ASADES

Energías Renovables y Medio Ambiente
Vol. 30, pp. 21 - 28, 2012
Impreso en la Argentina. ISSN 0328-932X

ESTÁNDAR DE DOCUMENTOS EMPAQUETADOS DE SIMUSOL, REPOSITORIO CSSAN Y OTROS AVANCES.

Diego Saravia y Dolores Alía

Facultad de Ciencias Exactas – Universidad Nacional de Salta (U.N.Sa.)
INENCO (U.N.Sa – CONICET)
Fax: 54-387-4255489, Av. Bolivia 5150. 4400 Salta, Argentina
email: Diego.Saravia@gmail.com, dsa@unsa.edu.ar,
loli@unsa.edu.ar, Dolores.Alia@gmail.com

Recibido: 09-08-12; Aceptado: 24-09-12.

RESUMEN.- Se presenta un estándar de paquetes para Simusol que permite contener en un solo archivo denominado "documento": diagramas, tablas, gráficos, módulos, resultados y otros elementos facilitando la edición, simulación, instalación, transporte y almacenamiento de los mismos, habilitando el trabajo colaborativo. Complementariamente se desarrolló un software de referencia para instrumentarlo; un repositorio de paquetes con su software; repositorios de control de cambio (con software Subversión); una interfaz gráfica que puede utilizarlos; y cambios al Simusol para que trabaje con las nuevas ideas.

Palabras claves: Simusol, estándar, paquetes, documento, GUI, repositorio.

SIMUSOL DOCUMENTS STANDARD PACKAGING , CSSAN REPOSITORY, AND OTHER ADVANCES.

ABSTRACT.- We present a package standard for Simusol which allows to contain in a single file, called a ``document'': diagrams, tables, charts, modules, results, and other elements, making it easier to edit, simulate,install, transport and store these same elements and so enabling collaborative work. Additionally, we developed a reference software to implement it, a repository of packages together with its software; exchange control repositories (with Subversion software); a graphical interface for them, and some changes to Simusol in order for it to be adapted to these new standard.

Keywords: Simusol, package standard, document, GUI, repository

1. INTRODUCCIÓN

En el INENCO hemos desarrollado y usado Simusol desde el 2002 (Alía de Saravia, D., Saravia, L., y Saravia, D., 2002), generando mejoras en forma continua hasta la fecha.

En este momento nos enfocamos en facilitar su uso, tanto proveyendo una interfaz gráfica imprescindible en Windows y útil en GNU/Linux, automatizando la instalación de extensiones y la creación y distribución de estos para que terceros los puedan usar, permitiendo el surgimiento de una comunidad de desarrollo de soluciones de Simusol.

Dada la cantidad de trabajos emprendidos por diversas personas con Simusol, y para que estos trabajos pueden ser utilizados por otros, en un marco colaborativo, resulta conveniente crear conceptos y herramientas que lo faciliten.

2. PROBLEMÁTICA

Simusol utiliza un conjunto de archivos:

- **Diagramas (.dia).** Son archivos xml. Todos los diagramas de Simusol tienen la extensión .dia. Se visualizan en la ventana (tela, canvas) del programa Dia, como un conjunto de objetos interconectados.

- **Objetos** que pueden ser de los siguientes tipos:
  - cuadros,
  - ramas,
  - nodos (simples y múltiples, en particular el nodo de referencia),
  - elementos,
  - modelos,
  - otros (que el Simusol ignora, como textos u otros símbolos sin uso establecido).

Los objetos del tipo elemento representan a un Capacitor, Fuente de tensión, Fuente de corriente, Inductor o Resistor y su descripción en una hoja debe terminar respectivamente en C,E,J,L,R; mayúscula o minúscula según que la forma tenga sus dos puntos de conexión o sólo uno. Los objetos tipo modelo representan objetos más complejos que un elemento y su descripción en una hoja debe terminar en T.

Algunos objetos del Dia son definidos por el mismo; otros, los que más interesan a los desarrolladores de Simusol, están definidos en las formas (shapes) que se encuentran listadas en archivos del tipo hojas (sheets). Los programas Dia y Simusol deben poder acceder a estos archivos para funcionar. Además de las formas asociadas a cada objeto podemos asociar diagramas a alguno de ellos.

Existen diagramas de varios tipos:

- **Principales (proyectos) y auxiliares.** Para cada simulación, procesamiento o corrida del Simusol (compilación + enlace + ejecución), existe un diagrama principal, generalmente referido como "proyecto". Pueden existir diagramas auxiliares convocados desde el principal que contienen indicaciones útiles a Simusol; cuál es el caso está determinado por el contenido del diagrama.
- **Elementos.** Estos diagramas contienen definiciones de objetos del tipo elemento o del tipo nodo. Puede haber objetos elemento o nodo sin diagrama pero es raro, y cada diagrama puede contener muchas definiciones de objetos. Cada objeto tipo elemento está asociado a un archivo de forma (shape) y su descripción está contenida en una o más hojas (sheets).
- **Modelos.** Estos diagramas dan la definición de un solo objeto del tipo modelo. Cada objeto modelo tiene un diagrama, además un archivo de forma (shape) y su descripción está contenida en una o mas hojas (sheets).

- **Formas y Hojas.** Son archivos xml de extension **.shape y .sheet** respectivamente, imprescindibles para el buen funcionamiento del Dia, que se identifican por un nombre interno que no tiene necesariamente correlación con el nombre del archivo. Dia no acepta nombres internos repetidos para las formas y hojas. Avisa del problema para las formas.

Cada forma se relaciona con un archivo conteniendo un icono para su representación gráfica en el menú del Dia.

Las hojas son un listado de menciones de formas u objetos que aparecen como un sub-menu que se elige en el Dia. Las hojas actúan como sub menus para ofrecer formas. Dia reconoce todas las formas que haya encontrado, pero sólo ofrece, mediante menús, incorporar las formas a un diagrama, las que estén en una hoja encontrada por Dia. Durante el uso de Dia, en su menú se ofrecen en todo momento, dos conjuntos de objetos: el primero es siempre el mismo; el segundo es el asociado a una hoja. El usuario puede cambiar cuál es la hoja que se ofrecerá sin que eso afecte al diagrama que se esté elaborando. No hay limitaciones para cuales objetos o formas aparecen en una hoja. Es sólo una cuestión de practicidad relacionada con el diagrama que se esté produciendo, a efectos de disminuir la necesidad de cambios de hojas y su frecuencia.

Las hojas y formas que usa Dia son las disponibles al momento de cargarlo. Si un usuario instala hojas o formas nuevas (elementos o modelos) debe recargar Dia. Durante el uso de Dia se visualizan en el menú, iconos que representan los objetos o formas, ofrecidos por la hoja elegida.

Día tiene previsto un directorio donde el usuario puede guardar sus definiciones de hojas; y un directorio a partir del cual puede guardar sus archivos de formas e iconos. También directorios del sistema a los que todos los usuarios pueden acceder.

Las formas incluidas en una hoja además de visualizarse en el menú con un ícono pueden tener descripciones que hace la hoja para ellas en uno o más idiomas. La descripción, en el idioma en uso de Dia, se visualiza al tocar con el ratón el icono del menú.

Hojas para Simusol. Algunas de las hojas y formas para Dia son relevantes para Simusol. Su nombre interno es de la forma Simusol–Algo. Ejemplos actuales de Algo: Termico, Electrico, Generico, Modelos; Hidraulico (en revisión). Una hoja de nombre Simusol–Algo puede mencionar cualquier objeto o forma; pero las formas que llamaremos propias de la hoja son las de nombre Algo – Cosa.

El objeto caja, o "UML - Object" de Dia, es muy usado y por ello conviene que esté presente en todas las hojas de Simusol.

Simusol también usa ramas para conectar shapes, las ramas están siempre disponibles para Dia.
Las shapes propias de Simusol de nombre Algo–Cosa se usan para representar formas simples, de clase: Nodo, Capacitor, Fuente de tension, Fuente de corriente, Inductor o Resistor o formas más complejas que llamamos Modelos.

Formas para Simusol. Simusol exige que las formas propias de Simusol cumplan ciertas condiciones:

- **Texto identificatorio**: Todas ellas deben admitir uno, salvo los nodos de referencia.
- **Puntos de conexión**: Los representantes de nodos de refrencia tienen un punto de conexión. Los representantes de los demás nodos, deben tener al menos uno, y pueden tener uno "principal"; las otras formas simples, uno o dos puntos de conexión, pero ninguno de tipo "principal"; los Modelos cualquier cantidad, pero ninguno "principal". Las formas simples con un solo punto de conexión tienen siempre otra conexión ya hecha a un nodo de referencia.
- **Descripciones en las hojas que los ofrecen**: Deben tenerlas; y esas descripciones, salvo que sirvan para representar nodos, tienen al final, luego de un punto y coma, una terminación. La misma terminación para todos los lenguajes.
- **Terminación**: Si sirve para representar a un Capacitor, Fuente de tension, Fuente de corriente, Inductor o Resistor deben terminar respectivamente en: C,E,J,L,R; mayúscula o minúscula según que la forma tenga sus dos puntos de conexión o sólo uno.

- **Funciones.** Escritas en fortran, c u otros lenguajes que puedan ser enlazadas (linkeadas) con Sceptre por el enlazador del gcc.
- **Datos.** De diverso tipo, generalmente en formato de tablas, que Simusol traduce para usar con Sceptre, o datos para comparar resultados, etc.

Para organizar y facilitar el uso, el compartir e instalar estos archivos en los lugares adecuados, se ha creado e instrumentado el concepto de paquete que se describe en este trabajo.

3. METODOLOGÍAS Y HERRAMIENTAS

Se ha utilizado: Perl (O'Reilly, 2012) como lenguaje principal de programación y Wxwidgets (Ollivier, 2012) para desarrollar la interfaz gráfica; Make (FSF, 2010) para desarrollar el instalador y el manejador de paquetes; Subversion (Apache Software Foundation, 2010), Perl y Apache para desarrollar el repositorio de paquetes. CMSimple (CMSimple.org 2012), para crear el sitio web, un sistema que crea un solo archivo "html" en el sitio y que no requiere bases de datos. Este archivo es respaldado con un sistema de control de versiones.

Todos los desarrollos se efectúan en los repositorios con control de versiones aquí descriptos. Uno por cada proyecto. Tuvimos un problema con el repositorio que utilizábamos antes y por lo tanto se partió de la última versión preexistente para crear la primera versión en estos repositorios.

4. RESULTADOS

Para cumplir los objetivos centrales previstos se realizaron varias acciones concretas:

- definición de un estándar de paquete para compartir trabajos con Simusol.
- creación de un software (Simpack) que permita crear, instalar, subir y bajar paquetes del repositorio
- cambios al Simusol para trabajar con las nuevas estructuras.
- creación de una interfaz gráfica que trabaja centrada en paquetes y que funcione en diferentes sistemas operativos.
- creación del sitio web del repositorio que permita administrar los paquetes: "cssan.simusol.org.ar" y el software que lo hace funcionar (SimpackSrv) Este sitio web ofrece:
- el registro de usuarios y un espacio para subir sus paquetes, validarlos y publicarlos.
- repositorios subversion para que los participantes puedan trabajar en sus paquetes,
- una lista de correo para desarrolladores,
- verificación de que ningún paquete use palabras reservadas para sus partes, recordando qué palabras reservó cada paquete previamente subido.
- La posibilidad de que un participante avale el trabajo de otro.

Los documentos (archivos s.tgz o paquetes) de Simusol son paquetes del tipo fuentes. En el futuro y mediante summapack (Saravia, 2010b) crearemos herramientas para crear paquetes tipo deb, rpm, ebuilds, y otros a partir de los paquetes fuentes.

4.1. El concepto de documento Simusol y su estándar

Un documento o paquete de Simusol es un contenedor para los diferentes archivos que se usan y generan al trabajar con el mismo. Permite su fácil uso, reuso, almacenamiento, transporte e instalación de modelos, elementos o proyectos que puedan ser usados o visualizados por terceros.

El estándar consiste en un conjunto de reglas que deben cumplir los paquetes de Simusol. Un paquete es un árbol de directorios empaquetado con tar y comprimido con gzip. en un solo archivo con extensión ".s.tgz". y podemos pensarlo como un "documento" de Simusol, por ello se usa indistintamente varias denominaciones: paquete, documento, archivo s.tgz.

La interfaz de usuario trabaja conceptualmente con estos documentos, abstrayendo al usuario de su estructura interna: abriéndolos, modificándolos y salvándolos. Cada documento puede contener uno o más diagramas principales, o ninguno en el caso de contener sólo documentos, modelos, elementos u otra información. La interfaz podrá procesar todos estos diagramas a la vez o uno por vez. Similarmente actúa con los documentos LaTeX contenidos. Un documento ya procesado permitirá mostrar los resultados sin reprocesar, (aún cuando no esté instalado Sceptre o Simusol), mediante los archivos gráficos de resultados ya almacenados.

Los documentos tienen nombre para poder identificarlos y versión para señalar la evolución del mismo con el tiempo. Su nombre es de la forma: nombre_del_paquete-version_del_paquete-[mod|ele|pro|other].s.tgz

La versión de la forma 24.12, consta de dos números separados por un punto, el segundo número será menor a 1.000.000. También en el repositorio existen los paquetes nombre-last.s.tgz, con la version "last", significando la última.

Un paquete puede depender de otros para ser procesado por Simusol. Por ejemplo un paquete que contiene un modelo que otro paquete usa. El paquete que lo usa requiere tener instalado el primer paquete para ser procesado. Por ello es útil el concepto de instalar en un sistema un documento Simusol, sea para permitir a otros documentos usar los módulos de los que depende o para visualizar los resultados (aún cuando el Simusol mismo no esté completamente instalado).

Cada paquete tiene una estructura de directorios para albergar todo tipo de contenido vinculado a Simusol y Sceptre:

projects: contiene diagramas principales. Por cada diagrama procesable habrá un subdirectorio con su nombre conteniendo diversos archivos producidos por su procesamiento, en subdirectorios denominados: tests, results, tmps. Podrá haber diagramas en el directorio raíz del paquete, pero no se aconseja;

data: todo tipo de archivos tipo tablas leibles por Simusol y que de corresponder éste incorpora a Sceptre o que se usan a otros efectos;

shapes: iconos y sus formas;
soft: todo tipo de programa, script, etc. no incluído en los anteriores, cada programa en un subdirectorio con su nombre, en un subdirectorio src con sus fuentes;
sheets: hojas que hacen referencia a sus formas y quizas a otras contenidas en los paquetes del cual este depende;
elements: diagramas de los elementos que se incluyan;
models: diagramas de los modelos que se incluyan;
docs: documentacion, en sus fuentes y al menos 3 versiones .txt .pdf y .html, cada documento en un subdirectorio (con el nombre), en el subdirectorio src las fuentes. También se colocarán en un subdirectorio man los .man, en otro info los .info y en un pod los .pod;
functions: funciones tanto en c, fortran u otro lenguaje que produzca objetos enlazables con los objetos del sceptre (gcc, gfortran), cada una en un subdirectorio (con su nombre);
objects: objetos del Dia;
graphics: graficos de interés, por ejemplos archivos .png de los diagramas, u otros usados en LaTeX o documentos HTML que se quieran poner allí; **auxs:** para diagramas o archivos .tex auxiliares, es decir llamados desde un principal;
Pm: para módulos perl;
other: otro material. No todos los paquetes tienen todos los directorios con contenido. Contiene en su raiz, los siguientes archivos:

- **INSTALL**: indicando cualquier instrucción particular del paquete para su instalación;
- **README** indicando entre otros funciones del paquete, sus conceptos, que contiene y como se usa;
- **COPYING*** licencias, el repositorio sólo acepta GPL, AGPL y FDL, cada una según el tipo de contenido. Los autores mantienen los derechos, no se pide transferencia de derechos de autor;
- **AUTHORS**: listado de autores;
- **Makefile** para su instalación y proceso, la genérica contenida en "skeleton" debe funcionar (no se recomienda usar otra);
- **NEWS**: noticias;
- **THANKS**: personas que colaboraron;
- **TODO**: tareas a realizar;
- **nombre_del_paqute. summa**: descripción del paquete. El archivo nombre_del_paquete.summa es un archivo que contiene datos del paquete en forma procesable automáticamente por bash con las variables:
  - **DEPENDS**: lista de dependencias;
  - **IDEPENDS:** lista de dependidos. (No escribirla. Al subir el summa el repo lo pone automáticamente);
  - **BASEVERSION:** mismo par de números que en el nombre del paquete;
  - **SIMUSOLTYPE:** other, mod, ele, proyect;
  - **DESCRIPTION:** descripción del paquete;
  - **SUMMARY:** descripción resumida;
  - **KEYWORDS:** palabras claves separadas por espacios , la primera debe ser el nombre del paquete;
  - **AUTHOR:** "nombre <mail>, nombre <mail>";
  - **MAINTAINER**: idem a author;
  - **VENDOR**: institucion como el author;
  - **PRIMARYSITE**: puede poner https://cssan.simusol.org.ar;
  - **SRC_URI**: = $PRIMARYSITE/pkgs/nombre;
  - **SRC_PKG_TYPE**: =simusol;
  - **CATEGORY**:=[secado,calentador agua, etc];
  - **SUMMAVERSION**: la versión del archivo summa dentro de la misma versión de paquete, ya que en cada cambio de versión del paquete cambia la variable BASEVERSION.

Para crear el paquete se recomienda hacerlo a partir de la estructura vacía del paquete "skeleton". Con el comando simpack -k, se crea un directorio skeleton con esta estructura. Conviene renombrar este directorio con el nombre del paquete que se desea crear y trabajar agregando contenido.

Los tipos ele, mod y pro se utilizarán para paquetes que contengan solamente material relacionado respectivamente a un elemento, modelo o proyecto (al menos un diagrama principal, sin modelos o elementos utilizables por otro paquete). Cualquier paquete que no sea exclusivamente de estos tipos o que tenga mezclas de tipos sera del tipo "other". Se aconseja fraccionar los paquetes uno para cada tipo. Si una función es usada por varios, lo mejor es ponerla en un paquete y designarlo como dependencia de los que la usan.

Simusol cuando es llamado por el Makefile y por lo tanto por la interfaz gráfica, usa como directorio de salida uno con el nombre del diagrama, colocando los archivos de salida en los directorios: results, tests y tmps.

Uno o más modelos pueden estar contenidos en un paquete, al igual que un diagrama principal. Para usar el diagrama principal hay que instalar (o crear) el paquete que lo contiene y además el o los paquetes con los modelos del cual depende. Cada vez que alguien instala un modelo para usar con Simusol, todos estos archivos se deben ubicar en un lugar apto para todos los programas que los requieren: Dia, Simusol, y Sceptre.

Si el instalador (mediante Simpack) tiene acceso a la cuenta de "root" o "administrador" y así lo desea, los archivos pueden instalarse en el sistema para que todos los usuarios accedan a los mismos; o en caso contrario sólo para un usuario en particular. En el caso de ser instalados para todo el sistema se lo puede hacer como paquete del sistema (.deb o rpm, o tgz), en el directorio /usr, o bien en el /usr/local como un programa "local". También es posible instalar estos archivos en el mismo directorio donde se trabaja con el diagrama que los convocará, o si el diagrama está contenido en un paquete "desempaquetado", en los directorios esperados. Los programas que utilizamos buscarán un archivo determinado eligiendo el primero que encuentren según el siguiente orden:

- Directorio corriente
- Estructura tipo paquete del directorio corriente
- Directorio donde está el diagrama principal,
- Directorios específicos del paquete "desempaquetado" que contiene al diagrama principal,
- Directorios donde el usuario instala paquetes (~/.dia, ~/.simusol, ~/share, ~/bin, etc.)
- Directorios del sistema (/usr/local/share/dia, /usr/local/share/simusol, /usr/local/lib/simusol)
- Directorios del sistema para paquetería del mismo (/usr/share/dia /usr/share/simusol, /usr/lib/simusol). Los "share" se usan para archivos independientes de la arquitectura de la computadora (scripts) y los "lib" para las librerías que sí lo sean (ejecutables).

Si en algún proyecto u otro archivo se especifica un camino desde la raíz, debe buscarlo allí, caso contrario a partir de las ubicaciones ya listadas. Este sistema de búsqueda puede estructurarse como una variable interna o de entorno, tipo PATH: SIMUSOLPATH.

La interfaz gráfica tiene un espacio de trabajo interno donde mantendrá abiertos y desempaquetados los paquetes y donde funcionan los comandos de bajo nivel del Makefile (llamados por la interfaz) y los de menor nivel llamados por el Makefile o por el usuario directamente en consola. Un usuario no puede reprocesar un proyecto "instalado" en el sistema, pero todos los usuarios pueden ver los resultados, de existir; además puede copiar los proyectos instalados, y otros elementos para su uso y/o reprocesamiento en sus propios proyectos.

4.2. Simpack

Es el software que maneja los paquetes de Simusol. Este software empaqueta y desempaqueta los archivos, los sube y baja del repositorio, los instala en el sistema y otras funciones requeridas. Desde la orden de comandos se convoca como:

simpack ORDEN PAQUETE

donde la ORDEN puede ser:

- **c:** prueba y crea el paquete en SHARE/simusol/pkg, en adelante PKG;
- **p:** copia el paquete al directorio de ejecución;
- **is:** abre, prueba e instala el paquete indicado (segun sea root o usuario comun) del directorio de ejecución o de PKG, sin dependencias;
- **a:** abre el paquete al directorio de ejecución y lo prueba;
- **ni:** abre y prueba el paquete (no instala);
- **nc:** prueba el directorio (no crea); **i:** baja e instala el paquete y sus dependencias (no prueba);
- **dni:** baja el paquete y sus dependencias pero no instala (no prueba);
- **ib:** instala el paquete y sus dependencias sin bajarlos, previamente bajados;
- **u:** sube paquetes al servidor;
- **k:** crea esqueleto de paquete en directorio de ejecución;
- **r:** limpia cache con arbol de paquetes abiertos (PKG);
- **DISTRO:** rpm, deb, ebuild, slack: generan paquetes en esos formatos;
- **SVN:** import, co: crea un repo en el sitio del simusol e instala version local con svn todos los otros como up y ci;
- **h:** ayuda.

Probar el paquete implica verificar que cumple con el estándar. El software también verifica que otros paquetes no usen los mismos nombres, ni contengan archivos con el mismo nombre que otros paquetes situados en el repositorio. Estos nombres: de paquetes, formas, hojas, modelos, elementos, etc. se reservan para el primer paquete que los contenga y no se podrá subir un nombre ya usado.

4.3. El repositorio de paquetes CSSAN

El repositorio permite compartir documentos Simusol y en particular los que contienen modelos o elementos. Los documentos pueden ser subidos y bajados fácilmente del repositorio.

Se utiliza el mismo esquema de uso que otras comunidades en el mundo del software libre, creando un espacio para compartir contribuciones, un almacén o repositorio de material útil para trabajar con Simusol. De allí el nombre CSSAN: "Comprehensive SimuSol Archive Network".

Es importante asegurar la calidad de los paquetes y generar antecedentes para los participantes, por ello se estructura un sistema de revisión por pares para evaluar y certificar los paquetes.

En el repositorio CSSAN cada autor suscrito puede subir sus paquetes. Todos podrán bajar los paquetes para su uso. Es libre la inscripción como autor. Hay un paquete ejemplo denominado "skeleton", vacío, para facilitar que los participantes incorporen sus contribuciones.

4.4. SimpackSRV

Es el software del sitio CSSAN. Se convoca desde la interfaz web de la siguiente forma:

spserv ORDEN USER PASSWD MAIL PROYECT CODE NUEVAPASSWD

Todo dato que no sea relevante se reemplaza con "-". Los comandos tipo define, almacenan internamente una acción, que al ser citada por su código producen el cambio predefinido. Es llamado principalmente desde apache mediante "cgi".

ORDEN:

- **-dnc:** define nuevo usuario, spserv -dnc USER - MAIL PROYECT CODE NUEVAPASSWD;
- **-dnm:** define nuevo mail, spserv -dnm USER PASSWD MAIL;
- **-drc:** define recuperacion de contraseña, spserv -drc USER - - - - NUEVAPASSWD;
- **-cc:** cambia contraseña, spserv -cc USER PASSWD - - - NUEVAPASSWD;
- **-np:** agrega un nuevo proyecto, spserv -np USER PASSWD - PROYECT ;
- **-code:** ejecuta la accion definida con ese código, spserv -code - - - - CODE;
- **-rp:** reviso paquete subido por el usuario y publico, spserv -rp USER PASSWD - - - - - PAQUETE;
- **-av:** avalo paquete publicado, spserv -av USER PASSWD - - - - - PAQUETE;
- **-pp:** prueba palabras, spserv -pp - - - - - - PALABRAS;
- **-au:** autentico, spserv -au USER PASSWD;
- **-basedir:** informo basedir, spserv -basedir;
- **-avales:** lista avales, spserv -avales ;
- **-paqpub:** lista paquetes publicados, spserv -paqpub;
- **-paqusr:** lista paquetes subidos por autor, spserv -paqusr USER;
- **-inicio:** ejecuta como root desde el makefile las acciones necesarias para instalar el sistema, spserv -inicio;

- **-reuser:** ejecuta como root a mano, recrea los usuarios, spserv -reuser.

4.5. Repositorios Subversion para los participantes

CSSAN provee de infraestructura para el desarrollo de software, consistente en un número ilimitado de repositorios subversion, para los participantes.

Estos repositorios, por defecto, permiten a cualquiera mirar el proyecto y al creador modificarlo. Existe un comando para incorporar más usuarios a cada repositorio y para que sólo puedan verlo los incorporados. Configuraciones más complejas (unos ven y otros escriben) son también posibles.

Se alienta a compartir la vista de los proyectos con todos y todas, salvo que se usen para guardar información personal, lo cual no se recomienda, dado que no es un sitio seguro, ni pensado especialmente para serlo.

Las acciones que se pueden efectuar en el sistema de repositorios son: 1. Crear usuario, 2. Crear Proyecto, 3. Importar Proyecto, 4 Crear copia de Trabajo, 5. Realizar cambios en proyecto, 6 Actualizar copia de trabajo, 7. Actualizar repositorio, 8. Acceder al repositorio.

Las acciones que se utilizan habitualmente son la actualización de la copia de trabajo desde el repositorio, para incorporar a la máquina de trabajo los cambios efectuados por otras personas o en otras máquinas, y la actualización del repositorio para "subir" las mejoras efectuadas por algún usuario.

4.6. Listas de correo electrónico, portal y micrositios de Simusol y CSSAN, otros recursos de internet

Se han creado dos listas de correo electrónico, una para usuarios (Google, 2011a) y otra para desarrolladores (Google, 2011b) También un grupo de Facebook (Facebook, 2011).

Para el dictado de cursos y para ayudar a la autoformación se subieron a un canal de Youtube, una serie de videos tomados del último curso de posgrado dictado. Todavía no están editados (YouTube, 2011).

Se ha creado un curso virtual en el sitio Moodle de la Facultad de Ciencias Exactas de la UNSa (Moodle, 2011).

Se ha migrado el sitio a un sistema denominado CMSimple, que permite guardarlo en un solo archivo html sin usar bases de datos.

4.7.Interfaz de usuario

La forma de trabajo habitual con Simusol implica el uso en la línea de comandos de programas en perl que interactúan con los archivos de datos y con el ejecutable Sceptre.

Se ha creado un Makefile, que además de instalar todos los programas necesarios para trabajar con Simusol, tanto en GNU/Linux como en CygWin (Windows), permite el control de los ejecutables anteriores y los paquetes en un nivel de abstracción mayor que los programas previamente existentes en Simusol. El Makefile abstrae cierta complejidad al usuario evitando la necesidad de usar directamente la línea de comandos.

Este Makefile es usado además por la interfaz gráfica a un nivel de abstracción aún mayor. Para usar el Makefile se debe tener instalado gnu-make. Una vez autoinstalado el Makefile puede ejecutarse con la línea de comandos bajo el nombre simusol-make. Para iniciar un proyecto se debe crear un directorio con el nombre que quiera darle al proyecto, y ejecute simusol-make desde ese directorio. Este comando puede activase con distintas órdenes y variables. Por ejemplo la variable proyecto que indica que diagrama se procesará. Así:

```
simusol-make proyecto=acumulador  muestra
```

mostrará los resultados existentes para el diagrama principal "acumulador.dia" Algunas otras ordenes son:

- **clean,** para borrar resultados;
- **[simusol|sceptre|ejecuta]** para procesar el diagrama con simusol; con simusol y sceptre; y finalmente con simusol, sceptre y postprocesamiento gráfico. Si una etapa ya fue cumplida no la realiza;
- **installpack** para instalar el paquete;
- **repo** para poner el paquete en el repositorio local;
- **upload** para subirlo al sitio,
- **help** para conseguir ayuda, etc..

4.8. Instalar Simusol con Makefile

La forma más simple de instalar Simusol desde las fuentes se efectúa usando el Makefile maestro de los proyectos y paquetes de Simusol. Descárguelo, ubíquelo dentro de cualquier directorio (donde no exista otro Makefile), y ejecute:

a) "make install", instalación para un usuario; o b) "sudo make install", instalación como supervisor para todos los usuarios. Es conveniente que el usuario tenga configurado el sudo para ser usado sin clave. El Makefile se instala en el directorio /usr/bin o eventualmente en el ~/bin (ver /usr/local) como simusol-make. Por lo que una vez bajado en un directorio de descarga, al ejecutar make install, dependiendo si se es o no root, se instalara en dicho sitio. A partir de este momento se podrá ejecutar y actuará en función de en cuál directorio estemos parados.

La instalación en un Ubuntu, tal cual se instala por defecto, suele tardar una hora con una conexión de internet vía ADSL de mediana calidad, (bajando a unos 30K). Descarga unos 60 MB en unos 90 paquetes que ocupan 160 MB del disco. A estos paquetes hay que sumarles los módulos Perl tomados del CPAN y el software propio de Simusol. Si la instalación falla, hay que reintentar.

Los proyectos, salvo que el autor lo modifique, suelen tener el mismo Makefile (o versiones anteriores) en su directorio. En tal sentido es un archivo autoinstalable, que instala y coordina el uso de Simusol. Si Ud. recibe un paquete Simusol, y lo desempaqueta con tar/gzip, tiene acceso a

Internet y este servidor está funcionando, ejecutando make install, también podrá instalar Simusol.

En algunos sistemas, si usted. es el primer usuario que instala Simusol, requerirá si o si, instalarlo como Supervisor, ya que el Makefile instala software que no sabe operar sin instalarse en áreas de su computadora que el usuario no puede cambiar. Si la instalación fue exitosa, aparecerá un icono apropiado para ejecutar la interfaz gráfica.

Ha sido probado en Ubuntu 11.10, Ututo 2012, Debian 6 y Suse 11.4, con Suse 11.3 y anteriores hay que tocar un poco el Makefile, en su interior está comentado el código necesario.

Cuando se ejecuta "sudo make install" por primera vez sobre el Makefile para instalar, en algunas oportunidades falla la instalación del Wx. Se suele solucionar ejecutando de nuevo la instalación.

4.9. Cambios al Simusol

Se adaptó Simusol para que encuentre los archivos en las ubicaciones previstas en los paquetes y para que los resultados los organice de acuerdo al estándar en tres directorios: results, tests, tmps; se habilitaron opciones en la línea de comandos.

4.10. El Instalador de Windows

Como Windows no tiene un sistema de paquetes centralizado como la mayoría de las distribuciones GNU/Linux, es necesario crear un software que ayude al usuario a instalar los diversos programas de los que depende simusol para su funcionamiento: Dia, emacs, cygwin, gnuplot, etc.

A estos efectos se creó el programa LiberaWin.exe que cumple esta tarea. Hay que desactivar todo tipo de firewall o analizador de virus antes de ejecutar este programa en Windows. En algunos casos interfieren con el proceso de instalación. Liberawin se descarga del sitio de Simusol.

El último paso de la instalación de LiberaWin, una vez instalado Cygwin, es ejecutar en el mismo el Makefile descripto anteriormente. Con lo cual Simusol en Windows, usando Cygwin, funciona exactamente igual que en GNU/Linux y tal como aquí se ha descripto.

4.11. La Interfaz Gráfica

Se abandonó la interfaz gráfica basada en XUL presentada en trabajos anteriores, debido a que cambios en las nuevas versiones del Firefox anularon algunas características que utilizábamos del XUL, y este ya no funcionaba.

Hemos adoptado el proyecto WxWidgets a los efectos de contar con una interfaz que pueda ser utilizada en diversos sistemas operativos.

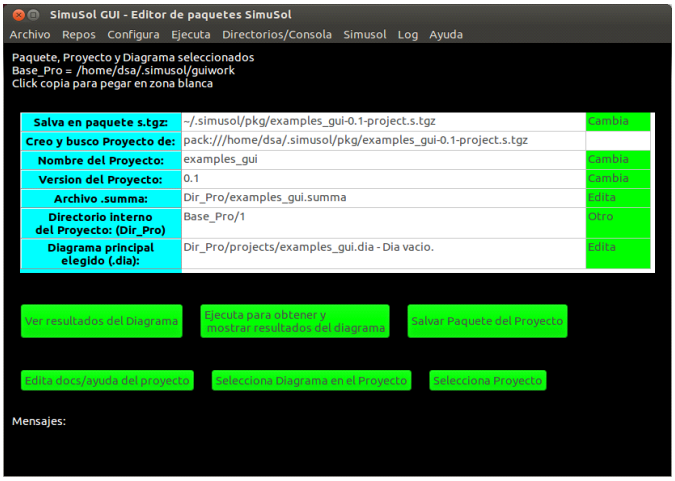

Fig. 1: Pantalla principal de simusol-gui

La interfaz integra el concepto de paquetes y orienta el usuario a estos. Así la forma habitual de trabajo es abrir un paquete (archivo s.tgz) y modificar el mismo, salvándolo cuando se lo desea.

Normalmente la interfaz tiene varios documentos (paquetes) abiertos y provee una ventana para cambiar el paquete con el que se trabaja.

Cada archivo podrá tener varios diagramas directamente procesables y otros que proveen de información adicional a estos, como auxiliares, modelos y elementos.

Desde la interfaz se puede abrir una ventana del editor de diagramas para todos estos, como también calcular la simulación prevista en los procesables.

En las primeras versiones de simusol-gui se denomina al paquete o documento como proyecto (puede tener muchos diagramas principales), en las siguientes, al igual que en este trabajo se denomina proyecto a un diagrama principal.

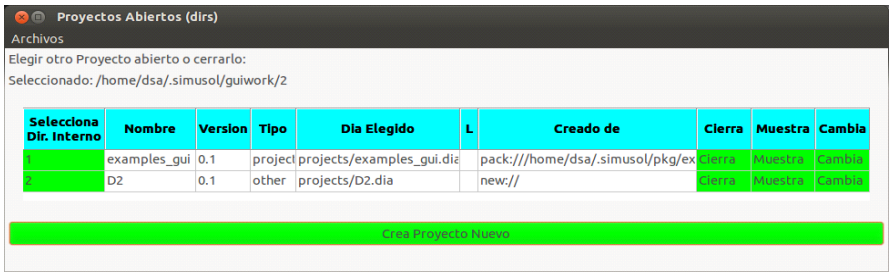

Fig. 2: Pantalla de selección de paquetes.

5. CONCLUSIONES

Se ha dotado al proyecto Simusol de poderosas herramientas y estándares para permitir el trabajo colaborativo. Herramientas habituales en todo proyecto de software libre: listas de correo, repositorios con control de versiones, un estándar para sus paquetes, junto con herramientas para su uso. La interfaz de usuario autoinstalable en su línea de comandos y la interfaz gráfica que maneja los nuevos paquetes, junto con el instalador de windows, hacen más fácil su uso y el intercambiar trabajos.

Se invita a todos aquellos que desarrollaron trabajos con Simusol a publicar los mismos en el repositorio CSSAN y a todos los usuarios a anotarse en las listas de correo, compartir por esa vía todas las inquietudes y avances.

REFERENCIAS

Alía de Saravia, D., Saravia, L., y Saravia, D., (2002). Avances introducidos en la capacidad del simulador de sistemas solares térmicos SIMUTERM (SIMUSOL). AVERMA. 2002, Vol 6 N2 8.31— 8.36. http://www.cricyt.edu.ar/lahv/asades/modulos/averma/trabajos/2002/2002-t008-a006.pdf

Alía de Saravia, D., Saravia, L. y Saravia, D., (2009). Sitio web del Simusol. http://www.simusol.org.ar

Apache Software Foundation, The (2010). Apache Subversion http://subversion.apache.org/

Apache Software Foundation, The (2012b). Apache HTTP Server, http://projects.apache.org/projects/http_server.html

CMSimple.org (2012), http://www.cmsimple.org/

Moodle 2011, Curso Virtual de Simusol. Facultad de Ciencias Exactas, UNSa, http://moodlexa.unsa.edu.ar/course/view.php?id=16

Murdock, I. (2007). How package management changed everything. http://ianmurdock.com/solaris/how-package-management-changed-everything

Nullsoft (2009). Nullsoft Scriptable Install System (NSIS) http://nsis.sourceforge.net/Main_Page

FSF (2010). GnuMake, http://www.gnu.org/software/make/

Facebook,(2011), Grupo Simusol. https://www.facebook.com/groups/237446849651219/

Google (2011a), Lista de correo de usuarios. http://groups.google.com/group/simusol?hl=es su dirección es: simusol@googlegroups.com

Google (2011b), Lista de correo de desarrolladores http://groups.google.com/group/simusol-develop?hl=es su dirección es: simusol-develop@googlegroups.com

Ollivier, Kevin (2012), WxWidgets, Cross Plataform GUI Library http://www.wxwidgets.org/ O'Reilly (2012). The Perl Programming Language. http://www.perl.org

Saravia, Diego. Alía de Saravia, Dolores. (2010). Mejoras al Sceptre. El paquete Sceptre INENCO. AVERMA. Avances en Energías Renovables y Medio Ambiente, Vol. 14, 2010 pp8.77. http://www.cricyt.edu.ar/lahv/asades/modulos/averma/trabajos/2010/2010-t008-a010.pd

Saravia Diego, (2010). Instalación de Software Libre en Windows: Psicro, Simusol y Sceptre. Interfaz gráfica para Simusol y Sceptre basada en XUL. AVERMA. Avances en Energias Renovables y Medio Ambiente. Vol. 14 2010 pp8.77. http://www.cricyt.edu.ar/lahv/asades/modulos/averma/trabajos/2010/2010-t008-a012.pdf

Saravia, Diego (2010b). SumaPack http://www.sumapack.org

Saravia, Diego (2011). Comprehensive SimuSol Archive Network. CSSAN Repositorio de paquetes para SimuSol, http://cssan.simusol.org.ar

Saravia, Diego, (2011b) Simusol-make. http://simusol.org.ar/downloads/Makefile

Saravia, Diego (2011c), Simpack. https://simusol.org.ar/downloads/simpack-ultimo.tgz

Saravia, Diego (2012). SUMAPACK: Herramientas para la compilación, empaquetado e instalación de software libre en diferentes arquitecturas, sistemas operativos y distribuciones GNU/Linux, Journal of Free Software and Free Knowledge Vol 1, No 1. http://www.icfoss.org/ojs/index.php/foss/article/view/4 tambien en AVERMA Avances en Energias Renovables y Medio Ambiente. Vol. 14 2010 pp8.77 http://www.cricyt.edu.ar/lahv/asades/modulos/averma/trabajos/2010/2010-t008-a011.pd

Saravia, Diego 2012, LiberaWin. http://www.simusol.org.ar/downloads/LiberaWin.exe

Wikipedia (2010d). Sistema de gestión de paquetes. http://es.wikipedia.org/wiki/Sistema_de_gesti%C3%B3n_de_paquetes

Wikipedia, (2012). Open Document. http://en.wikipedia.org/wiki/OpenDocument

YouTube (2011), Canal para Simusol http://www.youtube.com/user/simusol2011

Algunas formas extras para los diagramas http://dia-installer.de/doc/en/custom-shapes-chapter.html#N41016