemas operativos.

La facilidad de programación de PHP es la que a su vez lo hace altamente extendido. De este modo, desarrollar en PHP no solamente es fácil sino que también más económico ya que por un lado la programación en lenguajes de lado de servidor es más económica, en el caso de PHP hay una gran cantidad de recursos disponibles para trabajar con él debido a su gran extensión. Una cuarta razón serían las capacidades de red de PHP, lo cual le permite ser una alternativa lista para trabajar en estos entornos. Otros lenguajes como Java o C++ requieren crear instrucciones extra para la comunicación con el servidor, lo cual implica la creación de un programa que se dedica a comunicarse con computador cliente para poder hacer lo mismo que PHP sabe hacer desde siempre. Si bien es cierto PHP ofrece muchas características atractivas, es importante dejar en claro que nunca será un lenguaje tan robusto

como por ejemplo lo es C++, dado que PHP se creó y se ha desarrollado para la realización de páginas web, una realidad muy distinta a la de C++. PHP es un lenguaje que se opera mediante la interpretación de instrucciones mediante un interpretador, lo cual dista bastante de la ejecución de un programa compilado. PHP satisface enteramente los requerimientos necesarios para el problema abordado, pero si se desea hacer una compleja modelación en 3D o algo de esos calibres, los lenguajes de programación de escritorio son la correcta elección sin duda alguna. No obstante esta diferencia, PHP y en general todos los lenguajes de lado de servidor están aptos para satisfacer los requerimientos de cálculo no tan sólo para el problema abordado sino que para una serie de problemas y necesidades mayores.

Tomando en cuenta que la programación en PHP es simple y flexible, se podrían extender sin mayor problema las funciones demostrativas de este programa e incorporar nuevas características tales como la integración con sistemas de control automático, gestión y administración a distancia, etc. Por otro lado, la aplicación de PHP para la solución de diversos problemas particulares está más que demostrada ya que la mayoría de servicios disponibles a través de internet han sido creados empleando el lenguaje PHP, desde programas para crear blogs hasta servicios para subir videos y compartirlos en línea.

Un desafío en este proyecto fue la realización de una intuitiva y amigable interfaz de usuario, la cual tiene como objetivo básico ser el nexo entre las instrucciones del código fuente y el operador. Pero si se busca ir más lejos, la interfaz es una condición básica para el éxito de un programa y si es agradable, intuitiva y simple, el operador no se preocupará de recordar cómo usar el programa, ni necesitará una extensa capacitación y en resumen, usar la herramienta no sería una molestia o una frustración para él. Esto es importante

ya que es muy probable que un operador trabajando de mala gana haga mal su trabajo, lo cual se traduce en un costo para la empresa. El valor agregado también consistió en la aplicación de una herramienta absolutamente inédita, exploración que sin lugar a dudas fue una de las tareas más gratificantes de este proyecto, ya que no tan sólo hablamos de código abierto sino que también de un programa de lado de servidor, de aplicación lista para trabajar en red, de características multiplataforma, etc. En resumen, se exploró bastante y se buscó ir más lejos de lo comúnmente establecido.

El código abierto fue una decisión trascendental y marcó todo el desarrollo del programa ya que la cantidad de recursos disponibles y de alternativas diferentes al momento de buscar una solución a un problema particular fue francamente enorme, en reiteradas ocasiones había más de diez opciones a elegir al momento de seleccionar un recurso, lo cual es muy útil ya que este surtido de alternativas ayuda a una mayor selectividad, junto con ésto, todo este abanico de recursos se traduce en mejor calidad de los mismos. Por otro lado, el uso de estos recursos ayuda a disminuir el tiempo y costo de desarrollo ya que no se tienen que programar todas las instrucciones realmente necesarias y se pueden usar rápidamente las instrucciones provistas por estos recursos. Por otro lado, el uso de programas gratuitos para el diseño de la interfaz y la escritura del código fuente también implica ahorro de costos, por ejemplo el valor de programas para desarrollar como Microsoft Visual Studio^[44] (US \$799 – US \$11,869) o Adobe Dreamweaver^[45] (US \$399) o Adobe Illustrator^[46] (US \$599) queda descartado y por ende el costo de desarrollo se minimiza.

⁴⁴ <http://www.microsoft.com/visualstudio/en-us/products/>

⁴⁵ <http://www.adobe.com/products/dreamweaver/>

⁴⁶ <http://www.adobe.com/products/illustrator/>

Si bien es cierto el conjunto de tecnologías seleccionadas y empleadas consiguen efectuar exitosamente todas las operaciones matemáticas necesarias para realizar el balance de masa, esto requirió que el lenguaje empleado adquiriera instrucciones para realizar tareas tan comunes como una operación matemática con matrices, lo cual es una realidad distinta a lo que se tiene en un documento en Microsoft Excel, donde existe un gran catálogo de funciones y operaciones matemáticas listas para usar. Esto deja de manifiesto que el desarrollo de una herramienta de este tipo es lógicamente más complicado que realizar una planilla en Excel, pero es mucho más flexible. Según los resultados se puede apreciar que el uso y aspecto del programa es bastante más amigable que una alternativa análoga realizada en Excel, pero el programa desarrollado requiere más tiempo de desarrollo, conocimientos avanzados en programación PHP, HTML, CSS, JavaScript, base de datos y diseño de interfaz gráfica. Comparando costos, el programa desarrollado tiene un costo estimado de \$662.500.- pesos en 7 semanas de desarrollo, mientras que una solución alternativa en Excel claramente toma menos tiempo, quizás 4 o incluso tan sólo 3 semanas y un costo aproximado de \$500.000.- sin tomar en cuenta el costo base de la licencia de Office (US \$499 para la versión Professional 2010) de donde se desprende que el mayor inconveniente de una solución desarrollada en Excel es que depende de Excel, haciendo necesario contar con este programa tanto para el desarrollador como para el cliente lo que significa un costo base en ambos casos. Si bien es cierto el programa desarrollado también tiene requerimientos básicos, éstos son completamente gratuitos. Se tiene entonces que la herramienta desarrollada es un 33% más cara y toma prácticamente el doble de tiempo, pero cuenta con características únicas y propias de un desarrollo completamente a medida.

Si se analiza someramente el costo de desarrollar un programa de este tipo en otros lenguajes como lo serían Java, C++ y C#, ahorrando los costos

asociados a los programas empleados para desarrollar y empleado recursos código abierto, se tendría que el tiempo de desarrollo sería menor ya que estos lenguajes de programación emplean entornos de desarrollo para la creación de la interfaz, los cuales en algunos casos son gratuitos y de muy buena calidad. La comparación se realiza tomando en cuenta que estos lenguajes también son altamente conocidos y que otorgarían las mismas o similares cualidades del programa. Se tiene que el tiempo promedio sería de 6 semanas en estos lenguajes y que el costo dependería del lenguaje empleando, donde el programa hecho en Java tendría un costo de desarrollo de \$1.045.000.- y por otro lado el costo de desarrollo en C++/C# sería de \$970.000.- Estimaciones que se hacen tomando en cuenta el salario aproximado de un programador con un año de experiencia, incluyendo el costo de diseño de la interfaz y minimizando costos asociados a programas y recursos empleados. Un desarrollo en Visual Basic implicaría perder las características multiplataforma ya que Visual Basic es sólo para Windows y tendría un costo adicional asociado a la licencia de Visual Studio (US \$799 – US \$11,869), programa necesario para desarrollar en Visual Basic. Por otro lado, lenguajes no tan conocidos y por ende más limitados de oferta laboral implicarían un mayor costo debido a la exclusividad que estos representan.

Si bien la realización de este trabajo implicó el conocimiento de la pirometalurgia del proceso, hay que tener en cuenta que la otra mitad del trabajo consistió en la práctica de disciplinas asociadas a la programación informática. Integración deja de manifiesto la posibilidad de extender las capacidades del ingeniero metalúrgico, haciendo del mismo un profesional con mayores conocimientos y habilidades, lo cual obviamente es un gran beneficio para su formación profesional. El hecho de que estos conocimientos fuesen desarrollados aprovechando los recursos educativos que ofrece internet da aún más trascendencia ya que hablamos de una educación al alcance de todos.

CAPÍTULO VI: CONCLUSIONES

El amplio espectro de tecnologías informáticas disponibles implica que para el desarrollo de una herramienta de este tipo es recomendable la realización de un análisis respecto del problema a enfrentar, permitiendo así la correcta selección del conjunto de tecnologías informáticas a emplear privilegiando la madurez de las mismas.

La selección condujo al uso del lenguaje informático PHP dado que económicamente es el más atractivo y además es ampliamente usado. Junto con esto, PHP ofrece interesantes características derivadas de la programación de lado de servidor.

Las funciones demostrativas de este programa son extensibles a una mayor complejidad de operación y desde luego a otros problemas y situaciones, sean relacionados o no con el proceso de fusión.

Las características propias del programa que le otorgan valor agregado, permiten ofrecer una herramienta más atractiva y por consecuencia más competitiva. Lo cual es un antecedente importante al momento de seleccionar o desechar una herramienta determinada.

El uso de recursos código abierto y de programas gratuitos permiten minimizar costos asociados al desarrollo de la herramienta, ya que los recursos código abierto permiten disminuir tanto tiempo como dificultad de desarrollo y los programas gratuitos implican un gasto menos.

El desarrollo de una herramienta de este tipo requiere al menos el doble de tiempo necesario y presenta mayor dificultad versus de una solución desarrollada en Excel, haciendo que el costo de desarrollo sea un 20% más caro. Pero el programa desarrollado presenta un mejor aspecto, es más simple de usar y posee sistemas de validación e integridad de datos, haciendo más competitiva esta diferencia.

Comparando el costo de desarrollo de la herramienta creada versus el eventual costo en otros lenguajes de programación, se tiene que la herramienta creada ofrece un ahorro del orden del 32% si se consideran los lenguajes informáticos C++/C#, por otro lado respecto de Java el ahorro es del orden de un 37%. Comparación que toma en cuenta que se minimizan costos en ambos casos y se usan recursos código abierto.

Abordar la solución de un problema particular de la metalurgia mediante la realización de un programa informático implica un conocimiento extra para el ingeniero metalúrgico, lo cual le otorga mayor valor a su formación profesional y lo integra más con su entorno, sin que este conocimiento sea necesariamente consecuencia una formación profesional anexa.

Finalmente, la herramienta desarrollada cumple con el objetivo propuesto, lo cual indica que el código abierto y el conjunto de tecnologías informáticas seleccionadas son aplicables satisfactoriamente al problema propuesto. Se probó la aplicación de PHP en un problema no tradicional y se obtuvo que ofrece un ahorro significativo si se considera un desarrollo y aplicación no dependiente de otro programa como lo es el caso de Excel.

CAPÍTULO IV: REFERENCIA BIBLIOGRAFICA

Arzion (2009), 10 Razones para usar Open Source.

<http://www.arzion.com/empresa-de-internet/posts/10-razones-para-usar-Open-Source>

Barros Alejandro (2009), Uso de Software Libre en el Sector Público.

<http://www.alejandrobarrros.cl/content/view/561344/>

Bravo Correa J. C.; “Metodología de Cálculo para la determinación de los Parámetros de Operación del Horno Flash”; Universidad de Santiago, Facultad de Ingeniería, 1997.

Causa Emiliano (2006), Tipos de programación.

http://www.proyecto-biopus.com.ar/emiliano/catedras2006/inter1/clase_prog_1.html

Davenport W. G.; King M.; Schlesinger M.; Biswas A. K.; “Extractive Metallurgy of Copper”; Editorial Pergamon; Cuarta Edición, 2002.

Evaluando ERP (2009), Open Source ¿Es una opción para las empresas?.

<http://www.evaluandoerp.com/nota-606-Open-Source-Es-una-opcion-para-las-empresas.html>

González Letelier A.; “Riquezas Minerales de Chile a Nivel Mundial”, Universidad de Chile, 2000.

McKendrick J.; “Open Source in the Enterprise: New Software Disrupts the Technology Stack”; IOUG, Septiembre de 2007.

Open Source Unleashed (2007), Successful commercial open source.
http://alex Fletcher.typepad.com/all_bets_off/2007/01/successful_comm.html

Redard G.; "Nociones Básicas y Glosario de Balance Metalúrgico"; COPPER-COBRE, Octubre de 2007.

Sun Microsystems, Inc; "Open Source in the Enterprise: Fulfilling the Promise"; Sun Microsystems, Julio de 2009.

SYS-CON MEDIA (2005), Dual-Licensing Open Source Business Models.
<http://linux.sys-con.com/node/49061>

Tekla S.; Voelcker J.; "Of mice and menus: Designing the user-friendly interface"; IEEE Spectrum, Septiembre de 1989, p. 46-51. Disponible en:
<http://www.guidebookgallery.org/articles/ofmiceandmenus>

Wikipedia (2010), Código abierto. http://es.wikipedia.org/wiki/Codigo_abierto

Wikipedia (2010), Software Libre. http://es.wikipedia.org/wiki/Software_libre

Wikipedia (2010), Commercial open source applications.
http://en.wikipedia.org/wiki/Commercial_open_source_applications

CAPÍTULO VIII: APÉNDICE

8.1. APÉNDICE A: Cálculo de la cantidad de Eje, Escoria y Fundente

8.1.1. Composición del Eje

Para poder resolver el sistema de ecuaciones 4x4 (ecuación 3.9), es necesario conocer previamente algunos porcentajes de los elementos y compuestos que forman parte del Eje. Las tablas 8.1 y 8.2 indican la composición porcentual de elementos y compuestos que forman parte del Eje.

Tabla 8.1. Elementos que forman parte del Eje.

ELEMENTOS	%
1.- Cu	74.00
5.- Fe	4.79
6.- Oxígeno	0.42
7.- S	20.79
TOTAL	100.00

Tabla 8.2. Compuestos que forman parte del Eje.

COMPUESTOS	%
2.- Cu ₂ S	92.67
3.- Fe ₃ O ₄	1.50
4.- FeS	5.83
TOTAL	100.00

El orden en que se determina cada uno de los elementos de las tablas 8.1 y compuestos de la tabla 8.2, es de acuerdo a la numeración entregada en las tablas mencionadas. A continuación se detalla el cálculo de cada elemento y compuesto de acuerdo al orden establecido:

1. La ley del eje es un parámetro de operación con el cual se desea operar el horno.
2. Al conocer la ley de Eje se puede determinar el porcentaje de Cu_2S .

$$\% \text{Cu}_2\text{S}_{\text{EJE}} = \frac{(\text{Ley del Eje}) \times \text{Pm}(\text{Cu}_2\text{S})}{2 \times \text{Pa}(\text{Cu})} \quad (\text{Ec. 8.1})$$

3. Al igual que la ley del eje, el porcentaje de magnetita es un parámetro de operación con el cual se desea operar el horno.
4. Dado que se conocen los porcentajes de todos los compuestos del Eje, menos el de FeS , éste se determina a través de la siguiente expresión.

$$\% \text{FeS}_{\text{EJE}} = 100 - \% \text{Cu}_2\text{S}_{\text{EJE}} - \% \text{Fe}_3\text{O}_{4\text{EJE}} \quad (\text{Ec. 8.2})$$

5. El porcentaje de Hierro se calcula de la siguiente manera:

$$\% \text{Fe}_{\text{EJE}} = \% \text{FeS}_{\text{EJE}} \times \frac{\text{Pa}(\text{Fe})}{\text{Pm}(\text{FeS})} + \% \text{Fe}_3\text{O}_{4\text{EJE}} \times \frac{3 \times \text{Pa}(\text{Fe})}{\text{Pm}(\text{Fe}_3\text{O}_4)} \quad (\text{Ec. 8.3})$$

6. El porcentaje de Oxígeno en el Eje se obtiene se la siguiente manera:

$$\% \text{Oxig}_{\text{EJE}} = \frac{4 \times \text{Pa}(\text{O})}{\text{Pm}(\text{Fe}_3\text{O}_4)} \quad (\text{Ec. 8.4})$$

7. El porcentaje de Azufre se obtiene se la siguiente manera:

$$\% \text{S}_{\text{EJE}} = \% \text{FeS}_{\text{EJE}} \times \frac{\text{Pa}(\text{S})}{\text{Pm}(\text{FeS})} + \% \text{Cu}_2\text{S}_{\text{EJE}} \times \frac{\text{Pa}(\text{S})}{\text{Pm}(\text{Cu}_2\text{S})} \quad (\text{Ec. 8.5})$$

8.1.2. Composición de la Escoria

Al igual que la composición del Eje, el sistema de ecuaciones 4x4 de la ecuación 3.9 requiere conocer los porcentajes de elementos y compuestos que forman parte de la escoria. Las tablas 8.3 y 8.4 indican estas composiciones.

Tabla 8.3. Elementos que forman parte de la Escoria.

ELEMENTOS	%
1.- Cu	5.14
5.- Si	11.27
9.- Fe	34.51
10.- Oxígeno	23.90
3.- S	1.30
7.- Otros	23.88
TOTAL	100.00

Tabla 8.4. Compuestos que forman parte de la Escoria.

COMPUESTOS	%
2.- Cu_2S	6.44
4.- SiO_2	24.10
6.- Fe_3O_4	17.00
8.- FeO	28.58
7.- Otros	23.88
TOTAL	100.00

El orden en que se determina cada uno de los elementos de las tablas 8.3 y compuestos de la tabla 8.4, es de acuerdo a la numeración entregada en las tablas mencionadas. A continuación se detalla el cálculo de cada elemento y compuesto de acuerdo al orden establecido:

1. La ley de cobre en la Escoria es un parámetro de operación con el cual se desea operar el horno.
2. Como se conoce el porcentaje de cobre en la escoria, se determina el porcentaje de Cu_2S .

$$\% \text{Cu}_2\text{S}_{\text{ESC}} = \frac{\% \text{Cu}_{\text{ESC}} \times \text{Pm}(\text{Cu}_2\text{S})}{2 \times \text{Pa}(\text{Cu})} \quad (\text{Ec. 8.6})$$

3. Conocido el porcentaje de Cu_2S se determina el porcentaje de Azufre.

$$\% \text{S}_{\text{ESC}} = \frac{\% \text{S}_{\text{ESC}} \times \text{Pa}(\text{S})}{\text{Pm}(\text{Cu}_2\text{S})} \quad (\text{Ec. 8.7})$$

4. El porcentaje de sílice es un parámetro de operación con el cual se desea operar el horno.
5. El porcentaje de magnetita es un parámetro de operación con el cual se desea operar el horno.
6. El porcentaje de otros para el sistema de ecuaciones 3.9 queda determinado una vez que se resuelve dicho sistema ya que el porcentaje de otros también es una incógnita a determinar. El porcentaje de otros en la escoria se determina con el cociente de la cantidad de otros en la escoria y la cantidad de escoria:

$$\% \text{Ot}_{\text{ESC}} = \frac{\text{Ot}_{\text{ESC}}}{\text{Escoria}} \quad (\text{Ec. 8.8})$$

7. El porcentaje de FeO se determina mediante la diferencia de 100 menos la suma de los porcentajes de los componentes restantes.

8. El porcentaje Fierro en la escoria se determina de la siguiente manera:

$$\%Fe_{ESC} = \%FeO_{ESC} \times \frac{Pa(Fe)}{Pm(FeO)} + \%Fe_3O_4_{ESC} \times \frac{3 \times Pa(Fe)}{Pm(Fe_3O_4)} \quad (Ec. 8.9)$$

9. El porcentaje de Oxígeno en la escoria se determina de la siguiente manera:

$$\begin{aligned} \%Oxig_{\cdot ESC} = \%SiO_2_{ESC} \times \frac{2 \times Pa(O)}{Pm(SiO_2)} + \%FeO_{ESC} \times \frac{Pa(O)}{Pm(FeO)} \quad (Ec. 8.10) \\ + \%Fe_3O_4_{ESC} \times \frac{4 \times Pa(O)}{Pm(Fe_3O_4)} \end{aligned}$$

8.2. APÉNDICE B: Balance de azufre y oxígeno

Para la determinación del coeficiente de oxígeno (ecuación 3.11), se procede primeramente con el balance de azufre y oxígeno, el cual está resumido en la tabla 8.5 y permite determinar la demanda de oxígeno.

Tabla 8.5. Balance de azufre y oxígeno.

ENTRA	Ton S/h	Ton O/h
Concentrado	47.26	0.00
Circulante	1.18	0.00
TOTAL	48.44	0.00

SALE	Ton S/h	Ton O/h
Eje	16.30	0.33
Escoria	1.37	11.72
Polvos	0.81	0.00
TOTAL	18.48	12.05

Dif. (Entra – Sale)	29.96	-12.05
---------------------	-------	--------

La demanda de oxígeno en Nm³/h se determina mediante la siguiente expresión:

$$\text{Demanda Ox.} \left[\frac{\text{Nm}^3}{\text{h}} \right] = \frac{22414}{32} \times [(\text{Ton S/h})_{\text{ent-sale}} - (\text{Ton O/h})_{\text{ent-sale}}] \quad (\text{Ec. 8.11})$$