UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE ESTUDIOS SUPERIORES ARAGÓN

"EJERCICIO PROFESIONAL EN LA FACULTAD DE CIENCIAS UNAM DE 2003 AL 2016"

T R A B A J O E S C R I T O
EN LA MODALIDAD DE INFORME DEL EJERCICIO PROFESIONAL
QUE PARA OBTENER EL TÍTULO DE:
I N G E N I E R O E N C O M P U T A C I Ó N
P R E S E N T A :

K A R L A I V O N N E
G O N Z Á L E Z R O S A S

ASESOR: M. EN C. MARCELO PÉREZ MEDEL

MÉXICO, 2017.

Dedico el presente trabajo con mucho cariño:

- A mi hija Frida Sofía, la pequeña luz de mi vida, mi mayor ejemplo de fuerza y vida, mi motivación para ser mejor día a día.
- A mi madre María Antonia, gracias por tu amor y apoyo constantes, así como tus grandes esfuerzos y sacrificios para asegurar nuestros estudios. Por encausarme e impulsarme a conseguir más allá de mis metas, por ser un ejemplo de dedicación y trabajo arduo.
- A mi padre Francisco Javier, al igual que mi tía Isabel y mi primo Darwin, por su confianza, amor y apoyo.
- A mis hermanos Francisco, Javier y María José, cómplices y confidentes, por permitirme compartir sus alegrías y tristezas, gracias por estar a mi lado cuando más los he necesitado.
- A Antonio Carrillo Ledesma, compañero de trabajo y amigo por varios años, quién me animó a continuar, definir y ampliar el presente trabajo. En especial ahora como compañero de vida, por su amor, paciencia, cuidados, cariño y comprensión, por ser un ejemplo de constancia, dedicación, pasión y entrega.
- A mis amigos por su amistad incondicional. En especial quiero agradecer a Martín, Gustavo, Circe, Cynthia, Sandra, Berenice, Michelle, Regina, Terpsícore, Getsemaní, Rocío, Rosa María y Norma, gracias por hacer el camino más grato.

AGRADECIMIENTOS

La realización y culminación del presente trabajo fue posible gracias al apoyo de múltiples personas que me acompañaron a lo largo del desarrollo del mismo, sin el cual no podría haber concluido este objetivo en mi vida.

- Quiero agradecer especialmente a mi tutor, el M. en C. Marcelo Pérez Medel, quién a pesar del tiempo transcurrido siempre creyera en mí, por brindarme su tiempo, conocimientos y amistad, por compartir su alegría y buena voluntad. A la formación recibida en las materias que con tanta dedicación imparte, así como la asesoría y enseñanzas a lo largo de las prácticas de servicio social realizadas en el Centro Tecnológico de la FES, campus Aragón, UNAM.
- A mis compañeros de trabajo, a quienes agradezco su amistad, apoyo, paciencia y enseñanzas. A través de nuestra interacción se afianzaron mis conocimientos en el área de Computación.
- A mis profesores presentes y a los ya ausentes, de quienes adquirí conocimientos invaluables dentro y fuera del aula. Un especial agradecimiento a los profesores Ricardo Gutiérrez Orozco y Luis Ramírez Flores.

CAPÍTULO 1. CONTENIDO

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

INTRODUCCIÓN

El presente trabajo es para obtener el título de Ingeniero en Computación mediante la modalidad de informe del ejercicio profesional. Para ello, hago un recorrido en las labores profesionales que he tenido principalmente en la Facultad de Ciencias de la Universidad Nacional Autónoma de México (UNAM).

Dichas labores, han sido desarrolladas durante los últimos 13 años, estas fueron iniciadas como ayudante de profesor de asignatura nivel “B” interino para después ser profesora de asignatura nivel “A” interino; a la par, fui técnica académica auxiliar “A” de medio tiempo interino y el más reciente como técnica académica asociada “A” de tiempo completo interino.

En los siguientes capítulos, expondré las actividades básicas que he ido aprendiendo y realizando como parte de las actividades académicas y de apoyo a la docencia e investigación durante mi vida laboral y estas, gracias a la formación profesional recibida en la carrera de Ingeniería en Computación en la Facultad de Estudios Superiores campus Aragón de la Universidad Nacional Autónoma de México, en la cual adquirí las herramientas necesarias para un desempeño óptimo en mi vida laboral.

Para cumplir con los objetivos del presente trabajo, inicio con una descripción breve de las funciones y actividades que la Facultad de Ciencias de la UNAM tiene a su cargo, y como mi labor académica apoya a dichas funciones en las labores que tengo encomendadas.

Facultad de Ciencias, UNAM

La Facultad de Ciencias de la UNAM (véase [1]) es una institución de educación superior que tiene como finalidad formar profesionales y científicos que realicen investigación, que eleven la cultura científica del país, inicialmente en las áreas de Biología, Física y Matemáticas, Actuaría, Ciencias Ambientales, Ciencias de la Computación, Manejo Sustentable de Zonas Costeras, Física Médica, Matemáticas Aplicadas y Ciencias de la Tierra (véase [2]).

Además de realizar investigación, docencia y difusión; una tarea primordial es hacer la divulgación que permita elevar la cultura científica del país; y propiciar la vinculación con los sectores de la sociedad.

La Facultad de Ciencias tiene una larga historia con más de 75 años, en este tiempo ha pasado por múltiples cambios, siempre para crecer y para cumplir con las múltiples responsabilidades que le han sido confiadas por la sociedad, una breve reseña histórica se da a continuación.

Reseña Histórica[1]

1942. Se fundó el Instituto de Matemáticas por acuerdo del rector Rodulfo Brito Foucher. Nápoles Gándara fue nombrado director. Los primeros investigadores fueron Alberto Barajas y Roberto Vázquez García (1915-1994), en Matemática Pura, Francisco Zubieta Russi (1911-2005), en Lógica Matemática y Carlos Graef, en Matemática Aplicada.

1943. A mediados de este año se concretizó en la Ciudad de México la Sociedad Matemática Mexicana. Al mismo tiempo se inició la publicación del Boletín de la Sociedad Matemática Mexicana en donde escribirían connotados matemáticos como George Birkhoff (1844-1944), Roberto Vázquez García (1915-1978), Francisco Zubieta Russi, Manuel Sandoval Vallarta(1899-1977) y otros más.

[1] http://www.fciencias.unam.mx/nosotros/historia/Index

1947. Los ingenieros Velarde y Solórzano, propusieron al ingeniero Nápoles Gándara la creación de la carrera de Actuaría, bajo el argumento de que un actuario debía saber más matemáticas que las que se estudiaban en otras escuelas.

1953. La Facultad de Ciencias se traslada de la escuela de ingenieros a la Ciudad Universitaria. Sin embargo su situación era bastante precaria, poseía una biblioteca muy pequeña, y el número de libros de física y matemáticas eran mínimos. El plan de estudios estaba formado por materias anuales. Fue Carrillo el que se dedicó a conseguir recursos económicos para la Universidad, pero particularmente para impulsar el desarrollo de la ciencia.

1965. Según el Dr. Samuel Gitler (1923), se contaba solamente con 25 doctores en matemáticas que hacían investigación, 5 de ellos habían sido egresados de la Facultad de Ciencias, que además había otorgado 75 licenciados en matemáticas.

1968. El Departamento de Matemáticas tenía 336 estudiantes de Matemáticas (Cepeda, 2006), y en el Posgrado de Matemáticas estaban inscritos alrededor de 30 alumnos. Para ese mismo año, se habían recibido 75 matemáticos y un alto porcentaje de los investigadores de los institutos de Física y Matemáticas eran egresados de la Facultad.

1970. Se contrataron cuatro profesores más: José Luis Abreu, Pablo Barrera, Ángel Carrillo y Helga Fetter. Las contrataciones siguieron aumentando a partir de 1974 y para 1978 se tenían 71 profesores de tiempo completo en el Departamento de Matemáticas.

1993. Se amplió el número de carreras ofrecidas hasta entonces en el Departamento de Matemáticas con la creación de la carrera de Ciencias de la Computación.

1994. Se inauguró el edificio de la Facultad de Ciencias en el Circuito Exterior. Este poseería una gran biblioteca (Amoxcalli) y tres nuevos edificios para docencia y laboratorios los cuales fueron inaugurados en 1998 el Tlahuizcalpan, 2015 el Yelizcalli y el último en el 2016.

La Facultad de Ciencias de la UNAM es una institución de educación superior que tiene como finalidad formar profesionales y científicos que realicen investigación, que eleven la cultura científica del país, inicialmente en las áreas de Biología, Física y Matemáticas, Actuaría, Ciencias Ambientales, Ciencias de la Computación, Manejo Sustentable de Zonas Costeras, Física Médica, Matemáticas Aplicadas y Ciencias de la Tierra.

Además de realizar investigación, docencia y difusión; una tarea primordial es hacer la divulgación que permita elevar la cultura científica del país; y propiciar la vinculación con los sectores de la sociedad.

Para cumplir con estos fines, la Facultad de Ciencias está organizada en Departamentos apoyados en un Consejo Técnico y la Administración (véase [1]). El personal académico de la Facultad de Ciencias está dividido en los Departamentos de:

- Biología Celular
- Biología Comparada
- Biología Evolutiva
- Ecología y Recursos Naturales
- Física
- Matemáticas

en el campus de Ciudad Universitaria, y en dos Unidades Multidisciplinarias de Docencia e Investigación foráneas (véase [2]). Y atiende a las carreras de:

- Actuaría
- Biología
- Ciencias de la Computación
- Ciencias de la Tierra
- Física

- Física Biomédica
- Manejo Sustentable de Zonas Costeras
- Matemáticas
- Matemáticas Aplicadas

además de las carreras, participa en programas de especializaciones, maestrías y doctorados.

En la actualidad[2] la Facultad de Ciencias cuenta con un número aproximado de 319 Profesores de Tiempo Completo, 194 Técnicos Académicos de Tiempo Completo, 1381 Profesores de Asignatura y 988 Ayudantes de Profesor.

Departamento de Matemáticas de la Facultad de Ciencias

La misión del Departamento de Matemáticas[3] -al cual estoy adscrita- es formar y preparar recursos humanos profesionales capaces de desempeñar docencia e investigación en la materia, a cualquier nivel académico y desarrollar investigación teórica y aplicada, que contribuya al desarrollo del país y de la docencia en su conjunto, difundiendo al efecto el material pertinente.

Así mismo, debe brindar con la mayor calidad las asignaturas correspondientes a las licenciaturas de la facultad; así como apoyar en lo posible a los programas de postgrado en los que participa (véase [3]).

Actualmente el Departamento de Matemáticas brinda cursos de matemáticas y cómputo a estudiantes de las carreras de:

- Actuaría
- Biología
- Ciencias de la Computación

[2] Plan de desarrollo institucional de la Facultad de Ciencias 2014-2018, http://www.fciencias.unam.mx/nosotros/plan/Index

[3] Misión del Departamento de Matemáticas, http://www.matematicas.unam.mx/index.php/nosotros/mision

- Ciencias de la Tierra
- Física
- Matemáticas
- Matemáticas Aplicadas

y se apoya con Profesores de los institutos de investigación como el Instituto de Matemáticas (IMATE), el Instituto de Física, el Instituto de Investigaciones en Matemáticas y Sistemas (IIMAS) y otras dependencias responsables de los Posgrados que en el área de matemáticas se ofrecen.

Mi labor académica en la Facultad de Ciencias es actualmente en apoyo a la docencia en el edificio de investigación y docencia experimental Tlahuizcalpan, en el cual, proporcionamos las herramientas computacionales necesarias para que más de cien grupos semestralmente usen el equipo de cómputo acorde a sus necesidades académicas como parte de su formación curricular de todas las carreras que apoya el Departamento de Matemáticas.

Las principales actividades que he desarrollado las enmarco en objetivos generales y particulares de mi labor académica en la Facultad de Ciencias, las cuales detallo a continuación.

Objetivo General

Mostrar las actividades que he realizado en los últimos 13 años de labor académica y el desarrollo profesional que he tenido y su impacto en el apoyo a la docencia en el área de trabajo en que me desempeño.

Objetivos Particulares

- Mostrar como la formación en la carrera de Ingeniería en Computación ayudó a familiarizarme con el entorno de trabajo.
- Detallar las actividades realizadas en cada puesto ocupado en la Facultad de Ciencias.
- Exponer los retos encontrados en los mismos y
- Describir las soluciones encontradas a dichos retos.

Para cumplir con estos objetivos, en el capítulo 1, describiré las actividades desarrolladas durante el tiempo en que fui Ayudante de Profesor de Asignatura nivel B y Profesor Asociado de Asignatura nivel A, así como de las materias vistas en dicho período y mi aprendizaje en las mismas; en el capítulo 2, comentaré acerca de mis inicios como Técnica Académica Auxiliar “A” medio tiempo, las actividades desarrolladas, algunos de los principales problemas y las soluciones encontradas; en el capítulo 3, mostraré las actividades que realice y realizo como Técnica Académica Asociada “A” tiempo completo; y concluyo, mostrando como la carrera de Ingeniería en Computación me dio las herramientas necesarias para participar en el proceso integral de proporcionar un marco adecuado y flexible de tecnologías de la información acorde a las necesidades de los académicos, investigadores y estudiantes que requieren el uso cotidiano de equipo de cómputo para prácticas académicas que se realizan en el edificio de docencia experimental Tlahuizcalpan que cuenta con veintiún talleres, aulas y laboratorios; y más recientemente se complementan con seis aulas más en el edificio Yelizcalli.

CAPÍTULO 1. Ayudante de Profesor De Asignatura Nivel B Interino, Profesor Asociado de Asignatura Nivel “A” Interino

Dentro de mi trayectoria profesional como pasante de Ingeniería en Computación en la Facultad de Ciencias de la Universidad Nacional Autónoma de México (UNAM), inicié como Ayudante de profesor de los años 2002 a 2008 y como profesor asociado en 2006, en el Departamento de Matemáticas.

Para poner en contexto la dinámica de mi trabajo en dicho Departamento, a continuación hago una breve semblanza del mismo.

Departamento de Matemáticas de la Facultad de Ciencias

El Departamento de Matemáticas de la Facultad de Ciencias de la UNAM, es uno de los principales departamentos de matemáticas en México, cuenta con cuatro carreras, las cuales están bajo su responsabilidad directa que son:

- Actuaría
- Ciencias de la Computación
- Matemáticas y
- Matemáticas Aplicadas

y apoya con cursos básicos y avanzados de matemáticas y cómputo a las carreras de:

- Biología
- Física
- Física Biomédica
- Ciencias de la Tierra

Para aunar un poco en el tema, a continuación esbozo brevemente los perfiles profesionales de las carreras asociadas al Departamento de Matemáticas, que

son los siguientes:

Actuaría

Perfil Profesional: Plan de estudios (véase [4]). Los actuarios son profesionistas que estudian, plantean, formulan y aplican modelos de contenido matemático, acerca de los fenómenos que involucran riesgos, con el fin de proveer información para la planeación, previsión y la toma de decisiones. Los egresados están capacitados para intervenir en ámbitos que van desde el demográfico y financiero hasta el ecológico y administrativo y para interactuar con los profesionistas que ahí se desempeñen. Su campo de trabajo está en los sectores públicos o de la administración pública descentralizada, así como en el sector privado en compañías aseguradoras, despachos de consultoría actuarial y estadística, de cómputo e informática y de finanzas.

Ciencias de la Computación

Perfil Profesional: Plan de estudios (véase [5]). El profesional de Ciencias de la Computación se podrá desempeñar en el sector público o privado y en instituciones de educación media superior o superior, en donde tendrá la capacidad, habilidad y destreza para realizar las siguientes tareas:

- Diseño e implementación de sistemas de Software, supervisión de otros programadores, manteniéndolos alertas al surgimiento de nuevos enfoques.
- Diseño de nuevas maneras de utilizar computadoras, estudiando algoritmos eficientes para Robótica, Bioinformática, redes sociales, entre otros, participando en proyectos o desarrollos de otras disciplinas como la biología, geografía, meteorología, por nombrar algunos de éstos.
- Desarrollo de mecanismos efectivos para resolver problemas computacionales, como el almacenamiento masivo de información, el despliegue de imágenes complejas, la comunicación entre diversos sistemas.
- Proseguir con un posgrado en Ciencias de la Computación, ya sea en el país o en el extranjero.

- Realizar actividades docentes en el nivel de bachillerato o licenciatura.
- Participar en proyectos de investigación o desarrollo tecnológico en los que se requieran procesos computarizados.
- Además, tendrá la capacidad para enfrentar retos serios de programación.

Matemáticas

Perfil Profesional: Plan de estudios (véase [6]). El matemático es el profesional capacitado para planear y ejercer la docencia de las matemáticas a todos los niveles, para llevar a cabo investigación pura en alguna rama de las matemáticas, o bien investigación aplicada en equipos interdisciplinarios que incluyan profesionistas de otras áreas como biólogos, médicos, economistas, financieros, etc., y también está capacitado para integrarse al aparato productivo a través de asesorías que permitan resolver problemas como optimización de recursos, cálculo de probabilidades, aproximación de resultados, organización y creación de proyectos, etc.

Matemáticas Aplicadas

Perfil Profesional: Plan de estudios (véase [7]). El egresado será un profesionista con:

- Una formación matemática sólida, con una preparación general en las áreas de más frecuente aplicación (Computación Científica, Ecuaciones Diferenciales, Estadística, Investigación de Operaciones y Probabilidad, entre otras) y conocimientos más especializados en una de las áreas de concentración que se ofrecen.
- Conocimientos en computación, incluyendo uno o varios lenguajes de programación, métodos numéricos y análisis de datos.
- Habilidad para la instrumentación computacional y el uso de nuevas herramientas de cómputo.
- Cultura general en otras disciplinas de ciencia y tecnología. En particular, experiencia con modelos matemáticos en distintas áreas.

- Experiencia y habilidad para la descripción de procesos y fenómenos en términos matemáticos.
- Experiencia y habilidad para colaborar con profesionales de otras disciplinas en la formulación de problemas reales en términos matemáticos y en su resolución o en la determinación del tipo de matemáticas requeridas.
- Flexibilidad para abordar y resolver problemas de distintos orígenes y en el uso de diversas herramientas matemáticas y computacionales.
- Capacidad, por su formación, de colaborar en el desarrollo de enfoques, modelos y procedimientos novedosos.
- Formación que le permita acceder a programas de posgrado afines.

En este contexto, inicie mis labores como ayudante de profesor, cabe destacar que las actividades de un ayudante de profesor son, principalmente coadyuvar para reforzar los temas vistos en clase por el profesor, la realización de prácticas asociadas al tema visto, así como resolución de dudas. Todo esto bajo la supervisión del profesor titular de la materia.

Nótese que para ser ayudante de profesor Nivel “B”, se debe tener el 100% de créditos de la carrera afín a la materia que se impartirá, así como un común acuerdo con el profesor titular del curso para trabajar de forma conjunta como apoyo durante el semestre.

Ayudante de Profesor de Asignatura Nivel “B” Interino

Inicié a trabajar en la Facultad de Ciencias de la UNAM, en el semestre 2003-1 como ayudante de profesor en las materias de Programación I (véase [B.1]) y Programación II (véase [B.2]) para la carrera de Actuaría y optativas para las carreras de Matemáticas y Física en la Facultad de Ciencias de la UNAM, y continué haciéndolo ininterrumpidamente hasta el semestre 2008-2.

Entre las diversas labores que tenía encomendadas, en los semestres 2003-1 a 2004-2 vi conceptos y realizamos prácticas de programación lineal en lenguaje C[4] para la materia de Programación I, mientras que para la materia de Programación II se utilizó el paradigma de programación estructurada (véase [8]) usando el lenguaje C (véase [9]). En ambas materias ayudé en la elaboración de la página Web y prácticas en laboratorio de cómputo, para el reforzamiento del conocimiento adquirido por los alumnos; a partir del semestre 2005-1 al 2008-2, con la intención de actualizar la enseñanza se comenzó a utilizar el paradigma de Programación Orientada a Objetos (véase [10])[5] en lenguaje C#[6] (véase [11]) en la materia de Programación I y para la materia de Programación II el manejo de estructura de datos con Programación Orientada a Objetos en lenguaje C# usando herramientas de libre distribución para su programación por parte de los alumnos, también participé en la elaboración de la página Web y prácticas de ambas materias.

Así mismo en el semestre 2005-2 fui ayudante en la materia de Base de Datos (véase [B.3]) para la carrera de Actuaría y Matemáticas en la Facultad de Ciencias de la UNAM, donde dimos una introducción a Bases de Datos (véase [12]) utilizando SQL[7], colaboré en la elaboración de la página Web del curso, así como en las prácticas de cómputo asociadas.

---

[4] C es un lenguaje diseñado por Dennis Ritchie y Brian Kernighan para ser el lenguaje de los sistemas operativos UNIX, es un lenguaje de nivel medio que combina elementos de lenguajes de alto nivel (tipos, bloques, etc.) con la funcionalidad de los ensambladores.

[5] Object-Oriented Programming (OOP). La Programación Orientada a Objetos (POO) es una forma de programar, trata de acercar como expresaríamos las cosas en la vida real, para escribir los programas en términos de objetos, propiedades, métodos, entre otros términos manejados en este paradigma.

[6] C# es uno de los lenguajes de programación de alto nivel que pertenecen al paquete .NET (paquete de desarrollo diseñado por Microsoft para simplificar el desarrollo de aplicaciones para Internet). Es una evolución de C/C++. Con él se pueden escribir tanto programas convencionales como para Internet.

[7] Structured Query Lenguaje (Lenguaje de Consulta Estructurado), lenguaje estándar de comunicación entre los diferentes motores existentes, esta implementado en motores de Bases de Datos.

Como parte de mi formación profesional, busqué cursos que me permitieran actualizar mis conocimientos en el área. La UNAM contaba con distintas células del programa MICROSOFT Academic Alliance, en ella, se capacitaba a profesores, ayudantes y alumnos en el uso y manejo de los productos de Microsoft. Éstos cursos fueron impartidos en el Laboratorio de Microsoft de la Facultad de Ingeniería, los cuales tomé con vista a una certificación por ser parte de la célula de Microsoft de la Facultad de Ciencias (la cual no se concluyó por problemas operativos de la Facultad). Dentro de este contexto, tomé los siguientes cursos de certificación:

Curso: 2389B Programming with Microsoft Ado.Net[8]
Construcción de aplicaciones centradas en datos y en servicios Web usando Microsoft ADO.NET, Microsoft SQL[9] Server 2000, y .NET Framework[10] (véase [13]).

Curso: 2071B Queryng Microsoft SQL Server 2000 with Transact-SQL[11]
Usos y formas del lenguaje Transact-SQL, uso de herramientas de consultas, escribir consultas SELECT para obtener datos, agrupar y resumir los datos al utilizar Transact-SQL, unir datos de diferentes tablas, modificación de datos en las tablas, consultar campos de texto con búsquedas de texto completas, describir como crear objetos de programación.

Curso: 2555A Developing Microsoft .Net Applications for Windows (Visual C# .Net) Construcción de Formas y aplicaciones utilizando Microsoft Framework, con lenguaje C#.

[8] https://msdn.microsoft.com/es-mx/library/e80y5yhx(v=vs.110).aspx

[9] https://msdn.microsoft.com/es-mx/library/bb545450.aspx

[10] https://msdn.microsoft.com/es-mx/library/zw4w595w(v=vs.110).aspx

[11] https://msdn.microsoft.com/es-mx/library/bb510741.aspx

Curso: 2124C Programming with C#

Revisión de los elementos principales del .NET Framework, la postura de C# dentro de la plataforma de .NET, análisis de la estructura básica de una aplicación de C#, documentar, depurar, compilar y ejecutar una aplicación simple, crear, nombrar y asignar valores a variables.

Así como:

Curso: Introducción a la programación orientada a objetos. [12]

Donde vimos fundamentos de la programación orientada a objetos usando Java.

Dichos cursos, fueron un complemento para la enseñanza de las materias de Programación I y II, así como la de Bases de Datos correspondientes a la carrera de Actuaría y optativas para las carreras de Física y Matemáticas. Adicionalmente, se impartieron dos cursos, uno en la Facultad de Ciencias orientado para profesores y otro más que impartió en el III Congreso Nacional de Informática y Ciencias de la Computación, realizado en noviembre de 2007 en la Universidad Autónoma de Sinaloa, en el cual participé con el Taller: “Primeros pasos en C #”.

Parte del buen desempeño como ayudante de profesor, lo debo a la formación recibida, durante mis estudios en la carrera de Ingeniería en Computación, la mayoría de los conocimientos adquiridos que utilicé como ayudante de profesor, los obtuve de las materias de Computadoras y Programación, Bases de Datos, Lenguajes Formales y Autómatas, Programación Orientada a Objetos, Diseño y Análisis de Algoritmos, Estructura de Datos, Estructuras Discretas y Sistemas Operativos. Estos fueron base fundamental para lograr una mejor comprensión y manejo de los temarios de los cursos de Programación I y II, Bases de Datos, mismas que me ayudaron a una mejor comprensión y aprovechamiento de los conocimientos adquiridos en los cursos para certificación.

[12] Dicho curso fue impartido en la UNAM, en la DGSCA como parte del Programa de Actualización y Superación Docente (PASD).

Retos Pedagógicos Encontrados en los Alumnos

En la carrera de Actuaría, los primeros grupos con que me tocó trabajar al principio era complicado acercarlos a la materia de Programación, debido a que las materias se daban en el 3º y 4º semestre respectivamente y había un poco de renuencia por parte de algunos alumnos, principalmente en la materia de Programación I, insistían en que la materia no les apoyaría en su formación profesional, la mayoría indicaban que les atraía el área de seguros, otros más, tenían una cierta aversión o miedo por desconocimiento a la materia, el resto, debido a que se les dificultaba comprender los temas vistos en la misma.

Así que, el primer reto era acercarlos a el área, indicándoles las ventajas que podrían obtener al utilizar las técnicas de programación como herramienta de apoyo a varias materias que cursarían más adelante en su formación académica, como Investigación de Operaciones, y que como complemento a la formación que reciben durante la carrera en temas que no fuesen de cómputo, el usar las herramientas computacionales les ayudaría a analizar y resolver problemas complejos dentro del área de su elección para complementar su desarrollo como profesionales en el área de Actuaría.

Una vez superado este reto, tocaba familiarizarlos con el uso de los equipos de cómputo, ya que en ese tiempo aún había alumnos que no tenían tanto contacto con ellos, así como guiarles con ayuda de alguna técnica y lenguaje de programación a plantear y resolver un problema mediante el análisis, diseño y la programación del mismo.

Uno de los principales problemas al comenzar a elaborar las prácticas, se daba al comenzar a escribir los primeros programas, debido a que al no estar familiarizados con la sintaxis utilizada en el lenguaje, era común que tuvieran pequeños errores gramaticales, que no identificaban fácilmente, en muchos casos eran cosas tan sencillas como la falta de un “;” al termino de una instrucción o alguna llave de apertura o cerradura “{}” para indicar el inicio o fin de algún bloque de instrucciones. En ocasiones se les olvidaba escribir la notación designada por el lenguaje de programación utilizado para indicar que se trataba

de un comentario, o el uso del signo de “=” para indicar la asignación del valor a una variable, entre otros problemas de ese tipo.

En la materia de Bases de Datos, que es una materia optativa impartida a grupos de 6º a 8º semestres en la carrera de Actuaría no se tenía dicho problema, pues los alumnos la eligen, ya que consideran que esta será de utilidad en su formación para el área que hayan elegido, la mayoría de ellos llegan con conocimientos previos de un lenguaje de programación, gracias a los cursos de Programación I y II, así como, las prácticas de algunas materias, como Investigación de Operaciones, resultándoles así, relativamente más sencillo el comenzar a interactuar con el diseño y uso de bases de datos.

Motivación de los Alumnos

Con la mayoría de los grupos y alumnos, fue sencillo lograr una comprensión de la técnica y el lenguaje utilizado, pero en algunos casos era difícil lograr esa afinidad debido a varios factores, que van desde la familiarización con el uso de computadoras, la renuencia a la materia, dificultad en la asociación de las técnicas de programación o el lenguaje utilizado.

Fue importante elegir una bibliografía actualizada y adecuada, ya que es la base de información y consulta extra a la clase, misma que se usó como complemento para resolución de dudas o ampliar aún más los conocimientos adquiridos en clase.

Independientemente de la bibliografía utilizada, el material desarrollado en las páginas Web fue un gran apoyo en el transcurso de cada semestre, servía como refuerzo a las clases, debido a que en ocasiones algunos alumnos faltan a estas y de alguna forma podían ponerse al tanto, consultando el material puesto en línea del tema visto y no atrasarse en la materia.

En algunos casos, fue un gran apoyo el uso de analogías para ejemplificar de una forma física, como por ejemplo, el hecho de que una caja de zapatos funja como una estructura que contiene un sobre representando una instrucción, así como el

tradicional juego del "teléfono" formado por un par de vasos unidos por un cordón/estambre, con los cuales podíamos representar un par de funciones que comparten dato(s) (punteros a funciones, funciones con parámetros) transformándolo/conservándolo e interpretándolo como una función o clase independiente, o un cartón de huevo visto a modo de una matriz, según el lenguaje utilizado.

De esta forma logramos una mayor motivación al momento de que la mayoría de los alumnos comprendieran los conceptos de los que hablábamos en clase, utilizando objetos con los que estamos familiarizados, así mismo, gracias a la interacción para ejemplificar los conceptos se lograba una integración a nivel grupal, así como una constancia de asistencia y participación en clase.

Me dio gusto ver que varios alumnos de la carrera de Actuaría, después de aprender y tomar gusto por la programación, aunado a la formación integral de la misma, al egresar, se colocaron en puestos donde programar es una parte vital de sus actividades profesionales.

Estrategias Educativas

Como parte de la evaluación en las materias, se les pedía la entrega de ejercicios prácticos, para ello, se daba un ejemplo en clase. No todos eran terminados en clase y se les pedía continuar el trabajo en casa y de esta forma detectar si había dudas, además de conocer la forma en que cada alumno analiza, diseña y programa un problema particular.

Para tratar de evitar que sólo algunos alumnos resolvieran los ejercicios y el resto sólo copiara y los reenviara con su nombre, se pedía que se entregaran por correo y con comentarios, algunos ejercicios eran iniciados y terminados en clase con la colaboración de todo el grupo como lluvia de ideas o de forma individual, otros con distintas opciones a resolver en casa, todos con una hora y fecha de entrega.

Se complementaban las clases con lecturas que se pedían realizar antes de la

clase, para acercarlos al tema que se vería. Así mismo se diseñaban proyectos finales, en los cuales se hacia uso de todos los conceptos vistos en clase para resolver un problema cotidiano, mismo que se trabajaba en equipo, debido a la complejidad del mismo, era necesario que todos los integrantes del equipo participaran en la solución, ya que la entrega y asignación de calificación final era de forma individual.

De la misma forma, se realizaban exámenes denominados "sorpresa", puesto que no se avisaba de los mismos, estos exámenes carecían de valor para la evaluación final, pero servían como referencia a nosotros los profesores, para conocer a cada alumno y principalmente como evaluación del avance grupal y así, saber los temas o conceptos que era necesario reforzar.

Al final, cada actividad realizada en clase era tomada en cuenta como un porcentaje para conformar la calificación final, en ocasiones también se elaboraban exámenes finales, para permitirles aprobar o mejorar la calificación de una sección vista. Intentando que al evaluar a cada alumno, este contara con los conocimientos mínimos descritos dentro de los objetivos del plan de estudios de la materia.

Actividades Desarrolladas en Esta Plaza

Las actividades que realicé en estas plazas han sido:

A) Programación I

Las actividades académicas principales realizadas en la materia de Programación I (véase Apéndice B.1) de la carrera de Actuaría en la Facultad de Ciencias de la UNAM, durante los semestres 2003-1 a 2004-2, fue basada en ver conceptos y realizar prácticas de programación lineal en lenguaje C; a partir del semestre 2005-1 al 2008-2 con la intención de actualizar la metodología de enseñanza, comenzamos a utilizar el paradigma de Programación Orientada a Objetos en lenguaje C#. A continuación haré una breve descripción de dichas formas de programación:

a) Programación Lineal

El nombre de programación lineal no procede de la creación de programas de computadora, sino de un término militar, en el que programar significa 'realizar planes o propuestas de tiempo para el entrenamiento, la logística o el despliegue de las unidades de combate'. Es una técnica matemática utilizada para dar solución a problemas que se plantean comúnmente en diversas áreas, mediante un conjunto de técnicas racionales de análisis y de resolución de problemas con el objetivo de ayudar a los responsables en las decisiones sobre asuntos en los que interviene un gran número de variables, tratando de maximizar y/o minimizar una función lineal de dos o más variables teniendo en cuenta que las mismas deben cumplir determinadas exigencias derivadas de la escasez de recursos disponibles en la realidad.

La investigación de operaciones en general y la programación lineal en particular recibieron un gran impulso gracias a las computadoras. Uno de los momentos más importantes fue la aparición del método simplex, este método, desarrollado por G. B. Dantzig en 1947, consiste en la utilización de un algoritmo para optimizar el valor de la función objetivo teniendo en cuenta las restricciones planteadas; este tipo de análisis se utiliza en casos donde intervienen hasta cientos e incluso miles de variables (véase [14]). Por otro lado, las nuevas técnicas de manejo de estructuras de datos son la base para aumentar la eficiencia de los métodos de solución de este tipo de problemas (véase [15]).

Ventajas:

- Se pueden formular o aproximar una gran variedad de problemas en diversos campos como modelos lineales.
- Permite comparar un amplio rango de soluciones alternativas y analizar sus consecuencias requiriendo para ello poco tiempo.
- Indica como emplear de forma más eficaz los factores, seleccionándolos y distribuyéndolos adecuadamente.
- Ayuda a ser más objetivos en las decisiones al obtener todos los

datos que puedan ser útiles para la formulación matemática del problema.

Desventajas:

- No es aplicable la paralelización del código.
- Dificulta la comprensión de lectura.

Estrategias Docentes: Debido a la formación matemática que proporciona la carrera de Actuaría desde los primeros semestres y a la factibilidad de plantear un problema mediante la Programación Lineal, era sencillo acercar a los alumnos al manejo del análisis de las variables y su interacción. Así, mediante el uso de estas técnicas matemáticas se les facilitaba la comprensión y mediante el uso del lenguaje de programación C se logró hacer la resolución de los problemas planteados. Para facilitar la comprensión, diseño y análisis de los ejercicios, era de gran ayuda el uso de diagramas de flujo[13].

b) Programación Estructurada

Inicialmente, en los primeros semestres que se impartió el curso de Programación II, se utilizaba como metodología de enseñanza la Programación Estructurada usando como lenguaje de programación el lenguaje C.

La programación estructurada es un paradigma de programación que intenta mejorar la claridad, calidad y tiempo de desarrollo de un programa de computadora, utilizando únicamente subrutinas y tres estructuras: secuencias, selección (if y switch) e iteración (bucles for y while) considerando innecesario el uso de la instrucción de transferencia incondicional.

Esta forma de programación se convirtió una de las formas de programar

13 El diagrama de flujo o diagrama de actividades es la representación gráfica de un algoritmo o proceso.

más extendida. Aplicando la técnica de "divide y vencerás" por medio de módulos se forma un árbol que integrara el programa completo, cuya raíz es el programa principal que implementa la solución al problema que afrontamos utilizando uno o varios módulos que realizan partes de dicha solución por sí solos o invocando a su vez otros módulos que solucionan subproblemas más específicos. A esta aproximación se le denomina diseño descendente o top-down.

El carácter auto contenido de los módulos o librerías hace que pueda ocultarse el funcionamiento interno de las funciones contenidas en un módulo, ya que para utilizarlas basta con saber con qué nombre y argumentos se invocan y qué tipo de valores devuelven. Al reunirlas en un módulo, se realza la relación entre las mismas separándolas del resto del programa.

Ventajas:

- Es mucho más fácil comprender completamente el trabajo que realiza una función determinada si todas las instrucciones que influyen en su acción están físicamente contiguas y encerradas por un bloque.
- La facilidad de lectura, de comienzo a fin, es una consecuencia de utilizar solamente tres estructuras de control, y de eliminar la instrucción de transferencia de control goto.
- Resolver varios subproblemas de forma aislada es con frecuencia mejor que el de abordar el problema global.
- Se mejora con respecto a otras formas de programación el trabajo simultáneo por distintos grupos de programadores. Debido a que los subprogramas son más sencillos, permite documentar el código de forma modular.
- Si está bien programado, posibilita la reutilización del código (especialmente de alguno de los módulos) en futuras aplicaciones.
- La opción de encapsular datos y operaciones en librerías facilita que se puedan utilizar en una aplicación ajena.

- La abstracción facilita la comprensión de la división de un programa en partes y la encapsulación permite aislar la implementación de cada una de las partes, aumentando la estabilidad y reutilización del código.
- Los programas pequeños y medianos en cuanto al número de subrutinas son más fáciles de leer y entender y por ende permite un mejor mantenimiento del código en su ciclo de vida.

Desventaja:

- Se debe de hacer un buen análisis y diseño de la aplicación, además hacer una correcta encapsulación de datos y funciones, de lo contrario, conforme aumenta el tamaño del código y/o subrutinas se hace problemático el manejo del código fuente y su mantenimiento futuro.

c) Programación Orientada a Objetos

Según Bjarne Stroustrup, autor del lenguaje de programación C++, para que un lenguaje se llame a sí mismo orientado a objetos debe soportar tres conceptos: objetos, clases y herencia. Sin embargo, ha llegado a pensarse más comúnmente que los lenguajes orientados a objetos son lenguajes construidos sobre el trípoide encapsulación, herencia y polimorfismo. La razón de este cambio de filosofía es que con el paso de los años se ha llegado a dar cuenta de que la encapsulación y el polimorfismo son partes tan integrantes de la construcción de sistemas orientados a objetos como la clase y la herencia (véase [16]).

Encapsulamiento

Se produce al combinar en un objeto los datos con las operaciones que se pueden ejecutar sobre los mismos y establecer sus partes públicas, accesibles al usuario, y privadas, no accesibles, lo que impide usos indebidos. Las partes no accesibles al usuario se dice que están encapsuladas en la clase. En general los datos miembros de un objeto son privados y para acceder a ellos se utilizan los comportamientos del objeto.

Herencia

La herencia es la capacidad para crear nuevas clases (descendientes) que se construyen sobre otras existentes, permitiendo que estas les transmitan sus propiedades. En programación orientada a objetos la reutilización de código se efectúa creando una subclase que constituye una restricción o extensión de la clase base o superclase, de la cual hereda sus propiedades.

La herencia es una de las características más importantes en la Programación Orientada a Objetos (POO) porque permite que una clase herede los atributos y métodos de otra clase. Esta característica garantiza la reutilización del código.

Con la herencia todas las clases están clasificadas en una jerarquía estricta. Cada clase tiene su superclase (la clase superior en la jerarquía, también llamada clase base), y cada clase puede tener una o más subclases (las clases inferiores en la jerarquía; también llamadas clases derivadas).

Las clases que están en la parte inferior en la jerarquía se dice que heredan de las clases que están en la parte superior en la jerarquía.

Polimorfismo

El polimorfismo consigue que un mismo mensaje pueda actuar sobre diferentes tipos de objetos y comportarse de modo distinto. Adquiere su máxima expresión en la derivación o extensión de clases; es decir cuando se obtienen nuevas clases a partir de una ya existente mediante la propiedad de derivación de clases o herencia.

Es la funcionalidad que permite al código antiguo invocar nuevo código. Este es probablemente el mayor beneficio de la programación orientada a objetos, porque le permite extender o mejorar su sistema sin romper o modificar el código existente.

Ventajas:

- Es una manera diferente de pensar en cómo diseñar y programar soluciones a los problemas, se describe de forma más clara el problema ya que se ajusta a descripciones de objetos de la vida cotidiana.
- El aprendizaje es más sencillo debido a la manera en que la POO ve a un programa como un conjunto de objetos que dialogan entre sí, para realizar las distintas tareas para las que ha sido escrito.
- Gracias a la flexibilidad que tiene, es mucho más fácil organizar, revisar y reutilizar código para programas grandes.

Desventaja:

- Cuando se maneja otra metodología de programación es necesario “desaprender” hábitos de programación para entender el uso de POO.

Estrategias Docentes: En ambas metodologías (Programación estructurada y Programación Orientada a Objetos), después de realizar un poco de análisis y diseño; con ayuda de la técnica de programación lineal, el cambio metodología de programación era relativamente sencillo, ya que solo era cuestión de comenzar a describir la resolución del programa con ayuda de lenguaje de programación siguiendo las reglas de sintaxis y de la propia metodología de programación, mediante la resolución de ejercicios prácticos, iniciando con ejercicios sencillos con los cuales comenzábamos a ver el uso de variables e instrucciones básicas.

El mayor problema era la familiarización de alumnos recursadores que habían llevado otro lenguaje en la materia de Programación I, la mayoría de las veces al inicio se tenían problemas básicos de sintaxis al comenzar a usar nuevo el lenguaje, pero poco a poco se familiarizaban y se les hacía cada vez más fácil el identificar y corregir los errores.

Después mediante ejercicios más complejos, a los alumnos se les facilitaba

la familiarización con los términos nuevos, para lo cual también nos apoyábamos en elementos de uso común como cajas de zapatos, hojas, canicas, estambre, globos, agua, vasos, pintura vegetal para hacer analogías respecto a los temas.

Cuando se utilizaba la metodología de Programación Estructurada era de gran ayuda analizar las posibles soluciones de los programas mediante diagramas de flujo, así mismo en la metodología de Programación Orientada a Objetos con la ayuda de diagramas UML (véase [17])[14].

B) Programación II

En la materia de Programación II (véase Apéndice B.2) durante los semestres 2003-1 a 2004-2 utilizamos el paradigma de programación estructurada utilizando lenguaje C; a partir del semestre 2005-1 al 2008-2 con la intención de actualizar la enseñanza comenzamos a utilizar el manejo de estructura de datos con Programación Orientada a Objetos en lenguaje C#.

Así mismo, en el semestre 2006-1 fui profesora titular en la materia de Programación II para la carrera de Actuaría en la Facultad de Ciencias de la UNAM, donde vimos el manejo de estructura de datos con Programación Orientada a Objetos utilizando el lenguaje C#, para la cual elaboré diversas prácticas y una página Web como apoyo a los temas vistos en el curso.

C) Base de Datos

En el semestre 2005-2 fui ayudante en la materia de Base de Datos (véase Apéndice B.3) para la carrera de Actuaría, en la Facultad de Ciencias de la UNAM, donde dimos una introducción a Bases de Datos utilizando SQL, también colaboré en la elaboración de la página del curso.

Un sistema gestor de bases de datos (SGBD) consiste en una colección de datos

14 Lenguaje Unificado de Modelado (LUM o UML, por sus siglas en inglés, Unified Modeling Language) es el lenguaje de modelado de sistemas de Software más conocido y utilizado en la actualidad. Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema.

interrelacionados y un conjunto de programas para acceder a dichos datos. La colección de datos, normalmente denominada base de datos, contiene información relevante para una empresa. El objetivo principal de un SGBD es proporcionar una forma de almacenar y recuperar la información de una base de datos de manera que sea tanto práctica como eficiente (véase [12], página 1).

Se diseñan para gestionar grandes cantidades de información, la gestión de los datos implica la definición de estructuras para almacenar la información como la provisión de mecanismos para la manipulación de la información.

Una parte importante de la Modelación Matemática es trabajar con datos de prueba, lo más cercano posible a la realidad, por lo que, es necesario contar con mecanismos para almacenar, editar y consultar una cantidad grande de datos, esto se logra usando las bases de datos.

Así el SGBD forma un conjunto de programas que permiten almacenar, modificar y extraer la información en una base de datos, además de proporcionar herramientas para añadir, borrar modificar y analizar los datos.

Los usuarios pueden acceder a la información usando herramientas específicas de interrogación y de generación de informes, o bien mediante aplicaciones.

Los SGBD también proporcionan métodos para mantener la integridad de los datos, para administrar el acceso de usuarios a los datos y recuperar la información si el sistema se corrompe, permiten presentar la información de la base de datos en varios formatos, la mayoría incluyen un generador de informes, también puede incluir un módulo gráfico que permita presentar la información con gráficos y diagramas.

Hay muchos tipos de SGBD distintos según manejen los datos y muchos tamaños distintos según funcionen sobre ordenadores personales y con poca memoria a grandes sistemas que funcionan en mainframes con sistemas de almacenamiento especiales.

Existe una gran variedad de paquetes para el manejo de base de datos los cuales existen tanto en las plataformas de Windows, Linux, Mac, de los que decidimos utilizar en clase fue MySQL[15], ya que al igual que en las materias de Programación I y II, se utilizaban herramientas de Software de libre distribución para su implementación.

Estrategias Docentes: Debido a que es optativa, los alumnos eligen la materia ya que consideran que les será útil para su formación en el área que hayan elegido, la mayoría llegan con conocimientos previos de un lenguaje de programación gracias a los cursos de Programación I y II, así como las prácticas en algunas materias, como Investigación de Operaciones, por lo cual es relativamente intuitiva la interacción con el diseño y uso de bases de datos. Aunado a que se desarrollaban en el curso ejercicios y proyectos lo más parecidos al mundo laboral, permitió que los alumnos comprendieran y manejaran el material indicado en el plan de estudios de la materia.

Los problemas más comunes que encontramos en la docencia, al inicio del curso era con la sintaxis utilizada y el análisis de como estructurar de forma adecuada las solicitudes al SGBD para manipular la información de las bases de datos y obtener los resultados deseados.

Profesor Asociado de Asignatura Nivel “A” interino

Para ser Profesor Asociado de Asignatura nivel “A” interino en la Facultad de Ciencias se debía tener el título o conocimientos equivalentes a nivel licenciatura de una carrera afín a la materia que se impartirá, así como el ser asignado por el comité académico de la carrera de Actuaría. Además, cuando se cuenta con un

15 MySQL es un sistema de gestión de bases de datos relacional, multihilo y multiusuario. Desarrollado por MySQL AB como Software libre en un esquema de licenciamiento dual. Por un lado se ofrece bajo la GNU GPL para cualquier uso compatible con esta licencia, pero para aquellas empresas que quieran incorporarlo en productos privativos deben comprar a la empresa una licencia específica que les permita este uso. Está desarrollado en su mayor parte en ANSI C.

número mínimo de alumnos (designado por el comité académico de la carrera) se tiene el apoyo de un ayudante de profesor para trabajar de forma conjunta.

En el semestre 2006-1 fui profesor titular en la materia de Programación II para la carrera de Actuaría en la Facultad de Ciencias de la UNAM, donde vimos el manejo de estructuras de datos implementando los programas mediante el paradigma de Programación Orientada a Objetos utilizando el lenguaje C#, para la cual elaboré diversas prácticas y una página Web como apoyo a los temas vistos en el curso.

Entre las diversas labores que tenía planeadas para la materia de Programación II, era continuar utilizando el paradigma de Programación Orientada a Objetos en lenguaje C# usando herramientas de libre distribución para su programación por parte de los alumnos.

Con los conocimientos adquiridos en algunos de los cursos de certificación mencionados en la sección 1.1, y con la preparación previa que adquirí como ayudante de profesor, decidí complementar mi formación académica en el área de enseñanza, tomando los siguientes cursos:

> **Curso:** (Integración y Cultura Universitaria) El cuerpo en el aula y la práctica docente en la relación maestro-alumno[16]
>
> **Curso:** La comunicación en el salón de clases a partir de las habilidades expresivas del docente.

Con dichos cursos, adquirí y mejoré mis habilidades para la docencia.

Actitud de los Alumnos

En la Facultad de Ciencias de la UNAM, una gran cantidad de alumnos suelen

16 Impartidos en la UNAM, en la Facultad de Filosofía y Letras, como parte del Programa de Actualización y Superación Docente (PASD).

abandonar los cursos por diversas razones, algunas ajenas a sus capacidades académicas. Por ello, al iniciar el curso de Programación II, es muy común encontrar alumnos con muy diversos conocimientos con respecto a la materia, además de que la mayoría de ellos no manejaban el lenguaje C#.

A sabiendas de estos hechos, y ya que la mayoría de los alumnos eran recursadores, fue necesario homogenizar los conocimientos de los alumnos y detectar las dificultades que cada uno de ellos tenia. Sin dicha homogenización a varios de ellos se les hubiese dificultado la comprensión de los temas designados en el plan de estudios de la carrera.

Motivación de los Alumnos

El primer reto era acercarlos a el área, indicándoles las ventajas que podrían obtener al utilizar las técnicas de programación como herramienta de apoyo a varias materias que cursarían más adelante como Investigación de Operaciones, y que como complemento a la formación que reciben durante la carrera en temas que no fuesen de cómputo, el usar las herramientas computacionales les ayudaría a analizar y resolver problemas complejos dentro del área de su elección para complementar su desarrollo como profesionales en el área de Actuaría.

Uno de los principales problemas al comenzar a elaborar las prácticas, se daba al comenzar a escribir los primeros programas, debido a que al no estar familiarizados con la sintaxis utilizada en el lenguaje, era común que tuvieran pequeños errores gramáticos, que no identificaban fácilmente, comúnmente les faltaba un “;” al termino de una instrucción o una “{}” para indicar el inicio o fin de algún bloque de instrucciones, en ocasiones se les olvidaba escribir la notación designada por el lenguaje de programación utilizado para indicar que se trataba de un comentario, o el uso del signo de “=” para indicar la asignación del valor a una variable, entre otros problemas de ese tipo.

Estrategias Educativas

En la metodología de Programación Orientada a Objetos, después de realizar un poco de análisis y diseño; con ayuda de la técnica de programación lineal, el

cambio de metodología de programación era relativamente sencillo, ya que solo era cuestión de comenzar a describir la resolución del programa con ayuda de lenguaje de programación, siguiendo las reglas de sintaxis y de la propia metodología. Mediante la resolución de ejercicios prácticos, iniciando con ejercicios sencillos (con los cuales comenzábamos a ver el uso de variables e instrucciones básicas) y continuando con problemas cada vez más complejos, me permitió cumplir con los objetivos trazados en el plan de estudios de la materia.

El mayor problema era la familiarización de alumnos recursadores que habían llevado otro lenguaje en la materia de Programación I, la mayoría de las veces al inicio se tenían problemas básicos de sintaxis al comenzar a usar nuevo el lenguaje, pero poco a poco se familiarizaban y se les hacía cada vez más fácil el identificar y corregir los errores. Después mediante ejercicios más complejos, se les facilitaba la familiarización con los términos nuevos, para lo cual también nos apoyábamos en elementos de uso común como cajas de zapatos, hojas, canicas, estambre, globos, agua, vasos, pintura vegetal para hacer analogías respecto a los temas.

Cuando se utilizaba la metodología de Programación Estructurada era de gran ayuda analizar las posibles soluciones de los programas mediante diagramas de flujo, así mismo en la metodología de Programación Orientada a Objetos con la ayuda de diagramas UML.

Actividades Desarrolladas en Esta Plaza

Las actividades académicas principales realizadas en la materia de Programación II de la carrera de Actuaría en la Facultad de Ciencias de la UNAM, fue ver conceptos y realizar prácticas de Programación Orientada a Objetos en lenguaje C#. Además de realizar material de apoyo, como la página Web del curso y dedicar tiempo fuera de clase para apoyo individualizado a aquellos alumnos que lo requiriesen.

CAPÍTULO 2. Técnica Académica Auxiliar “A” Medio Tiempo Interino

Ingresé como Técnica Académica Auxiliar “A” medio tiempo en la Facultad de Ciencias de la Universidad Nacional Autónoma de México (UNAM) el primero de marzo del 2003 y permanecí en el puesto hasta el siete de octubre del 2004, periodo en el que estuve adscrita a la Sala 3 de cómputo ubicada en el edificio Amoxcalli, como parte de un equipo de 2 Técnicos Académicos asignados para la administración y mantenimiento de dicha sala, la cual contaba con 31 equipos de cómputo y daba servicio a múltiples grupos de las carreras del Departamento de Matemáticas en horario continuo de 7 a 21 hrs., así como apoyo a proyectos como PEUVI (Programa de Extensión Universitaria y Vinculación), mi horario de servicio fue de 12 a 16 hrs.

Actividades Realizadas en Esta Plaza

Como parte del mantenimiento integral de los equipos de cómputo del aula, realizábamos revisiones periódicas para detectar fallas, así como revisión de la integridad del sistema operativo y de esta forma garantizar el buen funcionamiento de los equipos de cómputo, también dábamos solución a los problemas reportados por los usuarios respecto al funcionamiento del equipo en cuanto al Hardware o Software. Así, mediante la evaluación de los recursos disponibles se llevaba a cabo una planeación de soluciones para satisfacer las necesidades de cómputo de los docentes y alumnos.

Algunos de los principales problemas detectados y reportados eran a nivel Hardware, principalmente por fallas en fuente de poder, tarjeta madre, memoria, disco duro, en las unidades lectoras de 31/2 o CD-ROM, ratón o monitor, en los que las soluciones normalmente eran dadas mediante una revisión en la cual se verificaban que los dispositivos estuvieran bien conectados, así como, una revisión de los cables, ya que era común que sufrieran desconexión o tuvieran un falso contacto, dándole mantenimiento correctivo al equipo o ante la falla inminente del mismo realizar el cambio del dispositivo.

A nivel Software el mantenimiento se comenzaba antes de iniciar el semestre donde se instalaba el sistema operativo Windows 98 (véase [18])[17], el Software solicitado por los usuarios y antivirus; durante el semestre a veces era necesario reinstalar los equipos por mal funcionamiento o por problemas de virus, también continuábamos realizando nuevas instalaciones de Software a petición de los usuarios.

Desde el inicio del servicio, el uso de equipos de cómputo ha generado una alta demanda de Software propietario (véase Apéndice A.2 y [19],[20]), a la par con el alto e incremental costo de algunos paquetes, así como el hecho de que no se cuenta con un presupuesto asignado para licencias de dicho Software. En ocasiones, algunas solicitudes se han cubierto con parte de otros presupuestos para comprar algunas licencias educativas o al hacer convenios con algunas compañías de Software, pero esto no ha sido suficiente para satisfacer todas las solicitudes de los usuarios, por lo que se comenzó a buscar Software libre (véase Apéndice A.1 y [19],[20]) que tuviera una utilidad similar al Software utilizado en la práctica, esto como una alternativa para llevar a cabo las prácticas programadas por los profesores.

Para una mejor comprensión del Software utilizado en las materias de las carreras en Ciencias de la Computación, Actuaría así como en la de Matemáticas, fue de gran ayuda la formación recibida en las materias de Computadoras y Programación, Programación Orientada a Objetos, Estructuras de Datos, Lenguajes Formales y Autómatas, Compiladores, Diseño y Análisis de Algoritmos, Sistemas Operativos, Redes de Computadoras, Microprocesadores y Microcontroladores de la carrera de Ingeniería en Computación.

Así también, el análisis y detección de fallas a nivel Hardware ha sido más sencillo gracias a los conocimientos adquiridos en las materias y prácticas de laboratorio como Electricidad y Magnetismo, Análisis de Circuitos Eléctricos,

[17] Windows 98 fue la quinta versión del sistema Operativo de Microsoft surgida en el año de 1998, incluía Internet Explorer 5 y era compatible con DVD y dispositivos USB.

Dispositivos Electrónicos, Medición e instrumentación, Diseño Lógico, Diseño de Sistemas Digitales, Redes de Computadoras, Microprocesadores y Microcontroladores, Procesamiento Digital de Señales, así como las prácticas realizadas durante mi servicio social.

Paquetes de Cómputo de Uso Común

Una de nuestras funciones básicas además de mantener los equipos de cómputo operativos, era la de buscar, conocer y recomendar programas propietarios y de Software libre; y en su caso apoyar a profesores y alumnos en el uso de características básicas como avanzadas de los paquetes. Dada la diversidad de cursos en la carreras de la Facultad de Ciencias y la gran cantidad de profesores y ayudantes de los mismos, los paquetes de cómputo solicitados es diverso, entre ellos destacan:

Java JRE y JDK	Microsoft SQL Server	Stata
SharpDevelop IDE	PostgreSQL	Statistica
DFD	SPSS	Octave
Turbo C IDE	PSPP	FreeMat
Developer Studio-Fortran IDE	SAS	Maple
Microsoft Visual Studio	MatLab	Mathematica
Microsoft Windows SDK	Vensim PLE	Compresores y descompresores de archivos Winzip, WinRAR, 7-zip
Compaq Visual Fortran	Statgraphics	Navegadores de páginas Web, como Internet Explorer o Mozilla
Microsoft Office	GPower	Adobe Reader
MathType	EViews	SSH Secure File Transfer
Scientific WorkPlace	Systat	

Tabla 1 Algunos paquetes de cómputo de uso común usados en las carreras de la Facultad de Ciencias

El tipo de programas usados, las plataformas de ejecución soportada y las diferentes versiones de un mismo paquete que solicitan los profesores y ayudantes crecen constantemente, así mismo, su complejidad y en su caso, el costo monetario de las licencias de uso también se incrementa.

Los paquetes comerciales en general proveen un ambiente integrado de trabajo que pueden ser usados en la preparación de estudiantes para aplicar sus conocimientos al egresar en las diversas áreas de las carreras que imparte la UNAM, particularmente en las carreras de Ciencias, esto les permite laborar en empresas pequeñas, medianas y grandes con un mínimo de capacitación técnica adicional.

Además, en un mercado tan competitivo como el actual, las organizaciones centran sus recursos en las estrategias más adecuadas para conducir a la compañía hacia el éxito. Los paquetes propietarios y los incipientes paquetes de Software libre pueden ayudar a conseguir este objetivo, completando la inversión ya realizada en sistemas operacionales.

Pero el hecho de que las organizaciones actuales, manejan una gran cantidad de información, la cual puede o no estar dispersa en sus múltiples sistemas operacionales, requieren usar paquetes propietarios que tengan integrado el manejo de las grandes bases de datos distribuidas o centralizadas, esta integración ofrece beneficios adicionales.

Por otro lado, notemos que, una vez que un producto de Software libre ha empezado a circular, rápidamente está disponible sin costo o a un costo muy bajo. Al mismo tiempo, su utilidad no decrece. El Software, en general, podría ser considerado un bien de uso inagotable, tomando en cuenta que su costo marginal es pequeño y que no es un bien sujeto a rivalidad (la posesión del bien por un agente económico no impide que otro lo posea).

Puesto que el Software libre permite el libre uso, modificación y redistribución, a menudo encuentra un hogar entre usuarios para los cuales el coste del Software

no libre es a veces prohibitivo, o como alternativa a la piratería, con las implicaciones legales que tiene. También es sencillo modificarlo localmente, lo que permite que sean posibles los esfuerzos de traducción a idiomas que no son necesariamente rentables comercialmente.

La mayoría del Software libre se produce por equipos internacionales que cooperan a través de la libre asociación. Los equipos de desarrollo están típicamente compuestos por individuos con una amplia variedad de motivaciones, y pueden provenir tanto del sector privado, del sector voluntario o del sector público.

En México el Software Libre nació en las Universidades y los Centros de Investigación. Es por eso que, desde hace tres décadas, los estudiantes y los profesores usan Software libre para fines didácticos y de investigación. Las universidades suelen optar por el uso de Software libre en vez de utilizar Software privativo porque satisface de una mejor manera sus necesidades de cómputo, dada su naturaleza de apertura del código y la libertad de compartir los resultados obtenidos. De forma colateral, no se tienen gastos adicionales derivados del pago de licenciamientos.

En forma burda los paquetes usados en los cursos de la Facultad de Ciencias pueden ser clasificados en:

- Programas de Cálculo Numérico
- Programas de Estadística
- Programas Ofimáticos
- Otros Programas

A continuación se da una breve descripción de estos tipos de paquetes:

Paquetes de Cálculo Numérico

Los paquetes de cálculo numérico, son programas matemáticos que ofrecen un entorno de desarrollo integrado (IDE) con un lenguaje de programación propio con

un amplio abanico de herramientas numéricas para la lectura, manipulación, análisis y graficación de datos.

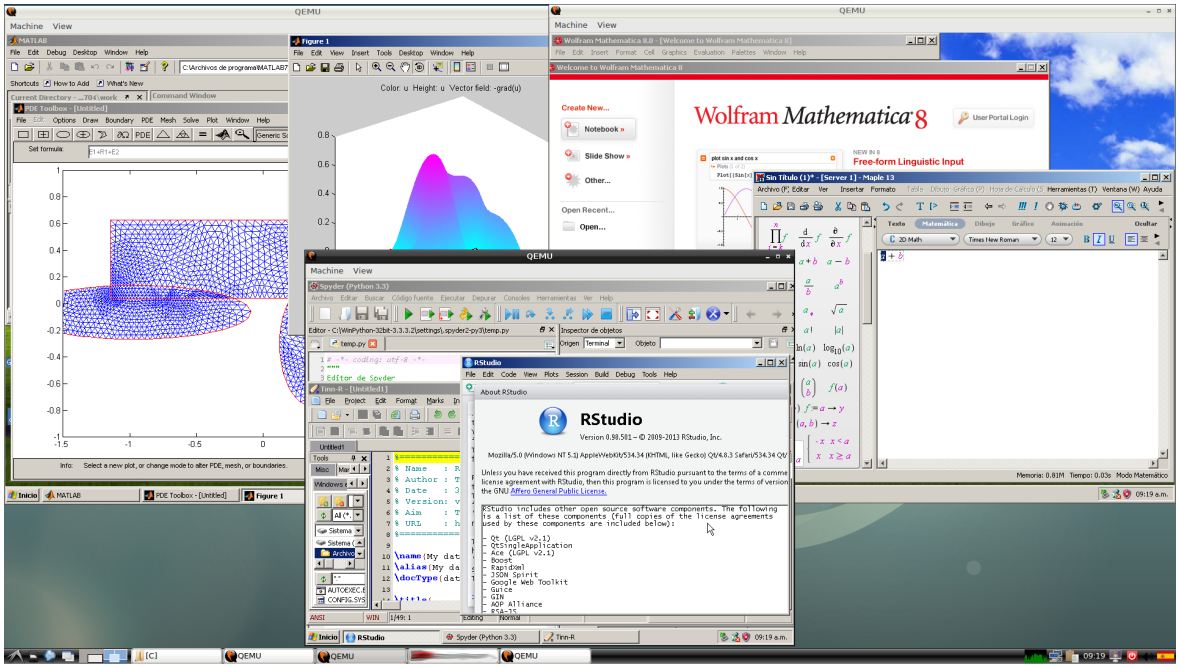

Figura 2.1 Distintos paquetes de Cálculo Numérico (Mathematica, MatLab y RStudio) corriendo dentro de múltiples máquinas virtuales simultáneamente bajo un anfitrión en Linux.

Entre sus prestaciones básicas se hallan:

- Manejo de números reales y complejos
- La manipulación de vectores y matrices tanto reales como complejas
- Manejo de funciones elementales y especiales
- Resolución de problemas de álgebra lineal
- Resolución de ecuaciones no lineales
- La representación de datos y funciones
- La implementación de algoritmos
- Integración de funciones
- Máximos y mínimos de funciones
- Manipulación de polinomios
- Integración de ecuaciones diferenciales
- Graficación de funciones en 2D y 3D

- La comunicación con programas en otros lenguajes de programación y con otros dispositivos Hardware.

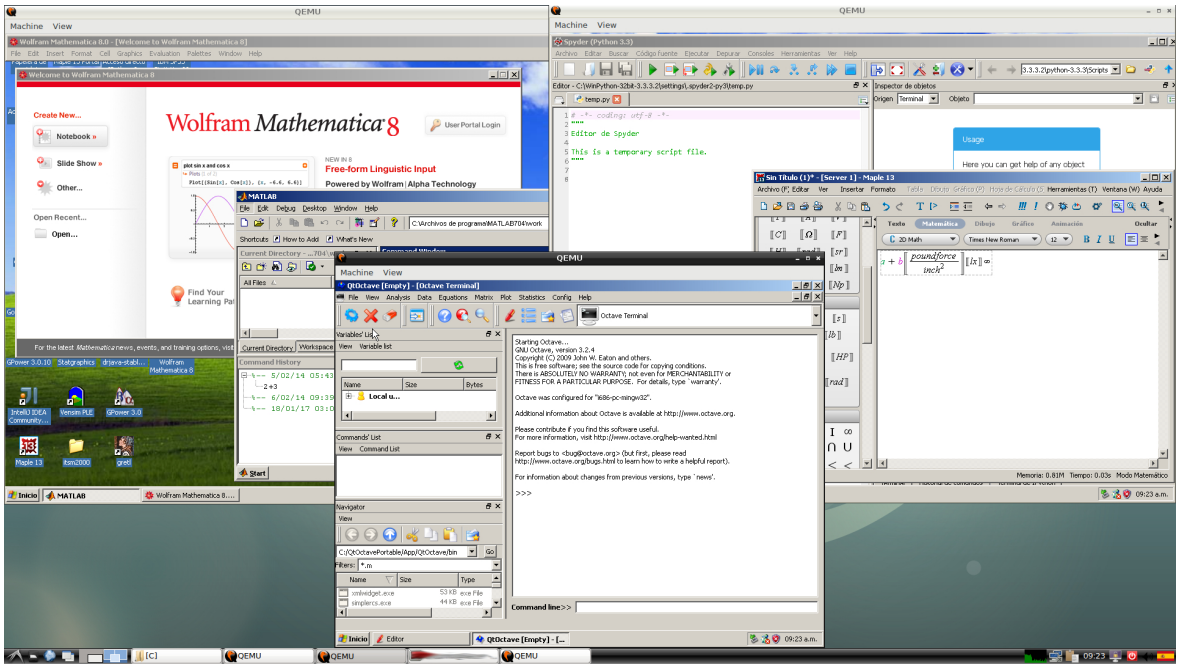

Figura 2.2 Distintos paquetes de Cálculo Numérico (Mathematica, Spyder-Python y QTOctave) corriendo dentro de múltiples máquinas virtuales simultáneamente bajo un anfitrión en Linux.

El programa comercial líder para el cálculo numérico es MatLab (véase [21],[22]) de la compañía MathWorks, este paquete salió a la venta en el año de 1984 con la versión 1.0 y casi año con año, ha generado nuevas versiones de su paquete y múltiples sistemas de licenciamiento.

Actualmente se comercializa la versión 7.14 (R2012a) del año 2012 con más de 8 tipos de licenciamiento. Está empresa cuenta con más de mil empleados y oficinas en más de doce países alrededor del mundo.

La idea detrás de paquetes como MatLab es la de emplear grupos de subrutinas escritas en principio en el lenguaje de programación FORTRAN (véase [23]) como son las librerías LINPACK (véase [24]) y EISPACK (véase [25]) para manipular matrices y vectores y proporcionar un sencillo acceso a dicho Software y así, ser

usado en los cursos de álgebra lineal y análisis numérico, sin necesidad de escribir programas en lenguajes de bajo nivel.

Estos paquetes, pueden disponer de herramientas adicionales que expanden sus prestaciones, como son el diseño y simulación de sistemas de control. Además, se pueden ampliar las capacidades base de dichos programas con las cajas de herramientas y con los paquetes de bloques. En algunos casos existen versiones para cómputo secuencial y paralelo -tanto en memoria compartida como distribuida, también para usar los múltiples Cores[18] gráficos CUDA (GPUs) (véase [26])[19] de las tarjetas NVIDIA (véase [27])[20] -.

Los paquetes de cómputo para el Cálculo Numérico más usados actualmente son:

- MatLab (abreviatura de MATrix LABoratory, "laboratorio de matrices") es un Software matemático que ofrece un entorno de desarrollo integrado (IDE) con un lenguaje de programación propio (lenguaje M). Entre sus prestaciones básicas se hallan: la manipulación de matrices, la representación de datos y funciones, la implementación de algoritmos, la creación de interfaces de usuario (GUI) y la comunicación con programas en otros lenguajes y con otros dispositivos Hardware. El paquete MatLab dispone de dos herramientas adicionales que expanden sus prestaciones, a saber, plataforma de simulación multidominio Simulink (véase [28]) y editor de interfaces de usuario GUIDE. Además, se pueden ampliar las capacidades de MatLab con las cajas de herramientas y las de Simulink con los paquetes de bloques.
- Octave (véase [29],[30]) es un programa de cálculo numérico de licencia GNU (véase [31]), conocido por buscar una sintaxis similar a la de MatLab,

[18] Número de núcleos que tiene un CPU

[19] Es una plataforma de cómputo paralelo y modelo de programación inventado por NVIDIA. Permite incrementos dramáticos en el rendimiento de la computación, aprovechando la potencia de la unidad de procesamiento grafico (GPU por sus siglas en inglés).

[20] Empresa multinacional especializada en el desarrollo de unidades de procesamiento gráfico y tecnologías de circuitos integrados para estaciones de trabajo, computadoras personales y dispositivos móviles.

existiendo una gran compatibilidad en las funciones de cálculo numérico.

- FreeMat (véase [32]) es un programa de cálculo numérico de licencia que proporciona un ambiente libre para el rápido desarrollo de prototipos para la Ciencia y la Ingeniería, además del procesamiento de datos. Es similar a MatLab y Octave, pero cuenta con una interfaz externa de código en los lenguajes de programación en C, C++ y Fortran, incluso distribuye el desarrollo de algoritmos en paralelo con la interfaz de paso de mensajes llamada MPI (véase [29],[33]) [21].
- Scilab (véase [34]) es un programa de cálculo numérico de licencia CeCILL compatible con GPL, desarrollado principalmente en Francia, que incluye su propia librería para gráficos. Es similar a MatLab, pero no busca una compatibilidad total, como lo puede hacer FreeMat y Octave. Scilab tiene una herramienta para el diseño y simulación de sistemas de control Scicos (véase [35]) similar a Simulink de MatLab.
- Scipy (véase [36]) es una librería de herramientas numéricas para Python (véase [37], [38]) con licencia Open Source. En su filosofía no ha tratado de imitar a ninguno de los paquetes anteriores y tiene detrás el respaldo de un auténtico lenguaje de programación orientado a objetos e interpretado que también puede ser compilado para ganar velocidad en la ejecución. Este hecho le confiere una gran potencia y la capacidad de beneficiarse de las mejoras del lenguaje base.

Existen otros paquetes que pueden ser usados en el cálculo numérico -estos poseen características que enriquecen las opciones clásicas de los paquetes de cálculo numérico-:

- R (véase [29], [39])
- Maple (véase [40])
- Mathematica (véase [41])
- Maxima (véase [29],[42])

[21] Es una especificación para los desarrolladores y usuarios de las bibliotecas de paso de mensajes. No es una biblioteca, sino la especificación de lo que debería ser una.

MatLab

El paquete MatLab tiene cientos de características, tan variadas como el segmento de usuarios al que dicho Software esta dirigido. Al ser un paquete tan completo, es difícil que un usuario promedio use las características avanzadas de dicho paquete; esto repercute en que el usuario promedio pague un alto costo por el valor de las licencias de uso sin usar dichas características; y esto contrasta con un vertiginoso desarrollo de nuevas características, que permite a la compañía lanzar una o más versiones por año, cada una con múltiples opciones de licenciamiento, según las necesidades del segmento al que están dirigidas.

Las múltiples características y los miles de usuarios, han creado una comunidad robusta, la que ha logrado una gran cantidad de funciones optimizadas, que han permitido la publicación de decenas libros, cientos de artículos y miles de páginas Web en los cuales se muestra como resolver diversos problemas de concomitantes en Ciencias e Ingenierías usando dicho paquete y la interacción con otros lenguajes como son C/C++, Fortran o Java (véase [43], [44])[22].

Entre las principales aplicaciones de MatLab incluyen la de métodos secuenciales y paralelos para resolver problemas de álgebra lineal con matrices (ralas, dispersas y densas), estadística, análisis de Fourier, optimización, integración numérica, resolución de ecuaciones diferenciales ordinarias y parciales, creación de gráficos y visualización de datos. Además de opciones para hacer interpolación y regresión de datos, cálculo de Eigen-valores y valores singulares, etc.

Entornos de Programación

Uno de los aspectos más agradables de MatLab es su entorno de programación, que permite centralizar la información en un entorno de ventanas. El depurador está también incorporado en el editor. Desgraciadamente, estas facilidades no se encuentran del todo desarrolladas en las otras herramientas -Octave, FreeMat,

[22] Es un lenguaje de programación y una plataforma informática comercializada por primera vez en 1995 por Sun Microsystems.

Scilab, Scipy-, pero es posible obtener la misma efectividad para las opciones básicas usando la línea de comandos.

Octave

El paquete Octave (véase [29],[30]), al mantener una sintaxis similar a la de MatLab, la gran mayoría de funcionalidades básicas son compatibles, pero no así, en el entorno de programación, este no tiene un ambiente propio de desarrollo y mucho menos un sistema de depuración. Existen proyectos de terceros que tratan de suplir dichas limitantes como en el ambiente de programación KOctave (véase [45]) y QtOctave (véase [46]). Pero desgraciadamente ninguno de ellos tiene el grado de avance esperado.

FreeMat

El paquete FreeMat (véase [32]), es un programa de cálculo numérico de licencia GPL que proporciona un ambiente libre para el rápido desarrollo de prototipos para la Ciencia y la Ingeniería, además del procesamiento de datos, además cuenta con una interfaz externa de código en los lenguajes de programación en C, C++ y Fortran, incluso distribuye el desarrollo de algoritmos en paralelo con la interfaz de paso de mensajes llamada MPI.

Es similar a MatLab, pero según sus desarrolladores es casi 95% compatible con MatLab, actualmente trabaja en Windows, Linux, MAC y algunas versiones de UNIX, las características soportadas son:

- Manipulación de arreglos N-dimensionales (N esta limitado a 6)
- Soporta enteros de 8, 16 y 32 bits y números de punto flotante de 32 y 64 bits y tipo complejos de 64 y 128 bits
- Soporta la aritmética de todos los tipos de datos soportados
- Resolución de sistemas de ecuaciones
- Descomposición de Eigen valores y valores singulares
- Arreglos heterogéneos
- Soporte completo para la estructura de arreglos dinámicos
- Soporte para FFT
- Paso por referencia

- Interfase para código externo C/C++/Fortran
- Soporte nativo para matrices dispersas
- Clases y sobrecarga de operadores
- Visualización de gráficos en dos y tres dimensiones

Algunas diferencias básicas son:

- Soporte parcial para construcción de Widgets/GUI
- FreeMat a MEX interfase para portar archivos de MEX de MatLab
- No implementación de GUI/Widgets

En cuanto al ambiente de desarrollo, FreeMat instala todo lo necesario para un trabajo cómodo y funcional. Ya que permite abrir el editor de texto en el cual podemos hacer la depuración del mismo, ejecutar los programas de demostración, acceder a las ventanas gráficas, controlar la ejecución en curso, abrir ficheros y ejecutarlos, observar el estado de las variables definidas, entre otras cosas.

Scilab

El paquete Scilab (véase [34]), tiene una licencia compatible con la de GPL, que por su madurez y la gran cantidad de usuarios, Universidades y Centros de Investigación que lo usan y dan retroalimentación del paquete, ha logrado ser un Software que permite hacer todas las opciones básicas de MatLab y algunas avanzadas también, es similar a MatLab y permite cómputo secuencial y paralelo, pero no busca una compatibilidad total, por ello hay diferencias entre ambos paquetes. Scilab provee una forma para tratar de importar código de MatLab y generar su equivalente en Scilab.

Algunas diferencias básicas son:

- Funciones: Las funciones en Scilab no son consideradas como archivos por separado -como archivos con extensión M en MatLab-, una o más funciones pueden ser definidas en un solo archivo y el nombre de archivo no necesariamente tiene que estar relacionado con el nombre de las funciones. También, la(s) función(es) no se cargarán automáticamente en

Scilab, como lo es en MatLab, después de que su nombre es invocado. Por lo general, se tiene que ejecutar el comando GETF("funcion_name") antes de ser capaz de utilizar la función. Las funciones también se pueden definir en línea con el comando deff, además en Scilab para ejecutar un archivo de comandos se debe de usar exec("archivo"), en MatLab solo es necesario escribir el nombre del archivo.

- Comentarios: En MatLab los comentarios inician con %, en Scilab los comentarios inician con //
- Variables: En Scilab las variables predefinidas usualmente tienen el prefijo % (%i, %inf, etc.) estas son protegidas y no pueden ser redefinidas. MatLab predefine a las variables i y j como la unidad imaginaria y pueden ocasionar problemas si se tratan de usar estas en índices de arreglos, esto no pasa en Scilab.
- Variables Booleanas: En Scilab las variables booleanas son %T, %F y 0, 1 en MatLab, ellas corresponden con los conceptos de verdadero y falso respectivamente. Haciendo uso de las variables booleanas pueden no producir el mismo valor en MatLab como en Scilab. Por ejemplo x=[1,2]; x([1,1]) [which is NOT x([%T,%T])] retorna [1,1] en SCILAB y [1,2] en MatLab. También si x es una matriz x(1:n,1)=[] or x(:)=[] no son validas en MatLab.
- Polinomios: En Scilab los polinomios y las matrices polinomiales son definidas por la función poly, ellas son consideradas como vectores de coeficientes en MatLab.
- Graficación: Excepto para los gráficos simples plot y mesh en MatLab y plot3D en Scilab, estos no son compatibles y no siempre es posible obtener gráficos con la calidad de MatLab.
- En Scilab existe un tipo de dato distinto, las listas, que en la práctica permiten definir una nueva clase. Aunque hay también celdas y estructuras como en MatLab, Scilab en sus últimas versiones recomienda usar las listas.

Estas y otras diferencias hacen que no sean compatibles los programas de MatLab en Scilab, pero si se usa como una plataforma en la cual se le enseñe a

los alumnos la sintaxis del paquete, Scilab proporciona todas las capacidades básicas de MatLab.

Por otro lado, Scilab tiene una interfaz para integrar funciones escritas en C o Fortran de manera relativamente sencilla, al menos para los tipos de datos habituales. También, puede ser llamado desde un programa en Fortran, C, C++ o Java las rutinas de Scilab, o incluso traducir un programa de Scilab a Fortran.

En cuanto al ambiente de desarrollo, Scilab instala todo lo necesario para un trabajo cómodo y funcional. Ya que permite abrir el editor de texto en el cual podemos hacer la depuración del mismo, ejecutar los programas de demostración, acceder a las ventanas gráficas, controlar la ejecución en curso, abrir ficheros y ejecutarlos, observar el estado de las variables definidas, entre otras cosas.

Scipy

El paquete Scipy (véase [36]), es imposible condensar en unas pocas páginas todas las posibilidades de un lenguaje de propósito general como Python. Al ser este un lenguaje interpretado, puede teclearse línea a línea desde la entrada de comandos, pero también es posible usar ficheros, estos por claridad suelen tener la extensión .py, éste podría ser ejecutado con la orden python nombre_fichero.py. En este sentido es similar a MatLab.

Ya que en su filosofía no ha tratado de imitar a ningún otro paquete y tiene detrás el respaldo de un auténtico lenguaje de programación orientado a objetos e interpretado que también puede ser compilado para ganar velocidad en la ejecución. Este hecho le confiere una gran potencia y la capacidad de beneficiarse de las mejoras del lenguaje base y posee herramientas secuenciales y paralelas -en memoria compartida y distribuida además de usar los Cores gráficos CUDA (GPUs) (véase [26]) de las tarjetas NVIDIA (véase [27])-.

Algunas diferencias básicas son:

- La sintaxis para introducir arreglos debe de incluir comas entre los

elementos. Además, debe haber tantos símbolos "[" como dimensiones del arreglo -y sus correspondientes cierres-.

- Los índices en los arreglos empiezan en 0.
- Es posible indicar el tipo de dato de cada elemento del arreglo en su creación.
- Existen verdaderas operaciones entre enteros, como en C.
- Salvo los tipos de datos muy simples, todos los demás objetos se pasan de una función lo hacen por referencia.
- En los arreglos de dos dimensiones las operaciones * y / se realizan elemento a elemento. La multiplicación de matrices y la multiplicación por la inversa deben hacerse con dot e inv. Pero también se puede usar el tipo matrix en lugar de un arreglo, donde se recupera una sintaxis similar a MatLab.

En cuanto al entorno de programación por omisión se tiene acceso a IDLE (véase [47]) es el entorno de desarrollo típico de Python. Permite compatibilizar una ventana interactiva con la edición de los programas. La ventana interactiva es similar a una consola normal de IPython (véase [48]), pero permite ver de forma gráfica los módulos accesibles, las clases y sus métodos, y sobre todo, una ventana de depuración que puede ser de gran ayuda. Existen otros entornos de programación como son Pydev (véase [49]), el cual es un programa que permite desarrollar código dentro del entorno de programación Eclipse (véase [29],[50]). Otra opción es usar The Eric Python IDE (véase [51]), es un auténtico entorno de desarrollo y gestión de proyectos de Python, este permite trabajar varias ventanas e incorpora el depurador.

Ventajas, Desventajas y Carencias

Notemos que el tener múltiples herramientas para realizar operaciones elementales y avanzadas de cálculo numérico, es en sí misma una gran ventaja. Para los centros universitarios y usuarios ocasionales, las herramientas de Software libre son una herramienta invaluable. En el caso de empresas o usuarios avanzados que requieren usar opciones especializadas o generadas por terceros,

MatLab destaca como la mejor herramienta de trabajo.

Pero hay que hacer notar que:

- Funcionalidades básicas: Todos los paquetes implementan las funcionalidades básicas, ya que todos llevan años desarrollándose.
- Funcionalidades especializadas: Por mucho, MatLab tiene implementadas cientos de funciones especializadas que pueden ser muy útiles para usuarios avanzados, pero rara vez son usados por los usuarios noveles o cotidianos.
- Gráficos e imágenes: Todos los paquetes permiten hacer gráficos en 2D y 3D y controlar las marcas, poner títulos, etc. Pero los que mejores prestaciones tienen para los usuarios finales son MatLab, FreeMat y Scilab.
- Potencia del lenguaje de programación: En este caso Scipy destaca por su modularidad, por tener una orientación a objetos más convencional, por su mayor precisión en los tipos de datos y porqué Python posee muchos módulos que permiten integrar otras tareas típicas de programación con el cálculo numérico.
- Fiabilidad: En los paquetes en desarrollo son comunes las caídas del programa, pero MatLab destaca por ser más fiable que los demás.
- Información: MatLab es el paquete con una abundante bibliografía y la propia ayuda del programa. FreeMat y Octave también tiene manuales aceptables y mucha información en Internet, aunque la ayuda es escasa. La documentación de Scipy es algo escasa, pero se esta trabajando en ella. Scilab tiene menos difusión y al final casi toda la documentación proviene de su página Web oficial, pero la ayuda del programa es buena.
- Facilidad de Manejo: Ninguno de los programas presenta grandes dificultades a la hora de su utilización. MatLab destaca por su entorno integrado, por las facilidades gráficas y por la opción de realizar acciones desde sus menús. Pero en menor o mayor medida, todos los paquetes presentan entornos de desarrollo funcional, pero perfectible.
- Costo: El costo de las diversas versiones de MatLab supera los dos mil pesos por licencia estudiantil, en el caso del Software libre, los paquetes se pueden descargar de la red sin más costo que el acceso a Internet.

Así, es posible resumir las características de los paquetes libres como:

- FreeMat y Octave son programas con una sintaxis muy similar a MatLab. Su uso no debe de suponer problemas para aquellos usuarios habituados a usar MatLab. En el caso de Octave quizás echará de menos algunas funciones, pero las que posee cubren un gran espectro de aplicaciones, y a partir de ellas se pueden implementar fácilmente las que falten, por otro lado, ambos proyectos son jóvenes y con gran empuje por parte de la comunidad GNU, lo cual, en algunos años permitirá tener productos maduros y de gran calidad, pero por el momento el desarrollo de la interfaz de usuario es algo limitada, al igual que su documentación, pero permiten hacer uso de los cada vez más comunes multiCores en los equipos de cómputo mediante el cómputo en paralelo.
- Scilab es también parecido a MatLab, pero dado que no busca tener una sintaxis similar, los nombres de las funciones pueden cambiar. Hay guías de paso de una herramienta a otra (véase [52]). Además, existe la utilidad de conversión directa de un fichero de MatLab a uno de Scilab. El comando se llama mfile2sci, aunque esta herramienta no es perfecta, es sin duda muy interesante contar con esta posibilidad.
- Scipy es el más diferente de MatLab, pero es un lenguaje rico en funcionalidades, también permite usar herramientas secuenciales y paralelas -en memoria compartida y distribuida además de usar los Cores gráficos CUDA (GPUs) de las tarjetas NVIDIA-. A pesar de que al principio puede parecer algo más pesado escribir con Scipy, pronto se comprueba la elegancia del lenguaje de programación y su lógica. Ser estrictos permite, a la larga, evitar errores y organizar mejor las ideas del programador. Además, hay que destacar los numerosos módulos de Python para tareas distintas a la del cálculo numérico: transmisión de datos por Internet, e-mail, manejo de bases de datos, creación de interfaces gráficas, etc.

Paquetes Estadísticos

Los paquetes estadísticos, son programas matemáticos que ofrecen un entorno de desarrollo integrado (IDE) con un lenguaje de programación propio con un

amplio abanico de herramientas la lectura, manipulación, análisis y graficación de datos estadísticos.

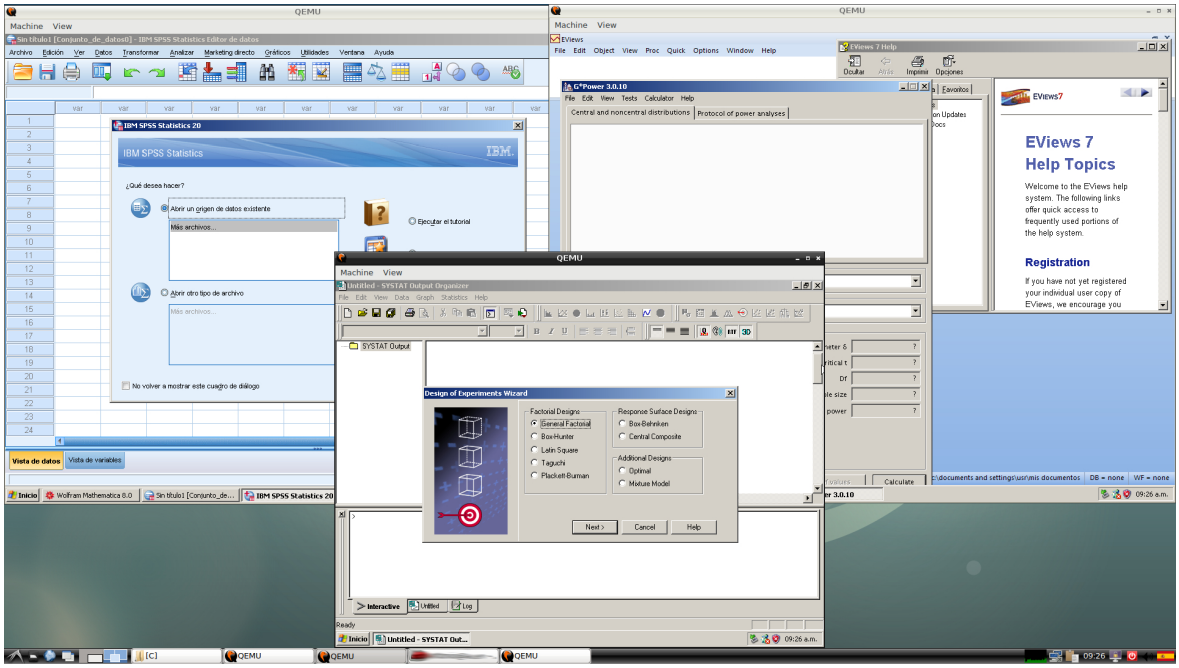

Figura 2.3 Distintos paquetes Estadísticos (IBM SPS, G*Power y Systat) corriendo dentro de múltiples máquinas virtuales simultáneamente bajo un anfitrión en Linux.

Entre sus prestaciones básicas destacan:

- Análisis de datos mediante operadores para cálculos sobre arreglos, matrices y/o Tablas
- Tablas Cruzadas
- Reordenamiento de Datos
- Análisis de la Varianza (ANOVA)
- Frecuencias
- Estadística Descriptiva
- Estadística Lineal
- Estadística no Lineal
- Estadística Biestadística
- Pruebas Estadísticas Clásicas
- Análisis de Serie de Temporales
- Modelos de Regresión
- Clasificación

- Fiabilidad
- Categorías
- Clustering
- Validación de Datos
- Tendencias
- Gráficos y Diagramas

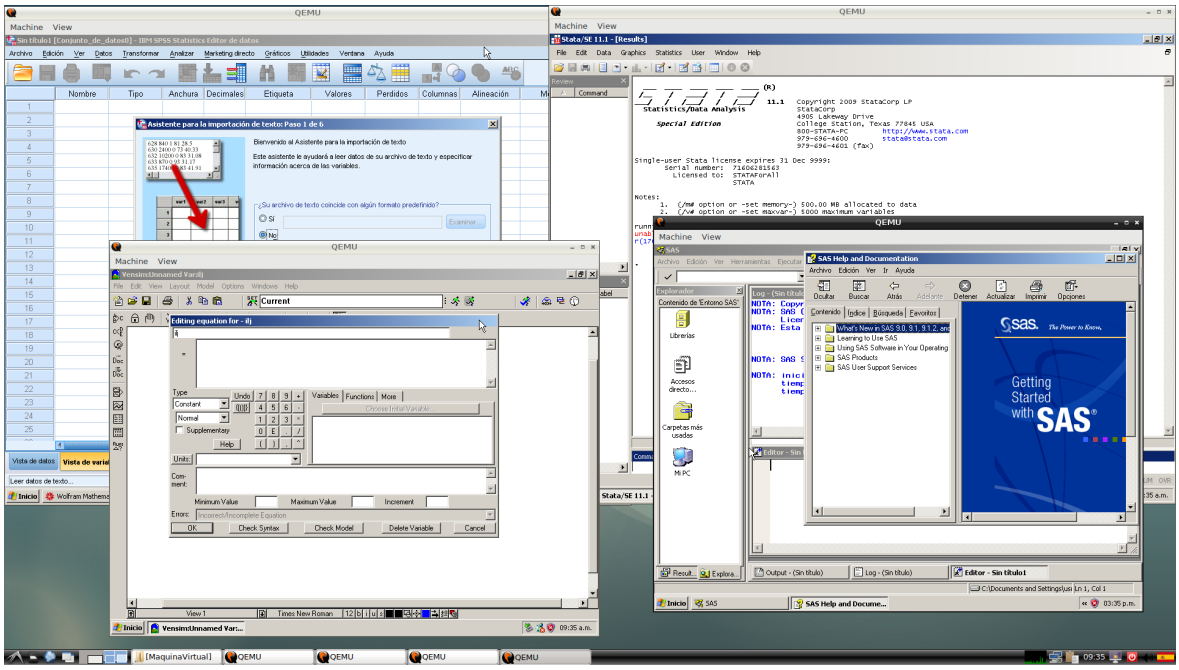

Figura 2.4 Distintos paquetes Estadísticos (IBM SPS, STATA/SE, Vensim y SAS) corriendo dentro de múltiples máquinas virtuales simultáneamente bajo un anfitrión en Linux.

Actualmente, los paquetes estadísticos usados en la carrera de Actuaría en la Facultad de Ciencias son:

- SPSS (véase [53])
- R (véase [29], [39])
- SAS (véase [54])
- PSPP (véase [29], [55])
- EViews (véase [56])
- Gretel (véase [57])
- Stata (véase [58])

- Statgraphics (véase [59])
- Statistica (véase [60])
- Systat (véase [61])
- Vensim (véase [62])
- Maple (véase [40])
- Mathematica (véase [41])
- MatLab (véase [21],[22])
- FreeMat (véase [32])
- Octave (véase [29],[30])
- Maxima (véase [29],[42])
- Scipy (véase [36])

Nos centraremos en los tres primeros paquetes, el resto de los paquetes son muy usados ya sea por sus características, facilidad de uso o la accesibilidad del paquete para los estudiantes. Ya que los paquetes de SPSS y SAS en cualquiera de sus versiones, es difícil conseguir versiones académicas de costo reducido para instituciones educativas.

SPSS

El paquete SPSS (véase [53]), es un programa estadístico informático muy usado en las ciencias sociales y las empresas de investigación de mercado. Originalmente SPSS fue creado como el acrónimo de Statistical Package for the Social Sciences aunque también se ha referido como "Statistical Product and Service Solutions". Sin embargo, en la actualidad la parte SPSS del nombre completo del Software (IBM SPSS) no es acrónimo de nada.

Es uno de los programas estadísticos más conocidos teniendo en cuenta su capacidad para trabajar con grandes bases de datos y una sencilla interfaz para la mayoría de los análisis. En las últimas versiones de SPSS se podían realizar análisis con millones de registros y miles de variables. El programa consiste en un módulo base y módulos anexos que se han ido actualizando constantemente con nuevos procedimientos estadísticos. Cada uno de estos módulos se compra por separado. SPSS soporta hacer interfaz con lenguajes de programación como

Python, R, C, C++, Visual Basic, .NET

Existe una versión incipiente que pretende ser un reemplazo libre para SPSS, se llama PSPP con una interfaz llamada PSPPire (véase [63]), esta es aplicación de Software libre para el análisis de datos multiplataforma, se presenta en modo gráfico y está escrita en el lenguaje de programación C. Usa la biblioteca científica GNU para sus rutinas matemáticas, y plotutils para la generación de gráficos. PSPP puede importar formatos de: Gnumeric, OpenDocument, hojas de Excel, bases de datos Postgres, valores separados por coma y archivos ASCII. Puede exportar archivos en formato SPSS y archivos ASCII. Algunas de las bibliotecas usadas por PSPP pueden ser accedidas vía programación.

R

El paquete R (véase [29], [39]), es un lenguaje y entorno de programación para análisis estadístico y gráfico. Se trata de un proyecto de Software libre, resultado de la implementación GNU del premiado lenguaje S. SPSS, R y S-Plus -versión comercial de S- son, probablemente, los tres lenguajes más utilizados en investigación por la comunidad estadística, siendo además muy populares en el campo de la investigación biomédica, la bioinformática y las matemáticas financieras. A esto se contribuye la posibilidad de cargar diferentes bibliotecas o paquetes con finalidades específicas de cálculo o gráfico.

Además, R puede integrarse con distintas bases de datos y existen bibliotecas que facilitan su utilización desde lenguajes de programación interpretados como Perl y Python. R soporta hacer interfaz con lenguajes de programación como C, C++ y Fortran.

Otra de las características de R es su capacidad gráfica, que permite generar gráficos con alta calidad. R posee su propio formato para la documentación basado en LaTeX (véase [29],[64],[65]). R también puede usarse como herramienta de cálculo numérico, campo en el que puede ser tan eficaz como otras herramientas específicas tales como FreeMat, GNU Octave y su equivalente comercial, MatLab. Se ha desarrollado una interfaz, RWeka para interactuar con

Weka (véase [66]) que permite leer y escribir ficheros en el formato arff y enriquecer R con los algoritmos de minería de datos de dicha plataforma.

Los ambientes de desarrollo integrado para R existen como proyectos externos, como pueden ser editores -que sólo soportan la sintaxis-, los IDEs (Integrate Development Envairoments) y los GUI (Graphical User Interfaces) -permiten editar, correr y depurar código desarrollado para R-. Hay más de 20 proyectos activos, dos de los más conocidos son Tinn-R (véase [67]) y RStudio (véase [68]).

SAS

SAS Institute (véase [54]), es uno de los principales fabricantes de business intelligence Software. Su nombre es el acrónimo de statistical analysis systems («sistemas de análisis estadístico») aunque, posteriormente, al extender su oferta de productos más allá de los meramente dedicados al análisis estadístico, pasó a utilizarlo como nombre propio. El primer producto de SAS Institute fue el SAS Software package, un lenguaje de programación para el análisis estadístico de datos en mainframes de IBM.

Actualmente, este lenguaje de programación, llamado SAS base, es el motor de una serie de herramientas para la dirección estratégica de empresas, la gestión del riesgo financiero, el desarrollo de modelos de minería de datos, etc.

El lenguaje SAS opera principalmente sobre tablas de datos: puede leerlas, transformarlas, combinarlas, resumirlas, crear informes a partir de ellas, etc. El núcleo del lenguaje (conocido habitualmente como SAS Base) incluye:

- Pasos data que permiten realizar operaciones sobre las filas de un conjunto de datos.
- Procedimientos de manipulación de datos que permiten ordenar tablas, enlazarlas, etc.
- Un intérprete de SQL.
- Un súper lenguaje de macros.

SAS Institute comercializa paquetes de procedimientos adicionales para el

análisis estadístico de los datos, tales como:

- SAS/IML, módulo que implementa un lenguaje alternativo similar a Octave, MatLab o R.
- SAS/STAT, un módulo con procedimientos para realizar determinados análisis estadísticos (regresiones, etc.)
- SAS/ETS para el análisis estadístico de series temporales
- SAS/OR para la resolución de problemas de investigación operativa
- SAS/GRAPH para generar gráficos

Además, SAS Institute ha desarrollado aplicaciones, denominadas interfaces, tales como SAS Enterprise Guide, SAS Data Integration Studio, SAS Enterprise Miner y otras que generan código SAS para aplicaciones específicas: ETL, minería de datos, etc.

Ventajas, Desventajas y Carencias

Las organizaciones actuales, manejan una gran cantidad de información, la cual puede o no estar dispersa en sus múltiples sistemas operacionales. Además, en un mercado tan competitivo como el actual, las organizaciones focalizan sus recursos en las estrategias más adecuadas para conducir a la compañía hacia el éxito. Los paquetes estadísticos pueden ayudar a conseguir este objetivo, completando la inversión ya realizada en sistemas operacionales y el hecho de usar paquetes estadísticos que tengan integrado el manejo de las grandes bases de datos ofrece beneficios adicionales.

De los paquetes mencionados, SAS ofrece soluciones en forma de una suite completa para la de gestión de datos y Software analítico para encontrar el llamado poder del conocimiento, pero el costo de las versiones completas es prohibitivo para la gran mayoría de las instituciones educativas, en particular para la UNAM. En general, los paquetes estadísticos proveen un ambiente integrado de análisis de datos, de la gran cantidad de Software existente, la determinación de cual usar en un caso particular, depende de la cantidad de datos y la forma de acceder a ellos.

Por ello, el resto de los paquetes estadísticos propietarios y libres ofrecen una ventaja competitiva, al permitirle al profesor y sus estudiantes contar con versiones completas y funcionales en las que pueden ser aplicados los conocimientos adquiridos en los diversos cursos de la carrera de Actuaría, dejando el manejo especializado de paquetes como SAS a cursos avanzados o para cuando el educando realice sus prácticas profesionales. De esta forma se pueden preparar a los estudiantes para aplicar sus conocimientos al egresar en diversas áreas de la carrera de Actuaría y con pocos conocimientos técnicos adicionales puedan laborar en pequeñas, medianas y grandes empresas.

Paquetes Ofimáticos

En la actualidad, los llamados paquetes ofimáticos, no son otra cosa que programas de cómputo integrado, que permiten automatizar múltiples tareas que permiten idear, crear, manipular, transmitir, almacenar información necesaria en una oficina.

Entre sus prestaciones básicas destacan:

- Procesamiento de Textos
- Hojas de Cálculo
- Bases de Datos
- Herramientas de Presentación y Multimedia

Existe una gran cantidad de paquetes ofimáticos, que van, desde los instalados en el equipo hasta los asequible a través de la Web, entre los más comunes son:

- Microsoft Office (véase [69])
- Libre Office (véase [70])
- OpenOffice (véase [71], [72])
- Calligra (véase [73])
- Google Docs (véase [74])
- Lotus Symphony (véase [75])

Procesamiento de Textos

Existe una gran cantidad de usos para los programas de edición de texto, pero en las carreras de la Facultad de Ciencias y en particular para la carrera de Actuaría, la edición de textos con tipografía científica es común. Es por ello que la gran mayoría de los procesadores de textos no proporcionan las herramientas necesarias para incluir en el texto fórmulas y/o notación matemática. En caso de proveer dichas herramientas, muchas de ellas son de tedioso uso, pues están diseñadas para uso ocasional.

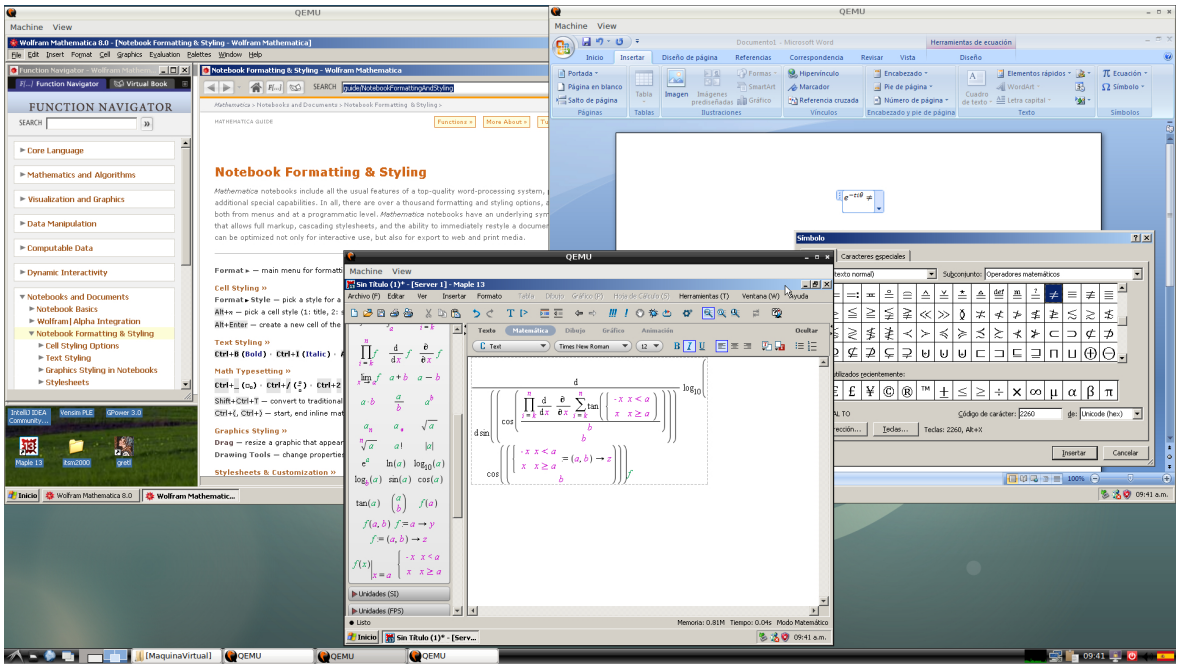

Figura 2.5 Distintos paquetes Ofimáticos (Mathematica, MS Office y Maple 13) corriendo dentro de múltiples máquinas virtuales simultáneamente bajo un anfitrión en Linux.

Para subsanar este hecho, existen herramientas y editores hechos específicamente para permitir la edición de textos científicos en los cuales numerar ecuaciones, usar tipografía matemática, manipular bibliografía y referencias cruzadas es una tarea sencilla de realizar.

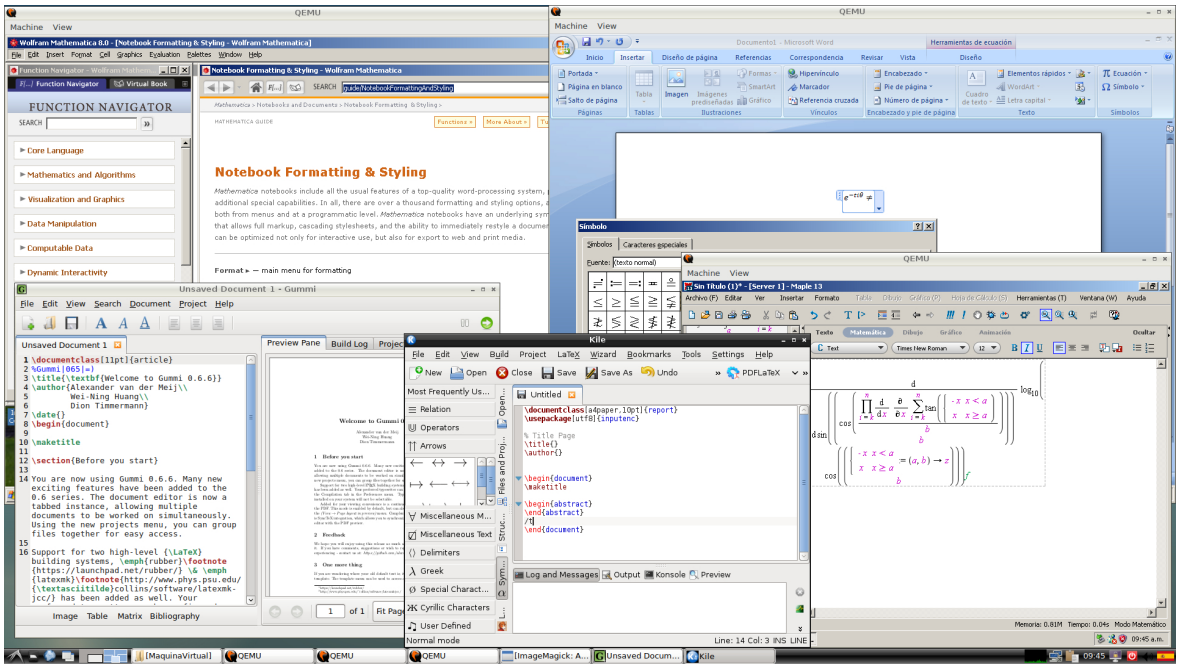

Figura 2.6 Distintos paquetes Ofimáticos (Mathematica, MS Office y Maple 13) corriendo dentro de múltiples máquinas virtuales simultáneamente bajo un anfitrión en Linux y Gummi y Kile nativos en Linux.

Existe una gran variedad de paquetes para la edición de textos científicos -los cuales existen tanto en las plataformas de Windows, Linux, Mac-, entre los que destacan:

- Editor de ecuaciones integrado en Word en Microsoft Office (véase [69])
- MathType para Word en Microsoft Office para Windows (véase [76])
- Scientific WorkPlace LaTex para Windows (véase [77])
- Gummi LaTeX (véase [78])
- Kile LaTeX (véase [79])
- LED LaTeX (véase [80])
- LyX LaTeX (véase [81])
- Texmaker LaTeX (véase [29],[82])
- TeXnicCenter LaTeX (véase [83])
- TextPad LaTeX (véase [84])
- TeXstudio LaTeX (véase [85])
- WinEdt LaTeX (véase [86])
- Formula de Libre Office (véase [70])
- Math de OpenOffice (véase [71], [72])

- Formula de Calligra (véase [73])

Salvo para los productos de Microsoft Office, el resto de los paquetes tienen una curva de aprendizaje de media a alta, pero en contraste permiten desarrollar textos y gráficos con tipografía científica de alta calidad. En la Facultad de Ciencias, desde hace ya varios años, semestre a semestre se imparten cursos a estudiantes, tesistas y profesores de LaTeX y el manejo de uno o mas editores que lo soportan; una cantidad importante de ellos se han impartido en las Aulas y Talleres del Departamento de Matemáticas en el Tlahuizcalpan, además de contar con el repositorio oficial de LaTeX (véase [87]) dentro de la Facultad.

Hojas de Cálculo

Las Hojas de cálculo, es un Software a través del cual se pueden usar datos numéricos y realizar cálculos automáticos de números que están en una tabla. También es posible automatizar cálculos complejos al utilizar una gran cantidad de parámetros y al crear tablas llamadas hojas de Trabajo.

Las hojas de cálculo permiten a los usuarios elaborar tablas y formatos que incluyan cálculos matemáticos mediante fórmulas, las cuales pueden usar operadores matemáticos como son: + (suma), - (resta), * (multiplicación). / (división), y ^(exponenciación); además de poder utilizar elementos denominados funciones como por ejemplo: Suma(), Promedio(), BuscarV(), etc.

Así mismo las hojas de cálculo son útiles para gestionar "Listas" o "Bases de Datos"; es decir Ordenar y Filtrar la información. Por lo tanto, la hoja de cálculo es una herramienta multiuso que sirve tanto para actividades de oficina, que implican la organización de grandes cantidades de datos, como para niveles estratégicos y de toma de decisiones al crear representaciones gráficas de la información sintetizada.

Existen una gran variedad de paquetes para el manejo hojas de cálculo -los cuales existen tanto en las plataformas de Windows, Linux, Mac-, entre los que destacan:

- Excel: Paquete de Microsoft Office (véase [69])
- Calc: Paquete Libre Office (véase [70])
- OpenCalc: Paquete OpenOffice (véase [71][72])
- Spread Sheet: Google Docs (véase [74])
- Sheets: Paquete Calligra (véase [73])

Por otro lado, Microsoft VBA (Visual Basic for Applications) es el lenguaje de macros de Microsoft Visual Basic que se utiliza para programar aplicaciones Windows y que se incluye en varias aplicaciones Microsoft. VBA permite a usuarios y programadores ampliar la funcionalidad de programas de la suite Microsoft Office. Visual Basic para Aplicaciones es un subconjunto casi completo de Visual Basic 5.0 y 6.0.

Microsoft VBA viene integrado en aplicaciones de Microsoft Office, como Word, Excel, Access y PowerPoint. Prácticamente cualquier cosa que se pueda programar en Visual Basic 5.0 o 6.0 se puede hacer también dentro de un documento de Office, con la sola limitación que el producto final no se puede compilar separadamente del documento, hoja o base de datos en que fue creado; es decir, se convierte en una macro (o más bien súper macro). Esta macro puede instalarse o distribuirse con sólo copiar el documento, presentación o base de datos.

Su utilidad principal es automatizar tareas cotidianas, así como crear aplicaciones y servicios de bases de datos para el escritorio. Permite acceder a las funcionalidades de un lenguaje orientado a eventos con acceso a la API de Windows.

Al provenir de un lenguaje basado en Basic tiene similitudes con lenguajes incluidos en otros productos de ofimática como Libre Office y Openoffice, pero no hay compatibilidad entre productos.

Así también, es común que en los cursos de la carrera de Actuaría, se requiera hacer el análisis estadísticos de datos, esto se realizan mediante el uso de

complementos de Excel -Herramientas para análisis y Solver-. En donde al usar estas herramientas, se proporciona los datos y parámetros para cada análisis y la herramienta utilizará las funciones de macros estadísticas o técnicas correspondientes para realizar los cálculos y mostrar los resultados en una tabla de resultados. Algunas herramientas generan gráficos además de tablas de resultados.

Entre las herramientas para análisis se incluyen:

- Análisis de Fourier
- Correlación
- Covarianza
- Estadística descriptiva
- Generación de números aleatorios
- Histograma
- Jerarquía y percentil
- Media móvil
- Muestreo
- Prueba t
- Prueba t para varianzas de dos muestras
- Prueba z
- Regresión
- Suavización exponencial
- Varianza

El acceso a tablas de Excel usando Visual Basic for Applications, además del uso de los complementos de Excel -Herramientas para análisis y Solver- para manejo de datos, ha generalizado el uso de los paquetes de Microsoft Office, esto redunda en el uso de dicha suite en una importante cantidad de cursos dentro de la carrera de Actuaría de la Facultad de Ciencias.

Bases de Datos

Una parte importante de la Modelación Matemática es trabajar con datos de

prueba, lo más cercano posible a la realidad. Ello implica que, es necesario contar con mecanismos para almacenar, editar y consultar una cantidad grande de datos, esto se logra usando las bases de datos.

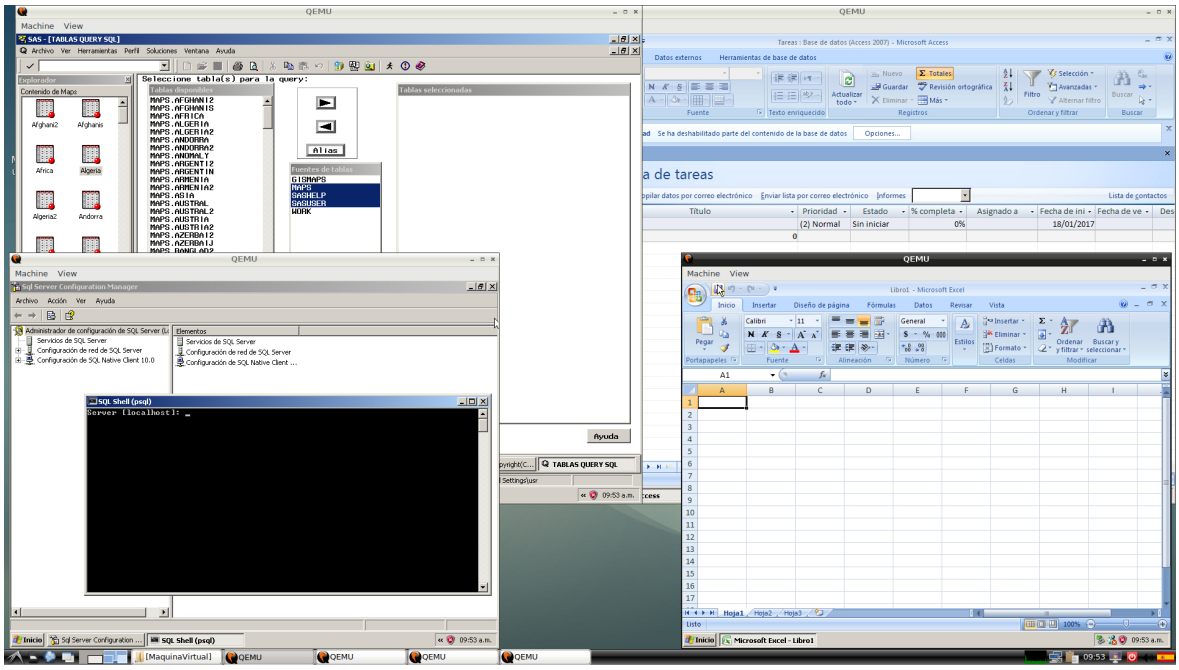

Figura 2.7 Distintos paquetes Ofimáticos (SAS, MS Access, MS Excel y SQL server) corriendo dentro de múltiples máquinas virtuales simultáneamente bajo un anfitrión en Linux.

Existen una gran variedad de paquetes para el manejo de base de datos -los cuales corren en las plataformas de Windows, Linux, Mac-, entre los que destacan:

- Access en Microsoft Office para Windows (véase [69])
- Microsoft SQL Server (véase [88],[89])
- PostgreSQL (véase [90])
- MySQL (véase [91])
- MongoBD (véase [92])

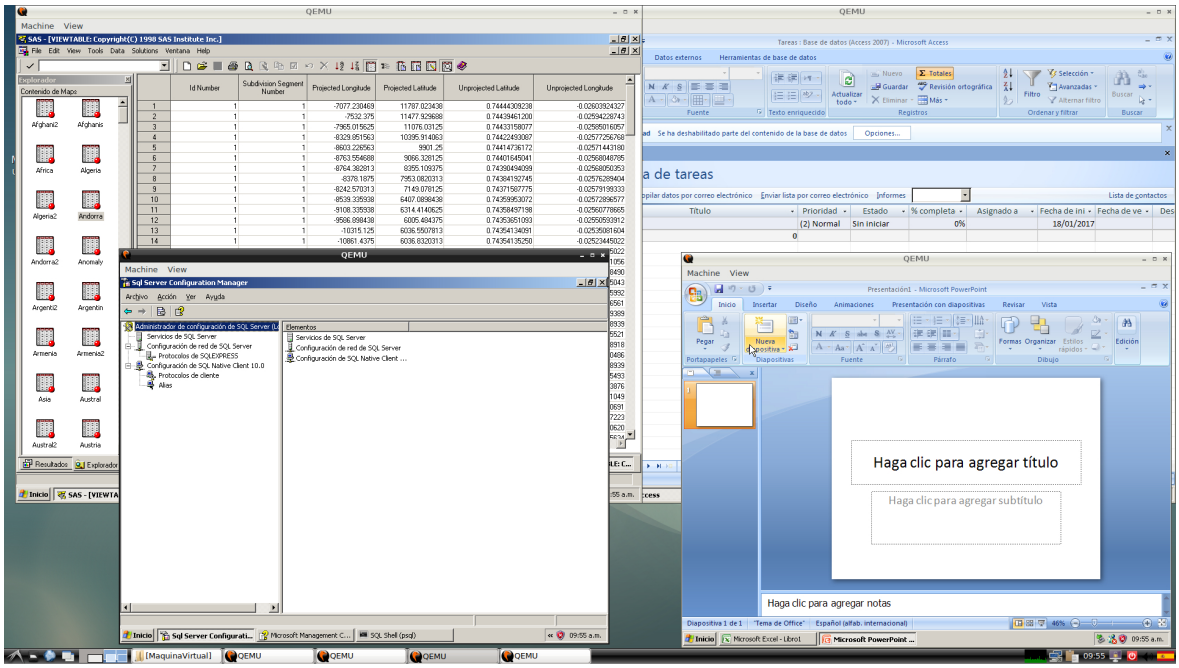

Figura 2.8 Distintos paquetes Ofimáticos (SAS, MS Access, MS Excel y SQL server) corriendo dentro de múltiples máquinas virtuales simultáneamente bajo un anfitrión en Linux.

En donde, entendemos a un Sistema de Gestión de Bases de Datos (SGBD) como un conjunto de programas que permiten el almacenamiento, modificación y extracción de la información en una base de datos, además de proporcionar herramientas para añadir, borrar modificar y analizar los datos. Los usuarios pueden acceder a la información usando herramientas específicas de interrogación y de generación de informes, o bien mediante aplicaciones al efecto.

Los SGBD también proporcionan métodos para mantener la integridad de los datos, para administrar el acceso de usuarios a los datos y recuperar la información si el sistema se corrompe. Permite presentar la información de la base de datos en variados formatos. La mayoría de los SGBD incluyen un generador de informes. También puede incluir un módulo gráfico que permita presentar la información con gráficos y diagramas.

Existen muchos tipos de SGBD distintos según manejen los datos y muchos tamaños distintos según funcionen sobre ordenadores personales y con poca memoria a grandes sistemas que funcionan en mainframes con sistemas de

almacenamiento especiales.

Generalmente se accede a los datos mediante lenguajes de interrogación, lenguajes de alto nivel que simplifican la tarea de construir las aplicaciones. También simplifican la interrogación y la presentación de la información. Un SGDB permite controlar el acceso a los datos, asegurar su integridad, gestionar el acceso concurrente a ellos, recuperar los datos tras un fallo del sistema y hacer copias de seguridad.

El uso generalizado de los paquetes de Microsoft Office, en particular el acceso a el paquete Access mediante tablas de Excel usando Visual Basic for Applications, además del uso de los complementos de Excel para manejo de datos, hace que sean propicios para que un importante número de cursos dentro de la carrera de Actuaría de la Facultad de Ciencias hagan uso extensivo de dicha suite.

Herramientas de Presentación y Multimedia

El programa líder del mercado es Microsoft PowerPoint (véase [69]), este es un programa de presentación desarrollado por la empresa Microsoft para sistemas operativos Microsoft Windows y Mac OS, ampliamente usado en distintos campos como la enseñanza, negocios, etc. Pero todos los paquetes ofimáticos tienen una herramienta para realizar presentaciones, en algunos casos son altamente compatibles con la desarrollada por Microsoft.

Estos programas se han diseñado para hacer presentaciones con texto esquematizado, animaciones de texto e imágenes prediseñadas o importadas desde imágenes de la computadora. Se le pueden aplicar distintos diseños de fuente, plantilla y animación. Este tipo de presentaciones suelen ser más prácticas que las de los paquetes de edición de textos.

Las herramientas de Presentación vienen integradas en los paquetes ofimáticos como un elemento más, que puede aprovechar las ventajas que le ofrecen los demás componentes del paquete para obtener un resultado óptimo.

En el caso de PowerPoint, se puede usar como complemento para la edición de

texto científico a MathType (véase [76]) para Microsoft Office, este es un paquete adicional pero es uno de los más usados en cuanto a la tipografía matemática.

Por otro lado, esta ganando terreno en la Facultad de Ciencias el uso de Beamer (véase [93]), el cual es una clase de LaTeX (véase [29],[64],[65]) para la creación de presentaciones. Este funciona con pdflatex, dvips, LyX entre otros. Al estar basado en LaTeX, Beamer es especialmente útil para preparar presentaciones en las que es necesario mostrar gran cantidad de expresiones matemáticas. En los últimos semestres se ha hecho una amplia difusión a los paquetes que usan a LaTeX como base, pues son ampliamente usados por la comunidad científica mundial.

El uso de herramientas de ofimática esta integrado a las carreras de Ciencias desde ya hace mucho tiempo, pero la gran mayoría se realiza con productos de Microsoft Office, lo cual no representa ningún problema técnico, pero si un problema para la institución y estudiantes, ya que las versiones actualmente usadas, no son del todo compatibles entre sí, ello implica que se requiere o tener la última versión del producto o diferentes versiones del mismo para trabajos cotidianos.

Para la Facultad de Ciencias, el contar con las licencias necesarias para que cada máquina a la que los alumnos tienen acceso cuente con una, es en extremo prohibitivo por el costo. Esto mismo sucede en el caso de los estudiantes, pues el costo de las licencias para uso académicos superan los mil pesos en las versiones que tengan todas las características usadas comúnmente en los cursos de la Facultad.

Es por ello que el uso de herramientas de Software libre se visualiza como un reemplazo natural a los paquetes de Microsoft Office, pero la realidad dista de ser tan simple. Ya que, actualmente no es posible obtener las características mínimas en Software libre para que puedan ser un reemplazo real de los paquetes de Microsoft Office. Este hecho ha ocasionado que existe un uso cada vez más generalizado entre profesores y alumnos a usar Software sin la licencia respectiva

(véase Apéndice A).

Otros Programas

Dentro del abanico de otros programas usados en las carreras que imparte el Departamento de Matemáticas de la Facultad de Ciencias de la UNAM, que no tienen cabida en los rubros anteriores, destacan:

- Lenguajes de Programación
- Entornos de desarrollo integrado y editores para programación
- Cálculo Simbólico
- Manipulación de Gráficos
- Navegadores Web, compresores y descompresores de archivos
- Y los propios sistemas operativos

Ventajas, Desventajas y Carencias del Software Libre Versus el Propietario

La ventaja de tener múltiples herramientas para realizar operaciones elementales y avanzadas de paquetes de cálculo numérico, simbólico, estadístico y ofimático es en sí misma una gran ventaja. Para los centros universitarios y usuarios ocasionales, las herramientas de Software libre son una herramienta invaluable (véase Apéndice A), en el caso de empresas que requieren usar opciones avanzadas o generadas por terceros, los paquetes propietarios destacan como herramientas de trabajo óptimas. Para todos los casos, hay que destacar las siguientes características:

- Funcionalidades básicas: Todos los paquetes implementan las funcionalidades básicas, ya que todos los paquetes llevan años desarrollándose.
- Funcionalidades avanzadas: Los paquetes propietarios tienen implementadas cientos de funciones avanzadas que pueden ser muy útiles para usuarios avanzados, pero rara vez son usados por los usuarios noveles o cotidianos.
- Fiabilidad: En los paquetes en desarrollo son comunes las caídas del programa, el Software propietario destaca por ser más fiable que los demás.
- Información: El Software propietario cuenta con una abundante bibliografía

y la propia ayuda del programa, esta es incipiente en el Software libre.

- Facilidad de Manejo: Ninguno de los programas presenta grandes dificultades a la hora de utilizarlo. Pero en menor o mayor medida, los paquetes de Software libre presentan entornos de desarrollo funcional, pero perfectible.
- Costo: El costo de las diversas versiones de Software propietario suele ser prohibitivo para instituciones educativas y usuarios ocasionales, en el caso del Software libre, los paquetes se pueden descargar de la red sin más costo que el acceso a Internet y los medios de instalación cuando son requeridos.

El Software libre es aún joven, en los miles de proyectos actuales se esta trabajando a diario en mejorar la parte computacional de los algoritmos involucrados en el paquete, haciendo y puliendo interfaces gráficas, generando ayuda en línea así como la documentación necesaria para que usuarios noveles y avanzados usen la mayor cantidad de opciones programadas en los paquetes.

Para muestra de este maravilloso avance, tomemos el proyecto del Kernel de Linux y su uso en los sistemas operativos Android, Ubuntu, Linux Debian, etc, que actualmente corren en millones de equipos y contiene miles de aplicaciones y están soportados por una gran cantidad de usuarios y empresas comerciales. Estos han logrado desplazar a muchos de sus competidores por sus múltiples bondades y bajo costo de desarrollo, al reusar miles de aplicaciones ya existentes que usan Software libre y permitir desarrollar otro tanto de aplicaciones bajo una plataforma que corre en los más diversos procesadores.

Así también, en los últimos años, muchos proyectos han pasado de ser simples programas en línea de comandos a complejas aplicaciones multiplataformas, que corren en distintos sistemas operativos como son Windows, Linux y Mac, con ambientes gráficos multimedia que en muchos casos han superado a sus contrapartes comerciales, por ejemplo los navegadores Web tipo FireFox y la suite ofimática tipo Libre Office, entre muchos otros.

Integración del Software Libre y Propietario en las Carreras de Ciencias

El uso de programas de cómputo de Software libre (véase Apéndice A) esta cada día más integrado al uso cotidiano que hacen profesores, ayudantes y estudiantes en la Facultad de Ciencias, pero todavía para el Sistema Operativo Windows, así como paquetes de uso común, no ha sido posible encontrar un adecuado reemplazo, los más comunes son MatLab, Mathematica, Maple, SPSS, SAS y Microsoft Office.

Para la Facultad de Ciencias, el contar con las licencias necesarias de cada diferente Software y en su caso, de diferentes versiones del mismo paquete para que cada máquina a la que los alumnos tienen acceso, es en extremo prohibitivo por el costo. Esto mismo sucede en el caso de los estudiantes, pues el costo de las licencias para uso académico superan los mil pesos en las versiones que tengan todas las características usadas comúnmente en los cursos de la Facultad.

Por lo anterior, el uso de herramientas de Software libre se visualiza como un reemplazo natural a los paquetes comerciales como Microsoft Office, pero la realidad dista de ser tan simple. Ya que, actualmente no es posible obtener las características mínimas en Software libre para que puedan ser un reemplazo real de los paquetes comerciales en cuanto a las opciones avanzadas. Este hecho ha ocasionado, que exista un uso cada vez más generalizado entre profesores, ayudantes y alumnos a usar Software sin la licencia respectiva.

Actitud de los Usuarios

Al inicio, hubo un poco de renuencia a usar Software libre (véase Apéndice A) por parte de algunos profesores, debido a que en las prácticas en clase se busca acercar a la realidad laboral a los alumnos, en ocasiones varias funcionalidades y el manejo del entorno del Software cambiaba de forma drástica entre el Software libre y el Software propietario, algunas veces el Software libre no ofrecía las mismas utilidades que el Software propietario, representando una desventaja para preparar a los alumnos a nivel laboral.

Posteriormente, gracias a las constantes actualizaciones y mejoras al Software libre, cada día es más utilizado no solo para prácticas, si no que ha sido adoptado por algunas empresas, de manera que al salir los alumnos al campo laboral les ayuda a tener aún más conocimientos.

CAPÍTULO 3. Técnica Académica Asociada “A” Tiempo Completo

En 2004 concursé por ocupar una plaza de Técnico Académico Asociado “A” tiempo completo que fue publicada en la Gaceta de la UNAM, de la cual resulté ganadora, a partir del 07 de octubre de ese mismo año, por designación del Consejo Departamental de Matemáticas estoy adscrita en el edificio de Investigación y Docencia Experimental Tlahuizcalpan.

Actividades Realizadas en Esta Plaza

A partir de octubre de 2004 a la fecha, atiendo junto con otros técnicos las 21 aulas y talleres del Departamento de Matemáticas de la Facultad de Ciencias, ubicados en el edificio de Investigación y Docencia Tlahuizcalpan, los cuales tienen un horario de atención continua de 7 a 21 hrs., de Lunes a Viernes, y de 8 a 13 hrs., los sábados de 8 a 13 hrs., se atiende a grupos de las carreras de Actuaría, Ciencias de la Tierra, Matemáticas Aplicadas, Matemáticas, Ciencias de la Computación, además se atienden a los cursos de Matemáticas para Biología.

En el periodo del 7 de octubre de 2004 al 2010 trabajé directamente en la carrera de Ciencias de la Computación del Departamento de Matemáticas, en la cual estuve a cargo de los Talleres de Sistemas Operativos, Ingeniería de Software, Cómputo Visual y Sistemas Simbólicos, así mismo apoyé en las aulas de Ciencias de la Computación I, II, III. En dichos espacios eran manejados los sistemas operativos Linux (véase [94]) y Windows XP (véase [18]), dependiendo de las necesidades de los alumnos y personal docente. De 2010 a la fecha fui reasignada para atender los talleres de las carreras de Matemáticas y Actuaría.

Administración de Equipos

Colaboré en la instalación de Software diverso utilizado para las distintas actividades académicas dentro de aulas y talleres de Ciencias de la Computación, así como en la instalación, configuración y actualización de los equipos que servían como base del sistema operativo Windows XP en aulas y talleres de

Ciencias de la Computación, de acuerdo a los requerimientos de los cursos de dicha carrera, actualicé constantemente las versiones de Software libre utilizado en los talleres.

Realicé y colaboré en la puesta a punto de equipos en Linux para prácticas de clase y servicio social en el taller de Sistemas Operativos, con las distribuciones de Debian (véase [95]), Fedora (véase [96]) y Ubuntu (véase [97]). También colaboré en la instalación de Software utilizado para las distintas actividades dentro de las aulas y talleres de Ciencias de la Computación, para satisfacer los cursos curriculares, y extracurriculares, como son el Programa de Extensión Universitaria y Vinculación (PEUVI) del Departamento de Matemáticas de la Facultad de Ciencias, mismos que regularmente son impartidos en dichas áreas.

De Septiembre de 2010 a la fecha por reasignación del Consejo Departamental de Matemáticas formo parte del apoyo académico en los talleres y aulas de las carreras de Matemáticas, Actuaría, Matemáticas Aplicadas, Ciencias de la Tierra y Matemáticas para Biología, donde continuo colaborando en diversas actividades necesarias para brindar un servicio de calidad para alumnos, profesores y ayudantes del Departamento.

Colaboré y colaboro en la instalación de Software diverso utilizado para las distintas actividades académicas dentro de aulas y talleres del Departamento de Matemáticas, así como en la instalación, configuración y actualización de los equipos cómputo, de acuerdo a los requerimientos de los cursos de las diversas materias que hacen uso de los espacios (véase [98]).

También colaboro en la instalación de Software utilizado para las distintas actividades dentro de las aulas y talleres, para satisfacer los cursos curriculares, y extracurriculares, como son el Programa de Extensión Universitaria y Vinculación (PEUVI) del Departamento de Matemáticas de la Facultad de Ciencias, mismos que regularmente son impartidos en dichas áreas.

Instalación de Software

Cuando inicié mis labores como Técnica Académica de medio tiempo, estábamos a cargo de 31 equipos de cómputo, proporcional a la demanda de uso de los mismos, así como el Software específico utilizado, se utilizaba el sistema operativo Windows 98 y Software compatible con el mismo, los 31 equipos se instalaban de forma individual respetando las restricciones de la licencia del sistema operativo.

Al cambio de plaza de tiempo completo se contaba con 21 aulas y talleres, así como nuevos equipos, para ser atendidos por un equipo de técnicos académicos, del cual formo parte, para satisfacer las necesidades de las distintas clases y cursos que se imparten en el Departamento de Matemáticas se designaron responsables técnicos por área, ya que cada carrera tiene distintos requerimientos. Inicialmente me fue asignada el área de la carrera de Ciencias de la Computación donde los requerimientos de las prácticas eran satisfechos con una instalación de arranque dual (véase [99])[23], compartida entre los sistemas operativos en Windows XP (véase [18]) y la distribución de Linux Red Hat (véase [99]), que posteriormente se cambió a la distribución de Linux Fedora (véase [71]), debido a los requerimientos del Software utilizado en las prácticas de cada materia, en el caso de Windows la instalación comenzó a hacerse mediante la configuración de una máquina con los requerimientos de cada aula o taller llamada máster, misma que era replicada a las demás mediante clonación (véase [100]) en el resto de aulas o talleres, en la parte de Linux colaboré en la instalación de equipos que eran manejados y configurados mediante un servidor propio de la carrera, mismo que maneja sesiones de alumnos grupos y profesores.

Posteriormente fui reasignada a la carrera de Matemáticas, en la cual se atienden también las solicitudes de las carreras de Actuaría, Matemáticas Aplicadas, Ciencias de la Tierra y Matemáticas para Biología, donde continuo actualmente, al

23 Arranque dual o doble arranque (inglés: Dual boot, Dual booting) es la capacidad de una computadora para poder tener e iniciar con más de un sistema operativo funcionando en un mismo disco rígido o equipo. La capacidad de elegir el sistema a arrancar está otorgada por el Gestor de arranque (o Boot loader).

inicio solo se trabajaba en Windows y la instalación se hacía mediante la preparación de un equipo base con Windows XP, el cual tenia el Software requerido al inicio de semestre por cada grupo en el aula o taller asignado, éste equipo base era clonado en el resto de los equipos del aula o taller, pero debido a la necesidad de cubrir los diversos requerimientos de los grupos y la falta de presupuesto o convenios del Software utilizado comenzó a hacerse una configuración de arranque dual, gracias a los continuos cambios en las distribuciones del sistema operativo Linux y la familiaridad de un entorno gráfico muy parecido a Windows, se continuó usando Windows XP (véase [18]) y la distribución Linux Debian (véase [95]), con el Software requerido por los grupos asignados a cada aula o taller.

En Windows se instalaba Software de paga usando en la medida de lo posible la versión estudiantil, por convenio o gratuita, según se tuviera la opción durante el semestre en curso, de la misma forma, en Linux era instalada sobre la distribución de Debian el Software requerido por los grupos y los equivalentes al Software requerido en Windows de los que no es posible obtener una licencia o convenio por su alto costo para que fuesen utilizados como alternativas en las clases, así como el Software de licencias o convenios adquiridos por el Departamento de Matemáticas.

Para agilizar la instalación en aulas y talleres en ambas áreas, se realizaba el mismo tipo de instalación por medio del método de clonación (véase [100]).

Clonación: Se denomina clonación al proceso de copiar el contenido del disco duro o de una o más particiones de una computadora a otro disco o a un archivo “imagen”. Una vez completada la clonación, es posible restaurar después el contenido al disco u otro equipo de cómputo con similares características .

Los tipos de clonación que se pueden hacer son:

- En el mismo disco: dentro de la misma partición del disco se puede guardar una copia del sistema operativo dentro de una imagen para posteriormente replicarla en el mismo equipo o en otro dentro de una partición.

- De equipo a equipo: Por medio de una imagen o partición base los programas de clonado de disco pueden vincular dos computadoras por medio de un cable paralelo ó transferir la información por medio de una red a una partición o el disco duro completo.
- Por red: Mediante un sistema de almacenaje conectado a la red (servidor SAMBA (véase [101], [102]), SSH (véase [29],[102]) etc.) se hace la imagen que se usará para clonar, misma que será guarda para después poder transferirla a una partición o en el disco duro completo del equipo a clonar.
- Dispositivo de almacenaje externo: Con ayuda de un dispositivo de almacenaje externo (CD, DVD, Disco Duro, USB, etc.) se guardan y se cargan las imágenes a una partición o el disco duro completo mediante un cable USB.

Debido a las características de los equipos, la infraestructura de la red y los recursos con que se contaban, la única forma de realizar la clonación era de equipo a equipo por medio de la red, de esta manera se realizaba una instalación medianamente rápida, ya que a mayor número de equipos el proceso de clonación por medio de un servidor en red, la transmisión de las imágenes era más lenta. De esta forma, se garantizaba medianamente la integridad del Software por virus, problemas de configuración, fallas del sistema operativo, etc., para garantizar el óptimo desempeño de los mismos.

Problemática Encontrada en el Desarrollo de las Actividades en la Plaza

El tiempo de solución de problemas a nivel Software que detectábamos o que eran reportados por los usuarios se veía limitado por la cantidad de espacios y equipos, ya que estos se atendían en el aula o taller al que correspondían en los horarios disponibles; si solo se trataba de un equipo, éste se dejaba en el aula o taller correspondiente o se sacaba para realizar la clonación (véase [100]), pero cuando fallaban varios equipos, el tiempo de solución se veía reducido con esta forma de instalación, debido a que el tiempo libre que se tenía para trabajar entre clases no era suficiente para terminar la clonación en el aula o taller, con el riesgo de que éstos fueran apagados durante el transcurso de la clonación por error al

iniciar una clase, por fallas en la transmisión de la imagen por red o fallas en la energía eléctrica, la última opción era sacar equipo por equipo para clonarlo, retrasando así el tiempo de solución.

Aunado a la instalación de nuevas peticiones de Software, así como las restricciones de compatibilidad entre un Software y otro, y de estos con el sistema operativo, la continua desconfiguración de equipos por el uso constante y los equipos infectados por virus, había una gran cantidad de equipos inoperativos, esto fue sobrepasando el tiempo de respuesta a la instalación de nuevas peticiones y la recuperación de equipos dañados, la situación se agravaba, por la falta de tiempo que quedaba libre entre las asignaciones iniciales y las continuas asignaciones de clases, haciendo insuficientes los espacios, los equipos y sus recursos, por lo que se tuvo que optar por una nueva metodología de instalación.

Solución: Se eligió realizar la instalación del sistema operativo base en Linux, mediante la distribución de Linux Debian Estable (véase [95]), así mismo se continuaron haciendo las instalaciones base en Windows XP, mismas de las que se generaron imágenes en máquinas virtuales (véase [103]) basadas en KVM (véase [92])[24], que en lugar de ser replicadas mediante clonación se sincronizan mediante RSYNC[25], de esta manera se aprovecha mejor la capacidad de los equipos sin saturarlos de instalaciones, evitando problemas de incompatibilidad entre versiones del Software requerido o de éstos con el sistema operativo, así como evitar la desconfiguración o problemas por virus, mediante la configuración se define que las máquinas virtuales no almacenen ningún cambio realizado en la sesión, ni el equipo, de esta forma cada inicio de sesión y del uso de la máquina virtual se realiza de forma siempre limpia, tal como se configuraron, asegurando así siempre tener los equipos funcionando con las últimas versiones del Software solicitado y en caso de tener alguna falla, se garantiza una recuperación rápida.

La instalación de las máquinas virtuales y del sistema base se realiza por medio

[24] Máquina virtual basada en el kernel (Kernel-based Virtual Machine o KVM)
[25] Sincronización Remota (Remote Sync RSYNC)

de un servidor SSH[26] y RSYNC en Debian Estable, donde se guardan las actualizaciones de la paquetería del sistema base para actualizar mediante APTITUDE[27], las máquinas virtuales se actualizan mediante RSYNC, las imágenes del disco duro y sus particiones se actualizan mediante CLONEZILLA[28], estas últimas son actualizadas y replicadas sólo cuando se requiere cambiar el Software base, ya que el sistema operativo y las máquinas virtuales se actualizan cuando así es requerido mediante actualizaciones vía red usando RSYNC y SSH cuando el equipo es usado por los usuarios, el tiempo de instalación y mantenimiento ha disminuido, ya que no es necesario retirar los equipos para realizarlas, así mismo para actualizar los equipos solo necesitan estar encendidos y conectados a la red, así por medio de dicha actualización se replican o actualizan las máquinas virtuales existentes o nuevas. Al igual que con el método de clonación la actualización del sistema depende de la continuidad de la energía eléctrica y la red, pero el tiempo de transmisión es menor, aunque se recuperen o actualicen varios equipos a la vez, ya que si el equipo pierde la conexión, al usar RSYNC esta se recupera a partir del último dato recibido y no es necesario rehacer la trasmisión completa.

Para resolver una problemática común, como lo es el almacenamiento y uso de archivos, se puso a disposición de alumnos y profesores un espacio compartido en el cual puedan guardar archivos de distintas características y así poder usarlo sin importar la máquina usada o el aula o taller en que se encuentren, para ello se realizan mini manuales, que explican como usar dichos recursos.

Las bases obtenidas en materias como Sistemas Operativos, Redes, así como las prácticas realizadas durante el servicio social, fueron parte fundamental para la comprensión y apoyo en el uso, adaptación e implementación de nuevas tecnologías para las instalaciones, así como el mejor aprovechamiento del equipo y la infraestructura con que se cuenta.

---

26 Servidor Shell Seguro (Secure SHell SSH)

27 Comando de actualización de paquetes usado en la distribución Linux Debian

28 Software de clonación de discos y particiones

Máquinas Virtuales

Entendamos por una máquina virtual (véase [103]) a un Software que simula a una computadora y puede ejecutar programas como si fuese una computadora real. Una característica esencial de las máquinas virtuales es que los procesos que ejecutan están limitados por los recursos y abstracciones proporcionados por ellas. Estos procesos no pueden escaparse de esta "computadora virtual". Como toda nueva tecnología, la virtualización tiene ventajas y desventajas, las cuales deben de ser sopesadas en cada ámbito de implementación.

Aunque las máquinas virtuales, permiten en un mismo equipo de cómputo correr más de un sistema operativo o distintas versiones del mismo, uno de los inconvenientes, es que agregan gran complejidad al sistema en tiempo de ejecución. Esto tiene como efecto algún tipo de ralentización del sistema, es decir, el programa no alcanzará la misma velocidad de ejecución que si se instalase directamente en el sistema operativo "anfitrión" (host) o directamente sobre la plataforma de Hardware. Sin embargo, a menudo la flexibilidad que ofrecen compensa esta pérdida de eficiencia. Si la virtualización es por Hardware, la velocidad de ejecución es más que aceptable para la mayoría de los casos.

En el caso de instituciones educativas y en particular en las Aulas y Talleres del Departamento de Matemáticas de la Facultad de Ciencias de la UNAM, es común que en un mismo equipo de cómputo sea necesario correr por un lado diferentes versiones de sistemas operativos -por ejemplo Linux, Windows XP, Windows 7, etc.- y por otro lado, en un sistema operativo, correr diferentes versiones de un mismo paquete -generalmente no se pueden tener instalados simultáneamente más de una versión-, una muestra del Software que se instala se detalla a continuación:

Java JRE y JDK	Libre Office	R
CodeBlocks IDE	OpenOffice	Tinn-R
NetBeans IDE	MathType	RStudio
Drjava IDE	Scientific WorkPlace	Gretl
IntelliJ IDE	Microsoft SQL Server	MatLab
BlueJ IDE	PostgreSQL	Scilab
Scite	SPSS	Octave
Jet Brains IDE	PSPP	FreeMat
SharpDevelop IDE	SAS	Maple
Alice	Vensim PLE	Mathematica
DFD	Statgraphics	NetLogo
Turbo C IDE	GPower	GeoGebra
Developer Studio-Fortran IDE	EViews	Compresores y descompresores de archivos Winzip, WinRAR, 7-zip
Microsoft Visual Studio	Systat	SSH Secure File Transfer
Microsoft Windows SDK	Stata	PDFCreator
Compaq Visual Fortran	Statistica	Adobe Reader
Microsoft Office	ITSM200	Navegador de páginas Web Internet Explorer, Google Chrome, Konqueror, Mozilla

Tabla 2 Parte del Software instalado en diversas máquinas virtuales usadas en aulas y talleres de Matemáticas en el edificio Tlahuizcalpan.

En este y en otros casos, las máquinas virtuales son una verdadera opción donde pueden coexistir simultáneamente diferentes versiones de sistemas operativos y dentro de un mismo sistema máquinas virtuales corriendo las diversas versiones de un mismo Software.

Además que son configuradas para que al momento de iniciarlas siempre corran a partir de una configuración e instalación base, de tal forma que al ser lanzadas el usuario pueda instalar, configurar e inclusive dañar la máquina virtual, pero al reiniciarse siempre está en una nueva sesión, todo queda igual a la versión base, de esta forma no hay posibilidad de infección de virus entre diversos lanzamientos de sesiones de la máquina virtual, la actualización es centralizada y se puede hacer por red, sin intervención del usuario.

Por ello, es que se evaluó como opción viable tener en una máquina un sistema huésped como la distribución de Linux Debian Estable y dentro de el, un grupo de máquinas virtuales de MS Windows -Windows XP, Windows 7, etc.-, en los que cada máquina virtual tenga instalado un grupo de Software agrupado por las características del sistema operativo necesario para correr todas las aplicaciones seleccionadas -por ejemplo agrupados por la versión de Service Pack-, tal como se cuenta en varias de las máquinas virtuales configuradas .

A continuación se muestra una ejecución de Linux Debian Testing stretch versión 9 como sistema anfitrión y dentro de el, se corre mediante KVM cuatro versiones de Windows XP en el que se corre un Office 2003, Office 2007, Office 2010 y una amplia variedad de Software comúnmente usado. Adicionalmente en cada una de las máquinas virtuales se muestra el Administrador de tareas en la que se muestra el consumo de memoria y recursos computacionales que es ínfimo ya sólo se instala el Software estrictamente necesario.

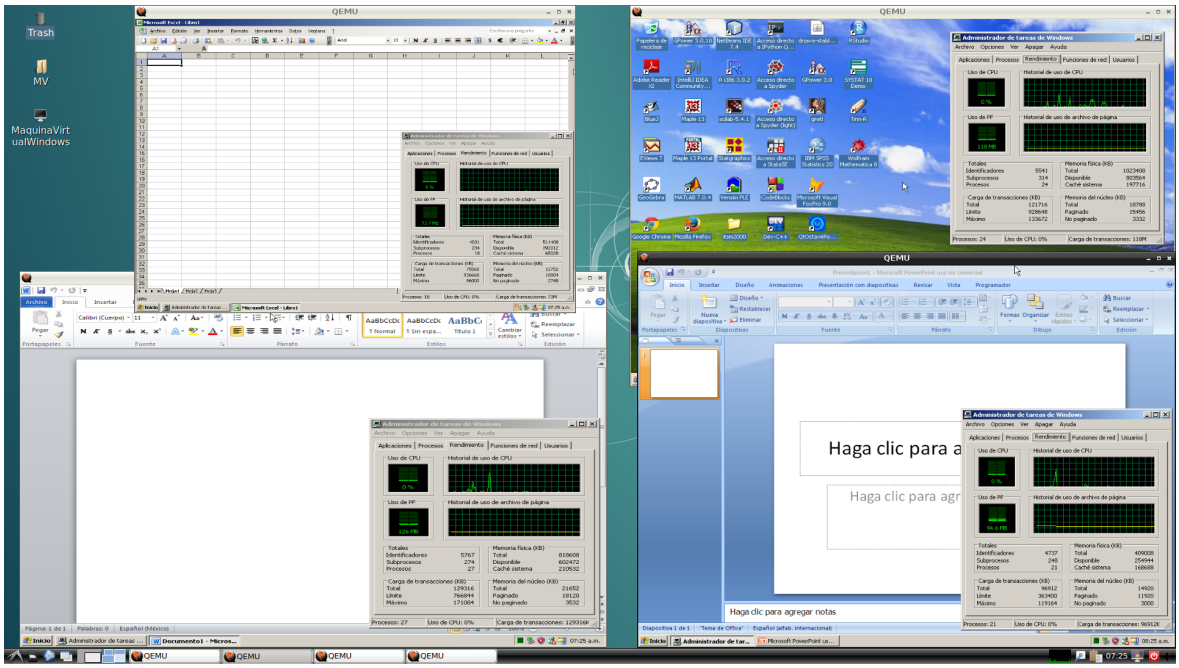

Figura 3.1 Virtualización de múltiples máquinas virtuales simultáneamente

Para entrar en materia de lo que es y no es una máquina virtual, es necesario especificar que entendemos por una máquina virtual, este es un Software (véase [103], [111], [112], [113] y [114]) que simula a una computadora y puede ejecutar programas como si fuese una computadora real. Una característica esencial de las máquinas virtuales es que los procesos que ejecutan están limitados por los recursos y abstracciones proporcionados por ellas. Estos procesos no pueden escaparse de esta "computadora virtual". Uno de los usos domésticos más extendidos de las máquinas virtuales es ejecutar sistemas operativos para "probarlos". De esta forma podemos ejecutar un sistema operativo que queramos probar (GNU/Linux, por ejemplo) desde nuestro sistema operativo habitual (Windows 7 por ejemplo) sin necesidad de instalarlo directamente en nuestra computadora y sin miedo a que se desconfigure el sistema operativo primario.

Tipos de Máquinas Virtuales

Las máquinas virtuales se pueden clasificar en dos grandes categorías según su funcionalidad y su grado de equivalencia a una verdadera máquina:

- Máquinas virtuales de sistema (en inglés System Virtual Machine). También llamadas máquinas virtuales de Hardware, permiten a la máquina física subyacente multiplicarse entre varias máquinas virtuales, cada una

ejecutando su propio sistema operativo. A la capa de Software que permite la virtualización se la llama monitor de máquina virtual o hypervisor. Un monitor de máquina virtual puede ejecutarse o bien directamente sobre el Hardware o bien sobre un sistema operativo ("host operating system").

- Máquinas virtuales de proceso (en inglés Process Virtual Machine). A veces llamada "máquina virtual de aplicación", se ejecuta como un proceso normal dentro de un sistema operativo y soporta un solo proceso. La máquina se inicia automáticamente cuando se lanza el proceso que se desea ejecutar y se detiene para cuando éste finaliza. Su objetivo es el de proporcionar un entorno de ejecución independiente de la plataforma de Hardware y del sistema operativo, que oculte los detalles de la plataforma subyacente y permita que un programa se ejecute siempre de la misma forma sobre cualquier plataforma.

Aplicaciones de las Máquinas Virtuales de Sistema

Varios sistemas operativos distintos pueden coexistir sobre la misma computadora, en sólido aislamiento el uno del otro, por ejemplo para probar un sistema operativo nuevo sin necesidad de instalarlo directamente. La máquina virtual puede proporcionar una arquitectura de instrucciones que sea algo distinta de la verdadera máquina. Es decir, podemos simular Hardware.

Varias máquinas virtuales -cada una con su propio sistema operativo llamado sistema operativo "invitado" o "guest"-, pueden ser utilizadas para consolidar servidores. Esto permite que servicios que normalmente se tengan que ejecutar en computadoras distintas para evitar interferencias, se puedan ejecutar en la misma máquina de manera completamente aislada y compartiendo los recursos de una única computadora. La consolidación de servidores a menudo contribuye a reducir el coste total de las instalaciones necesarias para mantener los servicios, dado que permiten ahorrar en Hardware.

La virtualización es una excelente opción hoy día, ya que las máquinas actuales -Laptops, desktops, servidores- en la mayoría de los casos están siendo "subutilizados" -gran capacidad de disco duro, memoria RAM, etc.-, llegando a un

uso de entre 30% a 60% de su capacidad. Al virtualizar, la necesidad de nuevas máquinas en una ya existente permite un ahorro considerable de los costos asociados -energía, mantenimiento, espacio, etc.-.

Técnicas de Virtualización

Básicamente se reconocen tres tipos de virtualización, algunas de las cuales son usadas actualmente en la gran mayoría de los sistemas operativos, generalmente sin que el usuario este consciente de que usa virtualización, el ejemplo más común y omnipresente es la máquina virtual del lenguaje de programación de JAVA.

Emulación del Hardware Subyacente (ejecución nativa)

Esta técnica se suele llamar virtualización completa -full virtualization- del Hardware, y se puede implementar usando un hypervisor de Tipo 1 o de Tipo 2:

1. Monitor de tipo I, se ejecuta directamente sobre el Hardware.
2. Monitor de tipo II, se ejecuta sobre otro sistema operativo.

Cada máquina virtual puede ejecutar cualquier sistema operativo soportado por el Hardware subyacente. Así los usuarios pueden ejecutar dos o más sistemas operativos distintos simultáneamente en computadoras "privadas" virtuales. Actualmente tanto Intel como AMD han introducido prestaciones a sus procesadores x86 para permitir la virtualización de Hardware.

Emulación de un Sistema no Nativo.

Las máquinas virtuales también pueden actuar como emuladores de Hardware, permitiendo que aplicaciones y sistemas operativos concebidos para otras arquitecturas de procesador se puedan ejecutar sobre un Hardware que en teoría no soportan. Esta técnica permite que cualquier computadora pueda ejecutar Software escrito para la máquina virtual. Sólo la máquina virtual en sí misma debe ser portada a cada una de las plataformas de Hardware.

Virtualización a Nivel de Sistema Operativo

Esta técnica consiste en dividir una computadora en varios compartimentos independientes de manera que en cada compartimento podamos instalar un

servidor. A estos compartimentos se los llama "entornos virtuales". Desde el punto de vista del usuario, el sistema en su conjunto actúa como si realmente existiesen varios servidores ejecutándose en varias máquinas distintas.

Ventajas y Desventajas

Como se menciono anteriormente, al tratarse de una nueva tecnología, la virtualización tiene ventajas y desventajas, las cuales se deben evaluar en el ámbito en que sean implementadas. Debido a que las máquinas virtuales permiten correr en un mismo equipo de cómputo más de un sistema operativo o distintas versiones del mismo, uno de sus inconvenientes es la gran complejidad que agregan al sistema en tiempo de ejecución ralentizando el sistema, debido a que los programas no alcanzan la misma velocidad de ejecución a comparación de que éstos fueran instalados directamente en el sistema operativo "anfitrión" (host) o sobre la plataforma de Hardware. Frecuentemente, la flexibilidad que éstas ofrecen compensa en gran medida la pérdida de eficiencia, de forma que al realizar la virtualización por Hardware, en la mayoría de los casos, la velocidad de ejecución es más que aceptable.

Ventajas

Además de permitir correr múltiples sistemas operativos, diferentes versiones de un mismo sistema pero con diferente Software que en principio puede ser incompatible entre sí, el hecho en si, de no tener problemas con los virus de Windows le confiere una gran ventaja desde el punto de vista administrativo y del usuario final. Además de, permitir una administración centralizada y que todas las máquinas virtuales tendrían la misma configuración y paquetes sin importar el Hardware subyacente en las que se ejecute el sistema operativo huésped.

Al tratarse de instituciones educativas y en particular, en base a la experiencia dentro de las Aulas y Talleres del Departamento de Matemáticas de la Facultad de Ciencias de la UNAM, comúnmente en un mismo equipo de cómputo es necesario correr por un lado diferentes versiones de sistemas operativos -por ejemplo Linux, Windows XP, Windows 7, etc.- y por otro lado, en un sistema

operativo, correr diferentes versiones de un mismo paquete -generalmente no se pueden tener instalados simultáneamente más de una versión sobre el mismo sistema operativo-.

La coexistencia simultánea de diferentes versiones de sistemas operativos y en un mismo sistema, así como máquinas virtuales corriendo las diversas versiones de un mismo Software, ayudan a aprovechar mejor los recursos de los equipos donde son utilizadas las máquinas virtuales.

El que se pueda configurar un equipo para que al momento de iniciarlo siempre esté operativa una configuración e instalación base, da autonomía a el usuario para instalar, configurar e inclusive dañar la máquina virtual. Así como la garantía de que al reiniciarse este en una nueva sesión y que toda la instalación base se encuentre intacta y a disposición del mismo o diversos usuarios, evitando al mismo tiempo de infección de virus entre diversos lanzamientos de sesiones de la máquina virtual. Al permitir una actualización centralizada y por red independiente del usuario facilita el mantenimiento.

El hecho de que el usuario puede instalar y probar programas de cómputo sin dañar la configuración existente, es en si una gran ventaja. Por lo tanto, es una opción viable y común tener en una máquina un sistema huésped como Linux Debian Estable y dentro de el, un grupo de máquinas virtuales de Windows -Windows XP, Windows 7, etc.-, en los que cada máquina virtual tenga instalado un grupo de Software agrupados por las características del sistema operativo necesario para correr a todas las aplicaciones seleccionadas -por ejemplo agrupados por la versión de Service Pack-.

Desventajas

Entre las principales desventajas de virtualizar sistemas propietarios como Windows -no así los sistemas libres como GNU Linux Debian (véase Apéndice A)- es que se puede violar el sistema de licenciamiento (véase Apéndice [A.2] y [A.3]) del Software instalado en las máquinas virtuales, esto es especialmente importante cuando se usa en más de una máquina, pues la licencia usada para la

instalación es violada cuando se tiene más de una copia de la máquina virtual o se ejecutan múltiples instancias de la máquina virtual.

En el caso de Windows XP Home, no se infringe la licencia mientras se cuente con número de licencias igual al máximo número de máquinas virtuales lanzadas simultáneamente. Para otras versiones del sistema operativo Windows como es Windows XP Profesional, la virtualización se maneja con licencias adicionales a la del sistema operativo original y se debe de contar con tantas licencias como el máximo número de máquinas virtuales lanzadas simultáneamente.

Además, es necesario contar con el tipo de licencia adecuada para virtualizar a todos y cada uno de los paquetes de cómputo instalados en cada máquina virtual y en las instancias para el número de máquinas virtuales lanzadas simultáneamente en uno o más equipos.

Uso de Máquinas Virtuales

En esta sección mostraremos como trabajar con las máquinas virtuales para probar imágenes ISO descargadas de la red, instalar y usar máquinas virtuales para Windows y Linux entre otras cosas, primero es necesario saber si nuestro equipo soporta la virtualización por Hardware o Software, para ello supondremos que tenemos acceso a una máquina en la cual inicializamos usando una versión "Live" de CDROM, DVD o USB para arrancar la computadora.

Revisión del tipo de virtualización soportado por la máquina. A continuación hay que revisar si hay soporte en Hardware para la virtualización, usando para ello:

$ egrep "vmx|svm" /proc/cpuinfo

si se soporta la virtualización por Hardware aparecerá la bandera:

Procesadores INTEL: *vmx*

Procesadores AMD: *svm*

Instalar y Usar Máquinas Virtuales

Por omisión los equipos de tecnología de bajo desempeño no soportan la virtualización a nivel de Hardware, por ello es común el uso de QEMU. Si la

computadora soporta virtualización a nivel de Hardware es posible usar KVM, estos tienen la misma sintaxis de uso, y sólo es necesario remplazar qemu por kvm y siempre se usara qemu-img para ambos paquetes.

Instalación de KVM (recomendado para virtualización por Hardware):

 # aptitude install kvm

Instalación de QEMU:

 # aptitude install qemu

Nota: El desempeño de QEMU (sin uso de asistencia de virtualización por Hardware) versus KVM es de varios ordenes de magnitud menor, pero una imagen creada con cualquiera de ellos es usable con el otro virtualizador (en el caso de Windows sólo hay que usar un Hardware parecido al de qemu, en caso contrario marca que es necesario registrar el sistema operativo, para ello se usa la bandera *-cpu*).

Por ejemplo:

Usar máquina virtual de Windows de QEMU en KVM:

$ kvm -localtime -m 400 -boot c -hda Windows.img -cpu qemu32

Suponiendo que se instalo QEMU, entonces podemos usar la máquina virtual con un archivo ISO, en este ejemplo no supondremos que tenemos disco duro, sólo cdrom:

$ qemu -cdrom knoppix.iso -m 700

Para un ejemplo completo de instalación y uso de una máquina virtual para Windows XP, necesitamos:

Crear el disco virtual, por ejemplo de 10 GB, mediante

$ qemu-img create -f qcow2 WindowsXP.img 10G

Luego, hacer la instalación básica de Windows XP a partir, por ejemplo del ISO, *es_winxp_pro_with_sp2.iso*

$ qemu -no-reboot -cdrom es_winxp_pro_with_sp2.iso -boot d -hda WindowsXP.img -m 400 -localtime

Concluir la instalación de Windows XP

$ qemu -no-reboot -boot c -hda WindowsXP.img -cdrom es_winxp pro_with_sp2.iso -m 400 -localtime

Después de la instalación, es conveniente compactar y desfragmentar la imagen usando

$ qemu-img convert -c WindowsXP.img -O qcow2 Windows.img

Y por último, el uso de máquina virtual de Windows XP es mediante

$ qemu -boot c -hda Windows.img -m 400 -localtime

A continuación se muestra una ejecución de Linux Debian Testing stretch versión 9 como sistema anfitrión y dentro de el, se corre mediante KVM cuatro versiones de Windows XP en el que se corre un Office 2003, Office 2007, Office 2010 y en otra se muestra que se ha instalado una amplia variedad de Software comúnmente usado en las carreras de la Facultad de Ciencias.

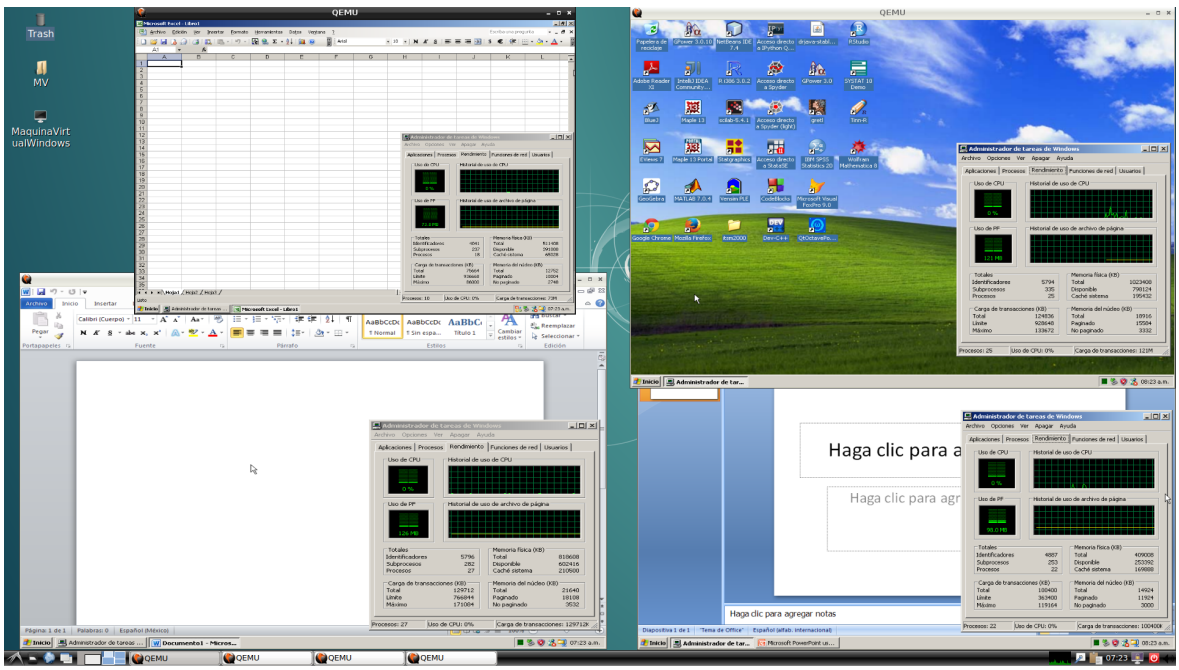

Figura 3.2 Múltiples máquinas virtuales con Windows XP corriendo simultáneamente en un anfitrión Linux

Nótese que cada máquina virtual de Windows XP tiene abierto el Administrador de tareas indicando el uso de CPU que se encuentra cada una de ellas, en donde se aprecia el bajo consumo de memoria y recursos computacionales, ya que sólo es necesario instalar Software que es estrictamente necesario usar sin sobrecargar con antivirus, pues la máquina se destruye sin guardar cambios al momento de cerrarla, ya que así nos conviene y disminuye la propagación de virus y otros Software malicioso.

Para un ejemplo completo de instalación y uso de una máquina virtual para Windows 7, necesitamos:

Crear el disco virtual, por ejemplo de 15 GB:

$ qemu-img create -f qcow2 Windows7.img 15G

Hacer la instalación básica de Windows 7 a partir, por ejemplo del DVD:

$ kvm -no-reboot -cdrom /dev/cdrom -boot d -hda Windows7.img -m 500 -localtime

Concluir la instalación de Windows 7:

$ kvm -no-reboot -boot c -hda Windows7.img -cdrom /dev/cdrom -m 500 -localtime

Después de la instalación, es conveniente compactar y desfragmentar la imagen usando:

$ qemu-img convert -c Windows7.img -O qcow2 Windows.img

Uso de máquina virtual de Windows 7:

$ kvm -boot c -hda Windows.img -m 500 -localtime

Una vez que se cuenta con una imagen de Windows, podemos instalar por ejemplo Office.

Instalar Windows Office 2003, aquí suponemos que tenemos un ISO de Office:

$ qemu -localtime -m 300 -boot c -hda Windows.img -cdrom Office-2003.iso

Si se tiene el cdrom, entonces podemos usar:

$ qemu -localtime -m 300 -boot c -hda Windows.img -cdrom /dev/cdrom/

Otro ejemplo, lo tenemos en la instalación de Debian estable a partir del archivo ISO bajado de la red.

Generar un disco virtual, por ejemplo de 10 GB:

$ qemu-img create -f qcow2 debianStable.img 10G

Instalar la imagen de Debian estable en un disco virtual:

$ qemu -no-reboot -boot d -cdrom debian-503-i386-netinst.iso -hda debianStable.img -m 300

Usar Debian estable:

$ qemu -hda debianStable.img -m 400

Detener y Reiniciar la Máquina Virtual en un Estado Dado

También podemos manipular la máquina virtual al usar la combinación de teclas:

[Ctrl] + [Alt] + [2]

ya en ella, por ejemplo podemos detener y grabar el estado de la máquina virtual:

savevm test.vm

quit

Para que en otro momento, podamos restaurar la máquina virtual tal como estaba cuando esta se detuvo:

$ qemu -boot c -hda .img -m 400 -localtime -loadvm test.vm

Optimización de Imágenes

Las imágenes de disco de QEMU después de generarlas (al instalar algún sistema operativo) son archivos dispersos, para optimizar su rendimiento es recomendable convertir la imagen dispersa en una que no tenga esta propiedad, usar:

$ qemu-img convert disk-sparse.img -O qcow2 disk.img

o puede ser compactada, usar:

$ qemu-img convert -c disk-sparse.img -O qcow2 disk.img

la cual decrecerá el tamaño, para descompactar una imagen e incrementar la velocidad de uso, usar:

$ qemu-img convert disk-compact.img -O qcow2 disk.img

Trabajar con una Imagen Virtual sin que se Altere

Es muy deseable al trabajar con una máquina virtual, el dejar la información de la máquina virtual base intacta y guardar los cambios que se requieran en otro archivo, una forma es hacer una copia y trabajar con la copia de esta o crear un archivo que almacene por separado los cambios a la imagen, para esto último usar:

$ qemu-img create -b debianStable.img -f qcow2 debian.img

y trabajar con la imagen que resultante (para este ejemplo debian.img)

Comunicación Entre la Máquina Virtual y el Sistema Anfitrión

Que hacer para tener comunicación entre la máquina virtual y el sistema anfitrión, hay varias maneras de hacer esto:

1) Lo mas sencillo es que la máquina virtual se conecte a un servidor en red del tipo samba, este puede ser una máquina Windows que comparta una impresora y/o disco o la máquina anfitrión tenga instalado SAMBA y comparta uno o mas servicios como son discos o impresoras.
2) Conectarse a un servidor de SSH mediante los programas SSH Server, así la máquina virtual puede acceder mediante SFTP a los archivos en el servidor. Es posible instalar el servidor de SSH en la máquina anfitrión y así poder prestar el servicio de SSH y FTP a la máquina virtual.
3) Leer un dispositivo USB montado en el sistema anfitrión desde la máquina virtual, para ello el dispositivo USB deberá estar conectado en la máquina anfitrión y deberá ser accesado directamente en la máquina virtual. QEMU necesita parámetros adicionales, el parámetro *-usb* activa el soporte en la máquina virtual de dispositivos USB. La emulación de Intel SB82371 UHCI-Controller tiene 8-puertos en el USB hub. Si se busca tener acceso a uno de los dispositivos físicos, se requiere encontrar los parámetros Vendor-ID and Product-ID. Esta información se obtiene examinando la salida del comando:

   # /sbin/lsusb

   o

   $ cat /proc/bus/usb/devices

Entonces es posible decirle a QEMU los datos de VendorID y ProductID a través de la línea de comandos:

$ qemu -usb -usbdevice host:<VendorID>:<ProductID> <otros parámetros>

o iniciar QEMU con soporte para dispositivos USB activados mediante

$ qemu –usb <otros parámetros>

después de iniciar la máquina virtual, cambiar al sistema de monitoreo de la máquina virtual presionando:

[Ctrl]+[Alt]+[2] e introducir el siguiente comando

usb_add host:<VendorID>:<ProductID>

cuando se retorne al ambiente gráfico al teclear *[Ctrl]+[Alt]+[1]*, se verá el mensaje de reconocimiento del dispositivo USB. Por ejemplo si se tiene una impresora HP Scanjet 3300C conectada en el puerto USB de la computadora, la salida del comando *lsub* es:

 # lsusb

Bus 003 Device 002: ID 03f0:0205 ScanJet 3300C

así, el comando en QEMU para dejar accesible el dispositivo es:

$ qemu -usb -usbdevice host:03f0:0205 <otros parámetros>

4) Usar la impresora conectada en el puerto paralelo, pera ello al invocar la ejecución de la máquina virtual usar:

$ qemu -parallel /dev/parport0 <otros parámetros>

5) Montar el contenido de un disco virtual y poder intercambiar información entre la máquina virtual y la huésped, primero convertir el disco a formato accesible a Linux:

$ qemu-img convert disco.img -O raw tmp.img

montar la imagen en Linux como root

 # mkdir disk

 # mount -o loop,offset=32256 tmp.img disk

trabajar con la imagen montada y al terminar desmontar esta

 # umount ./disk

y puede ser regresada al formato original mediante

$ qemu-img convert -c tmp.img -O qcow2 disco.img

Algunos Problemas Comunes con la Red

Por lo general las máquinas virtuales detectan correctamente la red, pero en el caso de Windows esto no siempre pasa, por ello es común emular una tarjeta de red lo más genérica posible, esta puede ser RTl8139, para ello es necesario que al lanzar la máquina virtual se indique *-net nic,model=rtl8139 -net user*, por ejemplo:

$ qemu -boot c -hda WindowsXP.img -m 400 -localtime -net nic,model=rtl8139 -net user

algunas de las otras opciones para la red son: NE2000 PCI, RTL8139, PCNET y NE2000 ISA.

Algunos Problemas con KVM

Si se detectan las banderas para virtualización por Hardware y al tratarlo de usar marca:

> open /dev/kvm: Permission denied

> Could not initialize KVM, will disable KVM support

entonces, sólo hay que agregar, el login del usuario al grupo kvm en el archivo */etc/group*.

> open /dev/kvm: No such file or directory

> Could not initialize KVM, will disable KVM support

entonces, sólo hay que activar en el BIOS la virtualización por Hardware.

Aumento de Desempeño

La virtualización normalmente es rápida, pero en algunas circunstancias se hace lenta generalmente esto es ajeno a QEMU o KVM y normalmente es por la constante grabación de datos al disco duro por parte de la máquina virtual. Para optimizar el desempeño de la máquina virtual es posible pedirle a QEMU o KVM que trate de usar un cache y baje lo menos posible a disco la información, esto aumentara notablemente el desempeño de la máquina virtual.

Para aumentar el desempeño, en lugar de usar:

$ qemu -boot c -hda WindowsXP.img -m 400 -localtime -net nic,model=rtl8139 -net user

usar en QEMU:

$ qemu -drive file=WindowsXP.img,cache=writeback,media=disk -m 400 -localtime -net nic,model=rtl8139 -net user

usar en KVM:

$ kvm -drive file=WindowsXP.img,cache=writeback,media=disk -m 400 -localtime -net nic,model=rtl8139 -net user

En el caso de usar un archivo ISO, usar:

$ kvm -m 512 -drive file=fedora.iso,cache=writeback,media=cdrom

Instalando UBUNTU 11.10 Mediante KVM

En este caso generaremos un disco de 10 GB, se carga Ubuntu y dentro de el procedemos a instalar Ubuntu en el disco recién generado, una vez terminada la instalación, podemos hacer uso de nuestra nueva imagen.

$ qemu-img create -f qcow2 disco.img 10G

$ kvm –no-reboot -boot d -drive file=ubuntu-11.10-desktop-i386.iso,cache=writeback,media=cdrom -drive file=disco.img,cache =writeback,media=disk -m 500

$ kvm -drive file=disco.img,cache=writeback,media=disk -m 500

aquí, se usa el cache para acelerar el desempeño de KVM.

Mejorando el Desempeño del Vídeo de la Máquina Virtual

Por omisión se tiene un tarjeta gráfica de pobre desempeño en la máquina virtual, si se necesita mayor resolución en la salida gráfica, una opción es usar la opción -VGA, donde dos de sus posibilidades es STD o VMWARE usándose como:

$ kvm -vga std -m 512 -drive file=fedora.iso,cache=writeback,media =cdrom

o

$ kvm -vga vmware -m 512 -drive file=fedora.iso,cache=writeback, media=cdrom

Usando QEMU o KVM con USB Live

Para el caso de tener un USB live y se quiera correr su contenido desde una máquina virtual con QEMU o KVM solo es necesario montar el USB, conocer el dispositivo mediante:

$ df

y usar ese dispositivo en:

$ kvm -usb /dev/sddx

Direcciones de Red Usadas en QEMU o KVM

Gateway/DHCP/TFTP Server: 10.0.2.2

DNS Server: 10.0.2.3

Samba Server: 10.0.2.4

Netmask: 255.255.255.0

Guest IP: any address above 10.0.2.15

Uso de Tarjeta de Sonido Dentro de QEMU o KVM

Por omisión el uso de la tarjeta de audio no esta habilitada, para habilitarla usar en la línea de comandos:

-soundhw sb16,es1370,adlib

por ejemplo:

$ qemu -boot c -hda Windows.img -m 400 -localtime -soundhw sb16,es1370,adlib

en KVM

$ kvm -boot c -hda Windows.img -m 400 -localtime -soundhw sb16,es1370,adlib

Uso de la Virtualización Dentro de una Virtualización

Algunas veces es necesario tener activa la virtualización dentro de otra virtualización, esto se logra mediante, por ejemplo, para un procesador AMD:

$ kvm -m 128 -cdrom TinyCore-current.iso -cpu phenom,+svm

Uso de VNC Como Salida Gráfica de la Virtualización

Si se desea usar VNC (véase [29]) como visualizador de la salida gráfica de KVM, primero se debe instalar un cliente de VNC, por ejemplo vncviewer, usando:

 # aptitude install vncviewer

Entonces, para correr la máquina virtual usamos:

$ kvm -m 128 -cdrom TinyCore-current.iso -cpu kvm64 -vnc :0

Y para ver la salida gráfica usamos

$ vncviewer 0

Uso de SSH para Correr una Máquina Virtual en Forma Remota

Si se tiene acceso a un servidor mediante SSH en el cual este activo X11 forwarding e instalado KVM/QEMU, entonces es posible correr una máquina remota en el servidor y visualizar la salida gráfica en la máquina donde se corre el comando SSH. Lo primero, al hacer la conexión mediante SSH, es necesario solicitar en la conexión se habilite X11 forwarding mediante:

$ssh -X -l usr 192.168.13.230

Después de hacer la conexión, ya podemos correr la máquina virtual como se indicó antes:

$ kvm -m 128 -cdrom TinyCore-current.iso &

Interacción de VirtualBox en KVM/QEMU

Uno de los manejadores de máquinas virtuales más conocidos es VirtualBox (véase [115],[116]) el cual dispone de diversas imágenes (véase [117]) funcionales listas para descargar y usar de varias decenas de distribuciones de Linux. Es por ello que KVM/QEMU ha desarrollado formas de usar, convertir y migrar máquinas de VirtualBox y otros manejadores de máquinas virtuales con un mínimo de esfuerzo, ejemplo de ello es que se puede descargar cualquier imagen VDI de VirtualBox y usarlo directamente en KVM usado la misma sintaxis que con sus propias máquinas virtuales.

Ejemplo de ello, es descargar LUBUNTU 12.10 (véase [118],[119]), descomprimir el archivo lubuntu1210.7z y esto dejará la imagen de VirtualBox de LUBUNTU cuyo nombre es lubuntu1210.vdi.

Entonces esta imagen (lubuntu1210.vdi) la usaremos directamente usando KVM/QEMU mediante:

$ kvm -m 2000 -hda lubuntu1210.vdi

o

$ qemu-system-x86_64 -enable-kvm -m 2000 -hda lubuntu1210.vdi

Nota: esta imagen usa: (username/password): lubuntu/lubuntu

Algunas veces es necesario montar y extraer el contenido de un disco virtual, supongamos que tenemos una máquina virtual de VirtualBox y queremos ver su contenido, para ello usamos:

$ qemu-img convert diskname.vmdk -O qcow2 diskname.qcow2

o para raw:

$ qemu-img convert diskname.vmdk -O raw diskname.raw

Instalar *nbd-client*

 # apt-get install nbd-client

Después:

 # qemu-nbd --connect=/dev/nbd0 /mnt/kvm/diskname.qcow2

 # fdisk /dev/nbd0 -l

 # sudo mount /deb/nbd0p1 /mnt/somepoint/

 # umount /mnt/somepoint/

Algunas Otras Opciones

Lanzar KVM con dos procesadores, 1536 MB de RAM, dispositivo de red e1000, MAC address 52:54:00:12:34:50, iniciando el DHCP en la dirección 10.0.2.40 y

reenviando la salida del puerto 22 de la maquina virtual al 5555 del equipo huésped, mediante:

```
$ kvm -smp 2 -drive file=debianStableTmp.img,cache=writeback,media=disk \
-m 1536 -device e1000,netdev=user.0,mac=52:54:00:12:34:50 \
-netdev user,id=user.0,dhcpstart=10.0.2.40,hostfwd=tcp::5555-:22 &
```

o lanzar kvm con dos procesadores, 1536 MB de RAM, dispositivo de red e1000 y reenviando la salida del puerto 22 de la maquina virtual al 5555 del equipo huésped

```
$ kvm -smp 2 -drive file=debianStableTmp.img,cache=writeback,media=disk
-m 1536 -device e1000,netdev=user.0 -netdev user,id=user.0, \
hostfwd=tcp::5555-:22 &
```

el redireccionamiento de puerto puede ser hecho también con:

```
$ kvm -m 512 -cpu phenom,+svm -hda b.qcow2 \
-redir tcp:5555:10.0.2.15:22 &
```

Si se desea usar ssh y scp en la máquina virtual usar:

```
# apt-get install openssh-server
```

Acceder usando:

```
$ ssh -p 5555 root@localhost
```

Y hacer copia del equipo huésped a la maquina virtual mediante:

```
$ scp -P 5555 file.txt usr@localhost:/tmp
```

A continuación se muestra la ejecución dentro de un anfitrión Linux la ejecución de una máquina virtual Linux y dentro de ella se ejecuta una máquina virtual con Windows, lo que se conoce cómo una virtualización dentro de otra virtualización, como se aprecia a continuación:

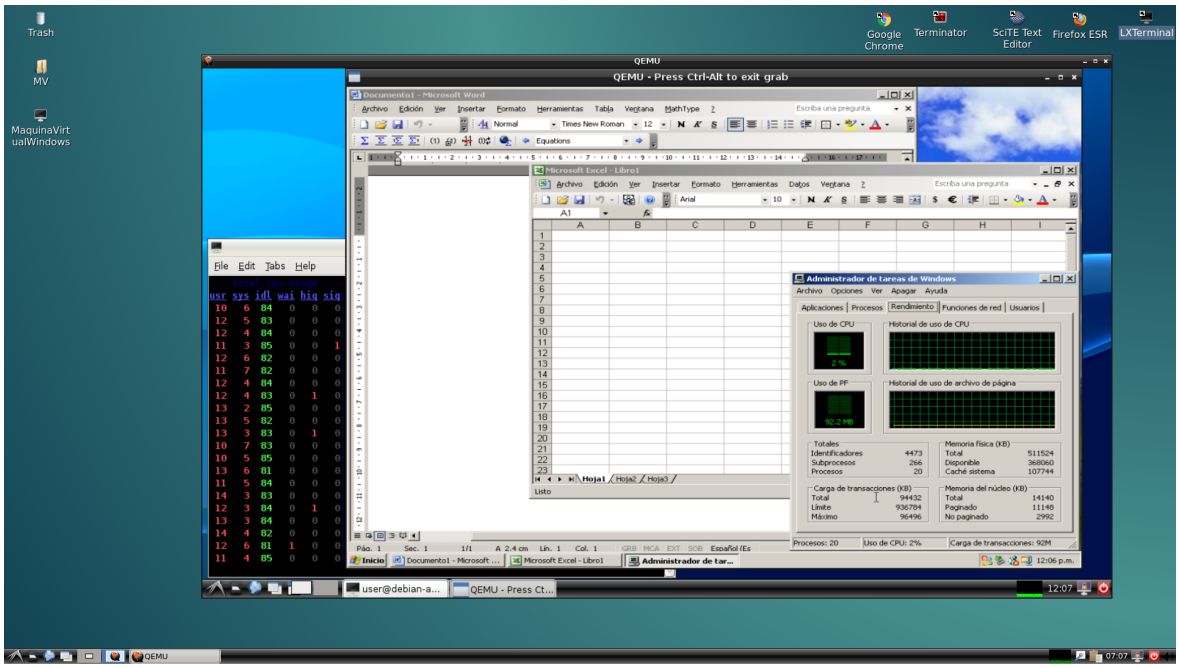

Figura 3.3 Múltiples sistemas virtualizados simultáneamente usando KVM y virtualización dentro de la virtualización.

En la imagen anterior se muestra una ejecución de Linux Debian Testing stretch versión 9 como sistema anfitrión y dentro de el, se corre mediante KVM una versión de Debian squeeze versión 6, y dentro de ella haciendo virtualización dentro de otra virtualización, se ejecuta un Windows XP en el que se corre un Office 2003 y se muestra la ejecución de Word y Excel simultáneamente. Adicionalmente se tiene en Windows el Administrador de tareas indicando el uso de CPU que se encuentra en 2%.

Acceso al Disco Compartido y Dispositivos USB y DVD para Profesores y Alumnos.

El acceso al disco compartido y demás recursos es necesario para tener acceso al disco compartido desde cualquier equipo Windows o Linux; y por otro lado, para permitir el acceso a los dispositivos *USB* y *DVD* desde las máquinas virtuales de Windows instaladas en los equipos anfitriones bajo el sistema operativo Linux Debian Estable en las aulas y talleres dentro de los edificios Tlahuizcalpan y Yelizcalli de la Facultad de Ciencias, UNAM.

El acceso al disco compartido, tiene dos variantes, una en la que los profesores y ayudantes pueden poner a disposición de los alumnos el material en formato de sólo lectura; y otra, en que los estudiantes pueden guardar archivos de forma temporal, a los cuales pueden acceder desde cualquier máquina (no importa el sistema operativo) conectada a la intranet de la Facultad de Ciencias, en especial desde las aulas y talleres dentro de los edificios de Tlahuizcalpan y Yelizcalli. Para acceder a los datos almacenados, disponemos de dos medios:

a) Mediante el uso del protocolo *HTTP*[29], usando un navegador de *Web* en *Windows* o Linux, en las direcciones:

Homologada: http://132.248.181.216/estud/

Local: http://192.168.13.230/estud/

Homologada: http://132.248.181.216/prof/

Local: http://192.168.13.230/prof/

b) Mediante el uso del protocolo *SAMBA*[30], para tener acceso a el, hay que conectarse a una unidad de red dentro del explorador de *Windows*. Para ello se abre el explorador, se selecciona *Herramientas*, para después solicitar conectarse a una unidad de red, lo cual mostrará la siguiente ventana:

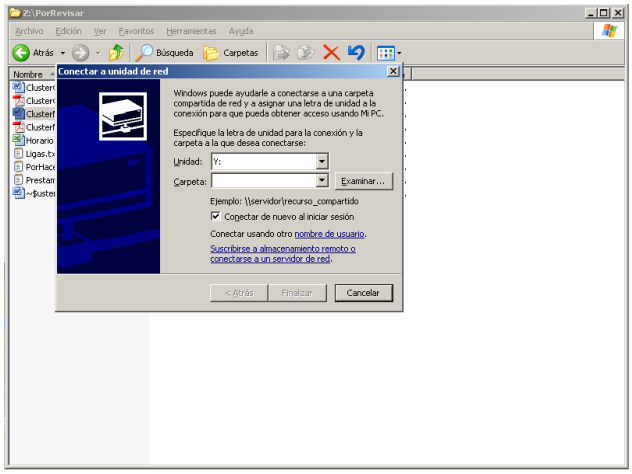

Figura 3.4 Ventana de conexión a una unidad de red

29 Protocolo de transferencia de hipertexto (Hypertext Transfer Protocol http)

30 Implementación libre del protocolo de archivos compartidos de Microsoft Windows para sistemas tipo Linux y UNIX

En este caso, para acceder como usuario ESTUD, se debe llenar con los siguientes datos:

Unidad: *Z:*

Carpeta: *192.168.13.230**estud*

Usuario: *estud*

Clave: *estud*

En este caso, para acceder como usuario PROF, se debe llenar con los siguientes datos:

Unidad: *Z:*

Carpeta: *192.168.13.230**prof*

Usuario: *prof*

Clave: *prof*

Una vez conectado se crea el directorio correspondiente a la materia y se guarda el material correspondiente. Si el material guardado ya no será usado posteriormente por los alumnos, debe ser borrado por el profesor ya que el disco duro es un recurso compartido y limitado.

Por otro lado, el acceso al *USB* y *DVD* de la máquina en el sistema anfitrión *Linux* Debian Estable no presenta ningún problema, en cuanto el dispositivo *USB* o *DVD* se inserte, es reconocido y esta listo para usarse mediante algún manejador de archivos (*Dolphin*, *Konqueror* o cualquier otro navegador de archivos disponible).

Para ver los dispositivos *USB* y *DVD* en la máquina virtual de *Windows*, previamente se debe montar estos en *Linux* y revisar mediante manejador de archivos que se tiene acceso a los datos del mismo (para intercambiar entre la máquina virtual de *Windows* y el sistema anfitrión de *Linux Debian Estable*, use las teclas *[Ctrl]* + *[Alt]* + *[Esc]*).

Para tener acceso a los datos en la máquina virtual de Windows, es necesario conectarse a una unidad de red dentro del explorador de *Windows*. Para ello hay que abrir el explorador, seleccionar *Herramientas*, en ellas, solicitar conectarse a una unidad de red con el usuario y clave de acceso con que se ingreso al equipo anfitrión en Linux, por ejemplo si se ingresa con el usuario *windows* y clave *windows* en los equipos de las aulas y talleres del Departamento de Matemáticas en el Tlahuizcalpan se usa:

Acceso al *USB* y *DVD*:

Unidad: *Y:*

Carpeta: *10.0.2.2**windows*

Usuario: *windows*

Clave: *windows*

Si se ingresa en el edificio Yelizcalli, el acceso se da mediante el usuario *actuaria* y clave *actuaria*:

Acceso al *USB* y *DVD*:

Unidad: *Y:*

Carpeta: *10.0.2.2**windows*

Usuario: *actuaria*

Clave: *actuaria*

Una vez conectado, es posible crear el directorio para almacenar los datos personales en el medio de almacenamiento externo y se guarda el material correspondiente, realizado durante la sesión de trabajo.

Las unidades de discos compartidos al igual que los dispositivos *USB* y *DVDS* tienen velocidades muy bajas de acceso a los datos, por ello, si se requiere hacer uso de uno o más archivos y en especial cuando los archivos son grandes, es recomendable copiar estos a la máquina virtual en cualquier directorio de la misma y trabajar con ellos de forma local. Para que al terminar de usarlos, si su

contenido fue modificado es necesario copiar dichos archivos al disco compartido o al dispositivo *USB* que se tenga montado.

Para desmontar la unidad *USB* se usa el manejador de archivos de *Linux* y con el botón derecho del *Mouse* se solicita que sea desmontado el dispositivo insertado. Los dispositivos montados en el sistema operativo *Linux* son visibles como una carpeta en el sistema operativo de *Windows*, por ello, no hay nada que desmontar en Windows.

CAPÍTULO 4. Conclusiones

Reflexiones

En las labores de Técnica Académica que he desempeñado durante los últimos 14 años, utilicé y he ampliado las diversas herramientas adquiridas dentro de la carrera de Ingeniería en Computación y gracias al amplio perfil de la misma (que permite incursionar en campos como la Docencia, la Técnica y Apoyo a la Investigación) me ayudo a desarrollar el esquema de procesamiento lógico, la capacidad de abstracción y de investigación durante mi estancia como estudiante en dicha carrera. Y me han permitido ahora como pasante, el participar en el proceso integral de proporcionar un marco adecuado y flexible de tecnologías de la información acorde a las necesidades de los académicos, investigadores y estudiantes que requieren el uso cotidiano de equipo de cómputo para prácticas académicas.

Durante el período en que fui ayudante de Profesor, aprendí de los profesores de Asignatura con quienes colaboré a realizar material de apoyo y prácticas de docencia (mismo que a su vez me sirvió como base al ser Profesor de Asignatura), parte fundamental de lo aprendido es la importancia de la motivación en los alumnos, es esencial interactuar con ellos para fomentar el interés por la materia y facilitar la comprensión hacia áreas no exploradas con anterioridad o las cuales se les dificulta por su falta de pericia y así, evitar la frustración y posible deserción de los alumnos.

Aterrizar conceptos abstractos con ejemplos cotidianos, ayuda a los alumnos a comprender mejor las bases teóricas, motivando así al grupo a avanzar si no al mismo nivel, con menor desventaja entre compañeros, aportando una comprensión equilibrada entre compañeros, el avance es a nivel personal, pero es mas sencillo si se cuentan con las herramientas básicas para comenzar y continuar aprendiendo.

Por otro lado, en las labores como Ayudante de Profesor y Profesor de Asignatura, usé herramientas adquiridas en varias materias, como son, las materias de Programación, Bases de Datos, Modelado de Estructuras de Datos, entre otras. Además de los cursos extra curriculares que he tomado, aunado a los conocimientos adquiridos de los profesores de la FES Aragón, con los cuales he complementado mi formación humana y profesional.

Desde que inicié mis labores como Técnica Académica Auxiliar de Medio Tiempo hasta ahora como Técnica Académica Asociada de Tiempo Completo, he ampliado mis habilidades en base a la continua practica y el cambio constante de las tecnologías de la información, aunado a lo aprendido de mis compañeros de trabajo y a la retroalimentación de los profesores a los cuales apoyamos, pues es un trabajo en equipo y colaborativo.

Además, durante las distintas designaciones laborales de apoyo a las carreras del Departamento de Matemáticas, he aplicado los diversos conocimientos técnicos-académicos adquiridos en la carrera de Ingeniería en Computación. La formación integral adquirida durante mis estudios me dio la capacidad de atender problemas reales, así como la facilidad de tener un aprendizaje continuo y autodidacta en temas diversos, así como la importancia del trabajo en equipo.

Todo en conjunto, me ha ayudado a resolver las distintas problemáticas que se han presentado en cada período del área laboral, desde que comencé mi colaboración en el Departamento de Matemáticas, como Técnica Académica en una sala de cómputo que daba servicio a los usuarios del Departamento, hasta ahora que se cuentan con 20 Aulas, Talleres y Laboratorios para satisfacer los requerimientos de cómputo ante el incremento de usuarios.

En los años que llevo trabajando en el Departamento de Matemáticas de la Facultad de Ciencias, he aprendido que es indispensable actualizarse constantemente para aprovechar las nuevas tecnologías de la información y su rápido cambio en cuanto a la gran cantidad de recursos computacionales que son puestos a disposición de profesores, ayudantes y alumnos; tratando de dar un marco integral para que se aproveche al máximo las herramientas

computacionales.

El tener la flexibilidad para aprender nuevas tecnologías y reaprender de los cambios de las existentes facilita el aprovechamiento de los recursos con que se cuentan. Reinventar la forma de trabajo siempre es esencial para avanzar si no al mismo tiempo en que se desarrollan las tecnologías, para utilizar al máximo los recursos que tenemos para resolver los problemas que se presenten.

Dentro de las principales actividades que he realizado en mis labores profesionales destacan:

- Apoyo académico en los talleres, aulas y laboratorios de cómputo del Departamento de Matemáticas en el Edificio de Investigación y Docencia Experimental Tlahuizcalpan
- Apoyar a los profesores a fin de que cada curso cuente con una página
- Canalizar reportes de fallas y anomalías a las instancias correspondientes
- Apoyar a los coordinadores de carrera en la conformación de horarios en:
    - Cursos Programados durante el semestre
    - Clases esporádicas en las que se requiera acceso y Software específico
    - Mantener actualizados los horarios durante todo el semestre
- impartir cursillos (Excel, Látex, Linux, MatLab, etc.) y elaborar notas para dichos cursillos
- Auxiliar a los grupos que lo requieran en el desarrollo de prácticas específicas
- Apoyar la actualización de ambientes de trabajo de los equipos
- Aclarar dudas de los usuarios en cuanto al uso del Software
- Elaborar documentos tipos manuales y "Preguntas frecuentes"
- Elaborar documentos tipo manuales que describan la equivalencia de Software libre con Software comercial
- Apoyo en la organización de redes de cómputo acorde a las necesidades de cada carrera
- Dominio de diversas tecnologías de la información y su uso en las diversas

áreas a las que atendemos

- Impartir cursos como ayudante y profesor de asignatura

Debido a lo expuesto anteriormente, considero que la formación de los Ingenieros en Computación estamos capacitados tanto para la docencia, apoyo a la investigación, así como para áreas técnicas, proporcionando los servicios necesarios para el uso y comprensión de las tecnologías de la información en la docencia y la investigación en áreas de la Ciencia y la Tecnología; y en particular en instituciones educativas como la Facultad de Ciencias de la UNAM.

Apéndice A. Software Libre y Propietario

Hoy en día los usuarios disponemos de una gran variedad de opciones en cuanto a Software se refiere. Por un lado, podemos emplear programas comerciales que en general nos venden licencias para el uso del Software, la cual, es restrictiva y restringe su uso a una sola máquina o conjunto determinado de ellas.

Por otro lado, existe el Software libre, desarrollados por usuarios y para usuarios que, entre otras cosas, comparten los códigos fuente, el programa ejecutable y dan libertades para estudiar, adaptar y redistribuir a quien así lo requiera el programa y todos sus derivados.

A.1. Software Libre

La definición de Software libre (véase [10], [20], [31], [104], [105], [106], [107] y [108]) estipula los criterios que se tienen que cumplir para que un programa sea considerado libre. De vez en cuando se modifica esta definición para clarificarla o para resolver problemas sobre cuestiones delicadas. «Software libre» significa que el Software respeta la libertad de los usuarios y la comunidad. En términos generales, los usuarios tienen la libertad de copiar, distribuir, estudiar, modificar y mejorar el Software. Con estas libertades, los usuarios -tanto individualmente como en forma colectiva- controlan el programa y lo que hace.

Cuando los usuarios no controlan el programa, el programa controla a los usuarios. El programador controla el programa y, a través del programa, controla a los usuarios. Un programa que no es libre, llamado «privativo o propietario», y

es considerado por muchos como un instrumento de poder injusto.

El Software libre es la denominación del Software que respeta la libertad de todos los usuarios que adquirieron el producto y, por tanto, una vez obtenido el mismo puede ser usado, copiado, estudiado, modificado, y redistribuido libremente de varias formas. Según la Free Software Foundation (véase [104]), el Software libre se refiere a la libertad de los usuarios para ejecutar, copiar, distribuir, y estudiar el mismo, e incluso modificar el Software y distribuirlo modificado.

Un programa es Software libre si los usuarios tienen las cuatro libertades esenciales:

0. La libertad de usar el programa, con cualquier propósito.
1. La libertad de estudiar cómo funciona el programa y modificarlo, adaptándolo a tus necesidades.
2. La libertad de distribuir copias del programa, con lo cual puedes ayudar a tu prójimo.
3. La libertad de mejorar el programa y hacer públicas esas mejoras a los demás, de modo que toda la comunidad se beneficie.

Un programa es Software libre si los usuarios tienen todas esas libertades. Por tanto, el usuario debe ser libre de redistribuir copias, tanto con como sin modificaciones, ya sea gratuitamente o cobrando una tarifa por la distribución, a cualquiera en cualquier parte. El ser libre de hacer estas cosas significa, entre otras cosas, que no tiene que pedir ni pagar el permiso.

También debe tener la libertad de hacer modificaciones y usarlas en privado para su propio trabajo o pasatiempo, sin siquiera mencionar que existen. Si publica sus cambios, no debe estar obligado a notificarlo a nadie en particular, ni de ninguna manera en particular.

La libertad de ejecutar el programa significa que cualquier tipo de persona u organización es libre de usarlo en cualquier tipo de sistema de computación, para cualquier tipo de trabajo y finalidad, sin que exista obligación alguna de comunicarlo al programador ni a ninguna otra entidad específica. En esta libertad,

lo que importa es el propósito de los usuarios, no el de los programadores. El usuario es libre de ejecutar el programa para alcanzar sus propósitos, y si lo distribuye a otra persona, también esa persona será libre de ejecutarlo para lo que necesite; nadie tiene derecho a imponerle sus propios objetivos.

La libertad de redistribuir copias debe incluir las formas binarias o ejecutables del programa, así como el código fuente, tanto para las versiones modificadas como para las no lo estén. Distribuir programas en forma de ejecutables es necesario para que los sistemas operativos libres se puedan instalar fácilmente. Resulta aceptable si no existe un modo de producir un formato binario o ejecutable para un programa específico, dado que algunos lenguajes no incorporan esa característica, pero debe tener la libertad de redistribuir dichos formatos si encontrara o programara una forma de hacerlo.

Para que la libertad 1 y 3 de realizar cambios y publicar las versiones modificadas tengan sentido, el usuario debe tener acceso al código fuente del programa. Por consiguiente, el acceso al código fuente es una condición necesaria para el Software libre. El «código fuente» ofuscado no es código fuente real, y no cuenta como código fuente.

La libertad 1 incluye la libertad de usar su versión modificada en lugar de la original. Si el programa se entrega con un producto diseñado para ejecutar versiones modificadas de terceros, pero rechaza ejecutar las suyas, una práctica conocida como «tivoización» o «arranque seguro» [«lockdown»] la libertad 1 se convierte más en una ficción teórica que en una libertad práctica.

Esto no es suficiente. En otras palabras, estos binarios no son Software libre, incluso si se compilaron desde un código fuente que es libre.

Una manera importante de modificar el programa es agregándole subrutinas y módulos libres ya disponibles. Si la licencia del programa específica que no se pueden añadir módulos que ya existen y que están bajo una licencia apropiada, por ejemplo si requiere que usted sea el titular de los derechos de autor del

código que desea añadir, entonces se trata de una licencia demasiado restrictiva como para considerarla libre.

La libertad 3 incluye la libertad de publicar sus versiones modificadas como Software libre. Una licencia libre también puede permitir otras formas de publicarlas; en otras palabras, no tiene que ser una licencia de copyleft. No obstante, una licencia que requiera que las versiones modificadas no sean libres, no se puede considerar libre.

«Software libre» no significa que «no es comercial». Un programa libre debe estar disponible para el uso comercial, la programación comercial y la distribución comercial. La programación comercial de Software libre ya no es inusual; tal Software libre comercial es muy importante. Puede haber pagado dinero para obtener copias de Software libre, o puede haber obtenido copias sin costo. Pero sin tener en cuenta cómo obtuvo sus copias, siempre tiene la libertad de copiar y modificar el Software, incluso de vender copias.

El término Software no libre se emplea para referirse al Software distribuido bajo una licencia de Software más restrictiva que no garantiza estas cuatro libertades. Las leyes de la propiedad intelectual reservan la mayoría de los derechos de modificación, duplicación y redistribución para el dueño del copyright; el Software dispuesto bajo una licencia de Software libre rescinde específicamente la mayoría de estos derechos reservados.

Los manuales de Software deben ser libres por las mismas razones que el Software debe ser libre, y porque de hecho los manuales son parte del Software. También tiene sentido aplicar los mismos argumentos a otros tipos de obras de uso práctico; es decir, obras que incorporen conocimiento útil, tal como publicaciones educativas y de referencia. La Wikipedia es el ejemplo más conocido.

Tanto la Open Source Initiative como la Free Software Foundation mantienen en sus páginas Web (véase [19], [20], [31], [104] y [105]) listados oficiales de las licencias de Software libre que aprueban.

Tipos de Licencias

Una licencia es aquella autorización formal con carácter contractual que un autor de un Software da a un interesado para ejercer "actos de explotación legales". Pueden existir tantas licencias como acuerdos concretos se den entre el autor y el licenciatario. Desde el punto de vista del Software libre, existen distintas variantes (decenas) del concepto o grupos de licencias, de las más importantes tenemos:

Licencias GPL

Una de las más utilizadas es la Licencia Pública General de GNU[31]. El autor conserva los derechos de autor (copyright), y permite la redistribución y modificación bajo términos diseñados para asegurarse de que todas las versiones modificadas del Software permanecen bajo los términos más restrictivos de la propia GNU GPL. Esto hace que sea imposible crear un producto con partes no licenciadas GPL: el conjunto tiene que ser GPL.

En la práctica esto hace que las licencias de Software libre se dividan en dos grandes grupos, aquellas que pueden ser mezcladas con código licenciado bajo GNU GPL (y que inevitablemente desaparecerán en el proceso, al ser el código resultante licenciado bajo GNU GPL) y las que no lo permiten al incluir mayores u otros requisitos que no contemplan ni admiten la GNU GPL y que por lo tanto no pueden ser enlazadas ni mezcladas con código gobernado por la licencia GNU GPL.

Licencias Estilo BSD

La licencia BSD (Berkeley Software Distribution)[32]. Son llamadas así porque se utilizan en gran cantidad de Software distribuido junto a los sistemas operativos BSD. El autor, bajo tales licencias, mantiene la protección de copyright únicamente para la renuncia de garantía y para requerir la adecuada atribución de

[31] https://www.gnu.org/licenses/licenses.es.html#GPL

[32] http://directory.fsf.org/wiki/License:BSD_4Clause

la autoría en trabajos derivados, pero permite la libre redistribución y modificación, incluso si dichos trabajos tienen propietario. Son muy permisivas, tanto que son fácilmente absorbidas al ser mezcladas con la licencia GNU GPL con quienes son compatibles. Puede argumentarse que esta licencia asegura "verdadero" Software libre, en el sentido que el usuario tiene libertad ilimitada con respecto al Software, y que puede decidir incluso redistribuirlo como no libre.

Copyleft

En copyleft[33] o licencia protectora hay que hacer constar que el titular de los derechos de autor de un Software bajo licencia copyleft puede también realizar una versión modificada bajo su copyright original, y venderla bajo cualquier licencia que desee, además de distribuir la versión original como Software libre. Esta técnica ha sido usada como un modelo de negocio por una serie de empresas que realizan Software libre (por ejemplo MySQL); esta práctica no restringe ninguno de los derechos otorgados a los usuarios de la versión copyleft.

Comparación con el Software de Código Abierto

Aunque en la práctica el Software de código abierto y el Software libre comparten muchas de sus licencias, la Free Software Foundation opina que el movimiento del Software de código abierto es filosóficamente diferente del movimiento del Software libre. Los defensores del término "código abierto", en inglés open source, afirman que éste evita la ambigüedad del término en ese idioma que es free en free Software.

Mucha gente reconoce el beneficio cualitativo del proceso de desarrollo de Software cuando los desarrolladores pueden usar, modificar y redistribuir el código fuente de un programa. El movimiento del Software libre hace especial énfasis en los aspectos morales o éticos del Software, viendo la excelencia técnica como un producto secundario de su estándar ético. El movimiento de código abierto ve la excelencia técnica como el objetivo prioritario, siendo la compartición del código fuente un medio para dicho fin. Por dicho motivo, la FSF

[33] https://copyleft.org/

se distancia tanto del movimiento de código abierto como del término "Código Abierto" (en inglés Open Source).

Puesto que la OSI sólo aprueba las licencias que se ajustan a la Open Source Definition (definición de código abierto), la mayoría de la gente lo interpreta como un esquema de distribución, e intercambia libremente "código abierto" con "Software libre". Aún cuando existen importantes diferencias filosóficas entre ambos términos, especialmente en términos de las motivaciones para el desarrollo y el uso de tal Software, raramente suelen tener impacto en el proceso de colaboración.

Aunque el término "código abierto" elimina la ambigüedad de libertad frente a precio (en el caso del inglés), introduce una nueva: entre los programas que se ajustan a la definición de código abierto, que dan a los usuarios la libertad de mejorarlos, y los programas que simplemente tiene el código fuente disponible, posiblemente con fuertes restricciones sobre el uso de dicho código fuente. Mucha gente cree que cualquier Software que tenga el código fuente disponible es de código abierto, puesto que lo pueden manipular. Sin embargo, mucho de este Software no da a sus usuarios la libertad de distribuir sus modificaciones, restringe el uso comercial, o en general restringe los derechos de los usuarios.

A.2. Software Propietario

No existe consenso sobre el término a utilizar para referirse al opuesto del Software libre. La expresión Software propietario[34] proviene del término en inglés "proprietary Software". En la lengua anglosajona, "proprietary" significa «poseído o controlado privadamente» («privately owned and controlled»), que destaca la manutención de la reserva de derechos sobre el uso, modificación o redistribución del Software. Inicialmente utilizado, pero con el inconveniente que la acepción proviene de una traducción literal del inglés, no correspondiendo su uso como

[34] http://www.linfo.org/proprietary.html

adjetivo en el español, de manera que puede ser considerado como un barbarismo.

El término "propietario" en español resultaría inadecuado, pues significa que «tiene derecho de propiedad sobre una cosa», por lo que no podría calificarse de "propietario" al Software, porque éste no tiene propiedad sobre nada (es decir, no es dueño de nada) y, además, no podría serlo (porque es una cosa y no una persona). Así mismo, la expresión "Software propietario" podría ser interpretada como "Software sujeto a propiedad" (derechos o titularidad) y su opuesto, el Software libre, también está sujeto al derecho de autor. Otra interpretación es que contrariamente al uso popular del término, se puede afirmar de que "todo Software es propietario", por lo que la forma correcta de referirse al Software con restricciones de uso, estudio, copia o mejora es la de Software privativo, según ésta interpretación el término "propietario" podría aplicarse tanto para Software libre como Software privativo, ya que la diferencia entre uno y otro está en que el dueño del Software privativo lo licencia como propiedad privada y el de Software libre como propiedad social.

Con la intención de corregir el defecto de la expresión "Software propietario" aparece el llamado "Software con propietario", sin embargo se argumenta contra del término "con propietario" justamente su similitud con proprietary en inglés, que sólo haría referencia a un aspecto del Software que no es libre, manteniendo una de las principales críticas a éste (de "Software sujeto a derechos" o "propiedad"). Adicionalmente, si "propietario" refiere al titular de los derechos de autor (y está claro que no puede referir al usuario, en tanto éste es simplemente un cesionario), no resuelve la contradicción: todo el Software libre tiene también titulares de derechos de autor.

La expresión Software no libre (en inglés non-free Software) es usado por la FSF para agrupar todo el Software que no es libre, es decir, incluye al llamado en inglés "semi-free Software" (Software semilibre) y al "propietary Software". Asimismo, es frecuentemente utilizado para referirse al Software que no cumple con las Directrices de Software libre de Debian, las cuales siguen la misma idea básica de libertad en el Software, propugnada por la FSF, y sobre las cuales está

basada la definición de código abierto de la Open Source Initiative.

Adicionalmente el Software de código cerrado nace como antónimo de Software de código abierto y por lo tanto se centra más en el aspecto de ausencia de acceso al código que en los derechos sobre el mismo. Éste se refiere sólo a la ausencia de una sola libertad por lo que su uso debe enfocarse sólo a este tipo de Software y aunque siempre signifique que es un Software que no es libre, no tiene que ser Software de código cerrado.

La expresión Software privado es usada por la relación entre los conceptos de tener y ser privado. Este término sería inadecuado debido a que, en una de sus acepciones, la palabra "privado" se entiende como antónimo de "público", es decir, que «no es de propiedad pública o estatal, sino que pertenece a particulares», provocando que esta categoría se interpretara como no referente al Estado, lo que produciría la exclusión del Software no libre generado por el aparato estatal. Además, el "Software público" se asocia generalmente con Software de dominio público.

A.3. Implicaciones Económico-Políticas

Una vez que un producto de Software libre ha empezado a circular, rápidamente está disponible a un costo muy bajo. Al mismo tiempo, su utilidad no decrece. El Software, en general, podría ser considerado un bien de uso inagotable, tomando en cuenta que su costo marginal es pequeñísimo y que no es un bien sujeto a rivalidad (la posesión del bien por un agente económico no impide que otro lo posea).

Puesto que el Software libre permite el libre uso, modificación y redistribución, a menudo encuentra un hogar entre usuarios para los cuales el coste del Software no libre es a veces prohibitivo, o como alternativa a la piratería. También es sencillo modificarlo localmente, lo que permite que sean posibles los esfuerzos de traducción a idiomas que no son necesariamente rentables comercialmente.

La mayoría del Software libre se produce por equipos internacionales que

cooperan a través de la libre asociación. Los equipos están típicamente compuestos por individuos con una amplia variedad de motivaciones, y pueden provenir tanto del sector privado, del sector voluntario o del sector público.

En México el Software Libre nació en las Universidades y los Centros de Investigación. Es por eso que, desde hace tres décadas, los estudiantes y los profesores usan Software libre para fines didácticos y de investigación. Las universidades suelen optar por el uso de Software libre en vez de utilizar Software privativo porque satisface de una mejor manera sus necesidades de cómputo, dada su naturaleza de apertura del código y la libertad de compartir los resultados obtenidos. De forma colateral, no se tienen gastos adicionales derivados del pago de licenciamientos.

Computólogos, Físicos, Químicos, Matemáticos, Actuarios y otros profesionistas y científicos utilizan Software libre como herramienta de investigación y creación. Un claro ejemplo de ello es la llamada Delta Metropolitana, que es una red de supercomputadores que están en varios puntos de la Ciudad de México, en el CINESTAV, el IPN, la UAM y la UNAM. Esa red de supercómputo utiliza Software libre para consolidar sus recursos, hacer investigación y generar conocimiento.

Por otro lado, dadas las características del Software de código cerrado, un usuario común ignora absolutamente el contenido del mismo y por tanto si existe dentro de las líneas del código alguna amenaza contra su equipo o su información, además el usuario no sólo tiene prohibido el intentar eliminar o cambiar esa parte del código sino que puede ser perseguido por la ley por el hecho de intentar conocer si existe tal amenaza en dicho Software.

Además, en una sociedad de la información, el Software se ha convertido en una herramienta importante de productividad, y una licencia de Software privativo constituye un acuerdo o contrato entre dos sujetos jurídicos que voluntariamente acuerdan las condiciones de uso de un programa, pero el costo económico de dicha licencia es cada vez más alto y en el caso de instituciones educativas, gubernamentales y sociedades civiles es en la mayoría de los casos -por decir lo menos- prohibitivo.

Por otro lado, se anunció en varios periódicos de circulación nacional (véase [109]) que:

> El Instituto Mexicano de la Propiedad Industrial (IMPI) anunció que en las próximas semanas dará inicio una serie de clausuras a negocios que utilicen Software licenciado de manera ilegal; esto como parte del acuerdo que tiene la dependencia con The Software Alliance (BSA) desde el 2002, el cual busca fomentar el uso de programas informáticos legales y disminuir el índice de piratería en el país.

De acuerdo a la BSA, el porcentaje de Software ilegal utilizado en el territorio aún se ubica en un 56 por ciento, cifra considerablemente menor a lo visto en el 2005, cuando el número ascendía a más del 65 por ciento. Sin embargo, México continúa siendo uno de los mayores compradores de piratería a nivel mundial, y lo que se busca con este tipo de acciones en el 2013 es disminuir, al menos, un punto porcentual.

"Si como consecuencia de una visita de inspección completa se encuentra la existencia de Software ilegal, se procede a la sanción. En el 2012 incrementaron hasta un 200% las sanciones por el uso ilegal de Software", dijo Kiyoshi Tsuru, director general en México de la BSA.

Aquí es donde algunos se preguntarán, ¿y qué autoridad tiene The Software Alliance para ejecutar estas determinaciones? Pese a que cuenta con el apoyo de empresas como Microsoft, Apple, Autodesk, Adobe, Aveva, AVG, CISCO, Dell, Hewlett Packard, IBM, SAP y Symntec, lo cierto es que la BSA no puede clausurar legalmente ningún negocio. La verdadera autoridad llega en su acuerdo con el IMPI, el cual sí tiene las facultades para aplicar sanciones.

Además, la UNAM, desde junio 9 del 2009 firmó un acuerdo (véase [110]):

> Con el objetivo de fomentar la cultura de la legalidad en lo que se refiere al uso del Software entre los estudiantes, la Universidad Nacional Autónoma de México y la Business Software Alliance (BSA) firmaron un convenio

general de colaboración.

Mediante este acuerdo, se promoverá el uso ético de las tecnologías de la información y comunicación, y se compartirán conocimientos en materia de propiedad intelectual y Software, a fin de apoyar el desarrollo y explotación de bienes digitales en la UNAM, así como definir programas para contribuir al avance de un mundo digital seguro, informaron ambas organizaciones en un comunicado.

El secretario general de la máxima casa de estudios, Sergio M. Alcocer Martínez de Castro, reconoció que la UNAM necesita hacer un esfuerzo en el ámbito institucional y en cada una de las entidades que la conforman, para brindar educación a sus alumnos, profesores y trabajadores en este campo.

"Se pretende", destacó, "que el convenio sea operativo y que se desarrolle en cada una de las entidades con la impartición de cursos y capacitación y en una rendición de cuentas para que el Software que se utilice sea legal".

El funcionario reconoció a los miembros de Business Software Alliance en Latinoamérica, y apuntó que la Universidad Nacional hará lo necesario para facilitar el uso responsable, ético y seguro del Software.

Informó también que ambas partes se reunirán seis meses después del inicio de este convenio de colaboración para analizar los avances.

En tanto, el director General de BSA-México, Kiyoshi Tsuru Alberú, resaltó que con la firma de este convenio podrán impulsar un plan conjunto relacionado con alta tecnología, ética y legalidad "Estamos seguros que en el mediano plazo se tendrán resultados importantes y se podrá hacer la diferencia", señaló.

Por su parte, el abogado general, Luis Raúl González Pérez, comentó que la UNAM busca difundir estos valores entre su comunidad, y llegar especialmente a los estudiantes que inician el bachillerato, porque desde edad temprana es importante fomentar la cultura de la legalidad.

Ante este escenario, una alternativa viable podría ser optar por el Software Libre, aunque, pese a su incipiente desarrollo es seguro que en un futuro podría alcanzar a suplir todas las necesidades básicas de los usuarios, dejando la adquisición de paquetes especializados sólo para los cursos avanzados que justifique el uso de Software privativo.

Apéndice B. Planes de Estudio

B.1. Programación I (véase [4])

Conocerá las ideas básicas de la programación y la solución algorítmica de problemas, tales como el diseño descendente, refinamientos sucesivos, abstracción de procedimientos, estructuras de control, tipos de datos y convenciones de entrada y salida básicos. Explicará las características sintácticas y de ejecución de un lenguaje de programación moderno, y las aplicará en la construcción y ejecución de programas completos que resuelven problemas algorítmicos sencillos. Identificará las representaciones básicas de datos numéricos y no numéricos directamente en la máquina, así como su aplicación concreta.

PROGRAMACIÓN I

CLAVE:		**SECTOR :**	BÁSICO
SEMESTRE:	2	**ÁREA:**	INFORMÁTICA
CRÉDITOS:	10	**SERIACIÓN:**	
		ASIGNATURA PRECEDENTE INDICATIVA: Cálculo Diferencial e Integral I, Álgebra Superior I y Geometría Analítica I.	
		ASIGNATURA SUBSECUENTE INDICATIVA: Programación II	
HORAS POR CLASE		**TEÓRICA:** 1	**PRÁCTICAS:** 2
CLASES POR SEMANA		**TEÓRICA:** 4	**PRÁCTICAS:** 1
HORAS POR SEMESTRE		**TEÓRICA:** 64	**PRÁCTICAS:** 32

Objetivos generales: Al final del curso el alumno:

- Conocerá las ideas básicas de la programación y la solución algorítmica de problemas, tales como el diseño descendente, refinamientos sucesivos, abstracción de procedimientos, estructuras de control, tipos de datos y convenciones de entrada y salida básicos.
- Explicará las características sintácticas y de ejecución de un lenguaje de programación moderno, y las aplicará en la construcción y ejecución de programas completos que resuelven problemas algorítmicos sencillos.
- Identificará las representaciones básicas de datos numéricos y no numéricos directamente en la máquina, así como su aplicación concreta.
- Conocerá los tipos de datos elementales y estructurados, así como los principios necesarios para la creación de tipos de datos definidos por el usuario y sus aplicaciones.
- Comprenderá y aplicará los fundamentos de los algoritmos recursivos en el proceso de solución de problemas.
- Comparará algunos algoritmos para búsquedas y ordenamientos lineales, y conocerá los problemas de complejidad y la contraposición entre espacio requerido y tiempo de ejecución.
- Reconocerá el contexto social en el cual la computación interacciona y sirve a la sociedad.

Tema 1. Conceptos fundamentales en métodos para la solución de problemas

Explicará las ideas básicas de programación y solución algorítmica de problemas. **19 horas teóricas**
9 horas teórico-prácticas

1.1. Abstracción de procedimientos; parámetros.
1.2. Estructuras de control: selección, iteración, recursividad.
1.3. Tipos de datos (i.e. números, cadenas, booleanos) y sus usos en la solución de problemas.
1.4. El proceso de diseño de programas; desde la especificación hasta la instrumentación; refinamientos sucesivos; representación gráfica.

Tema 2. Introducción a un lenguaje de programación

12 horas teóricas
7 horas teórico-prácticas

Comprenderá las características y la utilización de un lenguaje de programación.

2.1. Declaración de tipos básicos.
2.2. Operadores aritméticos y de asignación.
2.3. Enunciados condicionales.
2.4 Iteraciones y recursividad.
2.5 Procedimientos, funciones y parámetros.
2.6 Arreglos y registros.
2.7 Estructura general de un programa.

Tema 3. Representación de datos a nivel de la máquina **6 horas teóricas**
Reconocerá las representaciones básicas de datos en máquina. **3 horas teórico-prácticas**

3.1. Representación de datos numéricos, esto es, binario, octal, hexadecimal, punto fijo, complemento a 1 y 2, con signo, punto flotante, decimal, BCD, XS3.
3.2. Datos no numéricos como alfanuméricos, ASCII, ISO.

Tema 4. Representación de tipos de datos **6 horas teóricas**
3 horas teórico-prácticas
Explicará las características de los distintos tipos de datos y la manera de crearlos.

4.1 Selección y representación de tipos de datos elementales: enteros, reales, booleanos, carácter.
4.2 Especificación y representación de tipos de datos estructurados: arreglos, registros, conjuntos.

Tema 5. Algoritmos recursivos **8 horas teóricas**
4 horas teórico-prácticas
Reconocerá los fundamentos y usos de los algoritmos recursivos y su aplicación en la solución de problemas.

5.1 Introducción a algoritmos recursivos.
5.2 Conexión con la inducción matemática.
5.3 Comparación entre algoritmos recursivos e iterativos.

Tema 6. Búsquedas y ordenamientos lineales **8 horas teóricas**
4 horas teórico-prácticas
Comparará algunos algoritmos e identificará algunos problemas importantes.

6.1 Algoritmo de ordenamientos de selección e inserción en arreglos a través de apuntadores, con asignación dinámica de memoria; complejidad en el tiempo y el espacio; mejores y peores casos.
6.2 Búsqueda lineal, búsqueda binaria y arboles binarios de búsqueda; complejidad en el tiempo y el espacio; mejores y peores casos.

Tema 7. Contexto histórico y social de la computación **5 horas teóricas**
2 horas teórico-prácticas
Explicará el contexto social en el que se desenvuelve la computación, y los compromisos de esta con la sociedad.

7.1 Contexto social e histórico de la computación.
7.2 Definición de su área de estudio y actividades profesionales.
7.3 Similitudes y diferencias con otras disciplinas científicas y profesionales.
7.4 Uso, mal uso y límites de la tecnología computacional.
7.5 Responsabilidades sociales (seguridad y privacía).
7.6 Tipos de riesgos: errores latentes, seguridad y privacía, mal uso, etcétera.

Bibliografía básica:

- Tucker, A.B. *Fundamental of Computing, I: Logic, Problem Solving, Programs and Computers.* 2nd edition. USA. McGraw-Hill. 1994.

- Warfords, S. J. *Computer Science.* USA. D.C. Heat and Company. 1991.

Bibliografía complementaria:

- Ledgard H. with Tauer J. *Professional Software Enginnering Concepts.* Vol.1. USA. Addison Wesley Publishing Company. 1987.
- Ledgard H. with Tauer J. *Professional Software, Programming Practice.* Vol. II. USA. Addison Wesley Publishing Company. 1987.
- Naur, P. *A Human Activity.* USA. ACM Press. Addison Wesley Publishing Company. 1992.
- Salmon, W.I. *Structures and Abstractions: An Introduction to Computer Science with Pascal.* USA. Richard D. Irwing, Inc. 1991.
- Warnier, J.D. *Logical Construction of Systems.* USA. Van Nostrand Reinhold Company. 1981.
- Ioren H. *Introduction to Computer Architecture and Organization.* 2nd edition. USA. John Wiley and Sons. Inc. 1989.

Sugerencias didácticas:

Se recomiendan:
Tareas semanales en las cuales el alumno aplique el material visto en clase.
Trabajos de investigación bibliográfica para que el alumno amplíe sus conocimientos y conozca diferentes enfoques del material del curso.
Prácticas de cómputo para la experimentación con los algoritmos vistos en clase.
Análisis y resolución de casos prácticos.

Forma de evaluación:

Se recomiendan de 3 a 4 examenes parciales y un examen final, así como la realización de tareas sobre los temas vistos en clase para reforzar los conocimientos teóricos adquiridos.

Perfil profesiográfico:

Egresado preferentemente de las licenciaturas en Ciencias de la Computación, Ingeniería en Sistemas, Matemáticas, Actuaría o alguna afín con experiencia docente y profesional en el área.

B.2. Programación II (véase [4])

Conocerá y aplicará los métodos y técnicas para especificación, validación y verificación de Software. Será introducido al concepto de tipo abstracto de dato (abstract data type). Estará capacitado para comprender los fundamentos relativos al concepto, la implantación y las aplicaciones de las estructuras de datos básicas- listas, arreglos tablas, pilas o stacks, colas, árboles, gráficas y los operadores asociados a ellas. Será introducido a la definición, implantación y aplicaciones de las estructuras de datos no lineales y los operadores asociados a ellas. Comparará distintos algoritmos para búsquedas y ordenamientos, con especial atención al balance entre complejidad y espacio contra tiempo.

PROGRAMACIÓN II

CLAVE:		**SECTOR :**	BÁSICO
SEMESTRE:	3	**ÁREA:**	INFORMÁTICA
CRÉDITOS:	10	**SERIACIÓN:**	
		ASIGNATURA PRECEDENTE INDICATIVA: Programación I.	
		ASIGNATURA SUBSECUENTE INDICATIVA: Ninguna	
HORAS POR CLASE		**TEÓRICA:** 1	**PRÁCTICAS:** 2
CLASES POR SEMANA		**TEÓRICA:** 4	**PRÁCTICAS:** 1
HORAS POR SEMESTRE		**TEÓRICA:** 64	**PRÁCTICAS:** 32

Objetivos generales: Al final del curso el alumno:

- Conocerá y aplicará los métodos y técnicas para especificación, validación y verificación de *software.*
- Será introducido al concepto de tipo abstracto de dato (abstract *data type).*
- Estará capacitado para comprender los fundamentos relativos al concepto, la implantación y las aplicaciones de las estructuras de datos básicas - listas, arreglos, tablas, pilas o *stacks,* colas, árboles, gráficas - y los operadores asociados a ellas.
- Será introducido a la definición, implantación y aplicaciones de las estructuras de datos no lineales y los operadores asociados a ellas.
- Comparará distintos algoritmos para búsquedas y ordenamientos, con especial atención al balance entre complejidad y espacio contra tiempo.

Tema 1. Especificación, verificación y validación **8 horas teóricas**
4 horas teórico-prácticas

Explicará los conceptos y aplicaciones de la especificación, verificación y validación.

1.1 Tipos de especificaciones: especificaciones operativas y descriptivas, construcción y uso.
1.2 Verificación. Metas, enfoques y pruebas; análisis; ejecución simbólica; depuración.
1.3 Lectura de código y diseño y recorridos estructurados.

Tema 2. Tipos abstractos de datos **8 horas teóricas**
4 horas teórico-prácticas

Reconocerá el concepto de tipo abstracto de dato y sus implicaciones prácticas.

2.1 Conceptos involucrados.
2.2 Instrumentación de TAD (ADT) en un lenguaje de alto nivel, con ejemplos.

Tema 3. Estructuras de datos básicas **20 horas teóricas**
10 horas teórico-prácticas

Comprenderá los principales conceptos y operadores relacionados con las estructuras de datos.

3.1 Definición de las estructuras de datos lineales. Estructuras secuenciales. Arreglos empacados.
3.2 Uso de las estructuras de datos básicas.
3.3 Implementaciones contiguas y ligadas, ajuste para que respondan al problema planteado, incluyendo contraposición entre tiempo y espacio.
3.4 Estructuras múltiples. Búsqueda, inserción y remoción de elementos. Listas circulares y bidireccionales. Listas múltiples.

Tema 4. Estructuras de datos no lineales **12 horas teóricas**
6 horas teórico-prácticas

Explicará la naturaleza de las estructuras de datos no lineales y la importancia de éstas en la computación y las matemáticas.

4.1 Presentación de las estructuras no lineales. Árboles y estructuras arborescentes. Árboles binarios. Representación de árboles arbitrarios en base a árboles binarios.
4.2 Listas y recolección de basura. Asignación dinámica de espacio.

Tema 5. Búsqueda y ordenamientos **16 horas teóricas**
8 horas teórico-prácticas

Comparará diferentes algoritmos para búsquedas y ordenamientos y sus implicaciones.

5.1 Algoritmos de ordenamiento de orden *nlong* (*quicksort, heapsort, mergesort*), complejidad entre el tiempo y el espacio: mejores y peores casos.
5.2. Otros algoritmos de ordenamiento (algoritmo de Shell, de cubetas y de radicales).
5.3 Comparación de algoritmos.
5.4 Funciones de dispersión y resolución de colisiones.
5.5 Algoritmos de búsqueda y manejo de árboles balanceados; árboles B y AVL.
5.6 Algoritmo de ordenamientos externos.

Bibliografía básica:

- Tucker, B. *et al. Fundamental of Computing, I: Logic, Problems Solving, Programs and Computers.* 2nd edition. USA. McGraw-Hill. 1994.
- Magidin, M. *Estructuras de datos.* México. Editorial Trillas. 1991.

Bibliografía complementaria:

- Aho, A.V. *et al. Estructuras de datos y algoritmos.* México. Addison Wesley Publishing Company, 1988.
- Base, S. *Computer Algorithms. Introducing to Design and Analysis.* USA. Addison Wesley Publishing Company. 1990.
- Cormen, T.H. *et al. Introduction to Algorithms.* USA. McGraw-Hill Book Company. 1990.
- Biggerstaff, T.J., Perles A. J.; edited by Software Reusability, Volume I, II, ACM PRESS Addison Wesley Publishing Company, 1989.
- Ghezzy,C. *et al. Fundamentals of Software Engineering.* USA. Prentice-Hall Inc. 1991.
- Glass, R. L. *Software conflict. Essays on the Art and Science of Software Engineering.* USA. Yourdon Press Computing series. 1991.
- Gehani, N., McGettrick, A.D. editors. *Software Specification Techniques.* USA. Addison Wesley Publishing Company. 1980.
- Gries, D. editor. *Programming in the 1990s. An Introduction to the Calculation of Programs.* Springer Verlag, 1990.
- Dijkstra, W. E. editor. *Formal Development of Programs and Proofs.* USA. Addison Wesley Publishing Company. 1990.

Sugerencias didácticas:

Se recomiendan:
Tareas semanales en las cuales el alumno aplique el material visto en clase.
Trabajos de investigación bibliográfica para que el alumno amplíe sus conocimientos y conozca diferentes enfoques del material del curso.
Prácticas de cómputo para la experimentación con los algoritmos vistos en clase.

Análisis y resolución de casos prácticos.

Forma de evaluación:

Se recomiendan de 3 a 4 exámenes parciales y un examen final, así como la realización de tareas sobre los temas vistos en clase para reforzar los conocimientos teóricos adquiridos.

Perfil profesiográfico:

Egresado preferentemente de las licenciaturas en Ciencias de la Computación, Ingeniería en Sistemas, Matemáticas, Actuaría o alguna afín con experiencia docente y profesional en el área.

B.3. Bases de Datos (véase [4])

Conocerá y aplicará los principios fundamentales acerca del diseño y utilización de bases de datos.

BASES DE DATOS

CLAVE:		**SECTOR:**	OPTATIVO
SEMESTRE:	6 - 8	**ÁREA:**	INFORMÁTICA
CRÉDITOS:	10	**SERIACIÓN:**	
		ASIGNATURA PRECEDENTE INDICATIVA: Materias del sector básico del Área de Informática	
		ASIGNATURA SUBSECUENTE INDICATIVA: Ninguna	
HORAS POR CLASE		**TEÓRICA:** 1	**PRÁCTICAS:** 2
CLASES POR SEMANA		**TEÓRICA:** 4	**PRÁCTICAS:** 1
HORAS POR SEMESTRE		**TEÓRICA:** 64	**PRÁCTICAS:** 32

Objetivos generales: Al finalizar el curso el alumno:

- Conocerá y aplicará los principios fundamentales acerca del diseño y utilización de bases de datos.

Tema 1. Conceptos básicos de Bases de Datos **12 horas teóricas**

6 horas prácticas

Explicará lo que son los sistemas de bases de datos y las herramientas existentes para manejarlos.

1.1 Revisión de conceptos y terminología básica.
1.2 Arquitectura de un sistema de base de datos.
1.3 Modelos de datos.

Tema 2. El modelo relacional **12 horas teóricas**

6 horas prácticas

Comprenderá y aplicará los conceptos fundamentales relacionados con este modelo.

2.1 Estructura. Dominios y relaciones.
2.2 Integridad.
2.3 Álgebra y Cálculo relacionales.
2.4 Lenguajes de consulta relacionales.

Tema 3. Teoría de diseño para bases de datos relacionales **12 horas teóricas**

6 horas prácticas

Conocerá los fundamentos sobre los que descansa el diseño de bases de datos relacionales.

3.1 ¿Qué constituye un mal diseño de una base de datos?
3.2 Dependencias funcionales.
3.3 Descomposición sin pérdida.
3.4 Formas normales (primera, segunda, tercera, BCNF, cuarta y quinta) y los conceptos asociados a las mismas.

Tema 4. Bases de datos orientadas a objetos **12 horas teóricas**

6 horas prácticas

Identificará los principios relativos al diseño y funcionamiento de este tipo de bases.

4.1 Conceptos básicos.
4.2 Modelos de bases de datos orientadas a objetos.
4.3 Consideraciones en cuanto a lenguajes, vistas, autorización, consultas para bases de datos orientadas a objetos.
4.4 El paradigma orientado a objetos-relacional.

Tema 5. Protección de la base de datos **12 horas teóricas**

6 horas prácticas

Analizará los diversos métodos que existen para garantizar la seguridad de una base de datos.

5.1 Recuperación.
5.2 Concurrencia.
5.3 Seguridad.
5.4 Integridad.

Tema 6. Las bases de datos y las organizaciones **4 horas teóricas**

2 horas prácticas

Explicará la relevancia de las bases de datos en el esquema de las organizaciones.

Bibliografía básica:

- Ullman, J. D. *Principles of Database and Knowledge-base Systems, vol I.* (s. l.) Computer Science Press. 1988.

Bibliografía complementaria:

- Alagic. *Object-Oriented Database Programming.* (s. l.) Springer-Verlag. 1988.
- Date, C. J. *An Introduction to Database Systems, vol I (Data Bases),* 3nd edition, (s. l.) Addison Wesley Publishing Company. 1982.
- Filman, R. E. and D. P. Friedman. *Coordinated Computing: Tools and Techniques for Distributed Software.* (s. l.) McGraw-Hill Inc. 1984. (Computer Science).
- Kershenbaum, A. *Telecommunications Network Design Algorithms.* (s. l.) McGraw-Hill Inc. 1991. (Computer Science).
- Korth, H. and A. Silbersschatz. *Database Sistems Concepts.* 2nd edition. (s. l.) McGraw-Hill Inc. 1991. (Computer Science).
- Pascal, F. *Understanding Relational Databases. with Examples in SQL.* (s. l.) Wiley & Sons. 1993.
- Robertazzi, T. G. *Computer Networks and Systems, Queuing Theory and performance Evaluation.* (s. l.) McGraw-Hill, 1990. (Computer Science).
- Wiederhold, G. *File Organization for Database Design.* (s. l.) McGraw-Hill. 1987. (Computer Science).

Sugerencias didácticas:

Se recomiendan tareas regulares en las cuales el alumno aplique el material visto en clase y esté obligado a revisar diversas fuentes bibliográficas para que amplíe sus conocimientos con diferentes enfoques.
Asimismo se sugieren prácticas de cómputo para la experimentación con los algoritmos vistos en clase y el análisis de casos prácticos.

Forma de evaluación:

Se recomiendan de 3 a 4 exámenes parciales y un examen final, así como la realización de tareas sobre los temas vistos en clase para reforzar los conocimientos teóricos adquiridos.

Perfil profesiográfico:

El profesor que imparta el curso deberá ser egresado de las carreras de Actuaría, Matemáticas o alguna afín, de preferencia tener un posgrado, y deberá tener experiencia docente en el área o en las aplicaciones de diseño y utilización de bases de datos.

BIBLIOGRAFÍA

[1] Facultad de Ciencias, Administración, página Web,
http://www.fciencias.unam.mx/nosotros/organizacion/Index

[2] Facultad de Ciencias, Organización, página Web,
http://www.fciencias.unam.mx/nosotros/organizacion/academica

[3] Facultad de Ciencias, Posgrado, página Web,
http://www.fciencias.unam.mx/nosotros/organizacion/resumen/7

[4] Plan de estudio Actuaria (anteriores vigentes 2000, 2006) actual 2015,
http://www.fciencias.unam.mx/licenciatura/resumen/101/2017

[5] Plan de estudio Ciencias de la Computación (anterior vigente 1994) actual 2013,
http://www.fciencias.unam.mx/licenciatura/resumen/104

[6] Plan de estudio Matemáticas actual 1983,
http://www.fciencias.unam.mx/licenciatura/resumen/122

[7] Plan de estudio Matemáticas Aplicadas actual 2017,
http://www.fciencias.unam.mx/licenciatura/resumen/136

[8] Molina, José Manuel; Chamorro, Félix. CEO. Programación en lenguajes estructurados. Ed. McGraw-Hill, ISBN: 84-481-4870-3.

[9] Kernighan, Brian ; Ritchie, Dennis . The C Programming Language. USA: Prentice Hall, 1978. ISBN 0-13-110362-8.

[10] Booch, Grady. Object-oriented analysis and design with applications (2nd ed.). Santa Clara, CA, USA: Addison Wesley, 1994. ISBN 0-8053-5340-2.

[11] Ceballos Sierra, Fco. Javier. El lenguaje de programación C#. Alfaomega, Ra-Ma, 2002. ISBN 84-7897-500-4.

[12] Silberschatz, Abraham; Korth, Henry F., S. Sudarshan. Fundamentos de bases de datos. España: McGraw-Hill, 2002, 771 pp.787.

[13] Joyanes Aguilar, Luis; Fernández Azuela, Matilde. C# Manual de programación. McGraw-Hill Profesional, 2002. ISBN 84-481-3624-1.

[14] Programación Lineal, página Web,
http://recursostic.educacion.es/descartes/web/materiales_didacticos/Programacion_lineal/

[15] Hernández Ayuso, María del Carmen. Introducción a la programación lineal. México, UNAM, Prensas de ciencias, 2007, 215 pp. 219.

[16] Archer, Tom. A fondo C#. McGraw-Hill, 2001. ISBN 978844813460.

[17] Fowler, Martin; Kendall, Scott. UML GOTA A GOTA. Addison Wesley Longman de México, 1999. ISBN 968-444-364-1.

[18] Windows, página Web,
https://www.microsoft.com/es-mx/windows/

[19] Stallman, Richard M. Software libre para una sociedad libre. España: Traficantes de Sueños, 2004. pp.317.

[20] Kay, Trevor. Linux+ Certification Bible (2002). NY: Hungry Minds Inc. ISBN: 0-7645-4881-6.

[21] Gilat, Amos. Matlab: una introducción con ejemplos prácticos. Ed. Reverté. ISBN-10: 84-291-5035-8.

[22] MatLab, MathWorks, página Web,
http://www.mathworks.com/products/MatLab/

[23] The Fortran Company, página Web,
http://www.fortran.com/

[24] LINPACK, página Web,
http://www.netlib.org/linpack/

[25] EISPACK, página Web,
http://www.netlib.org/eispack/

[26] CUDA, página Web,
http://www.nvidia.com/object/cuda_home_new.html

[27] NVIDIA, página Web,
http://www.nvidia.com.mx/page/home.html

[28] SIMULINK Simulation and Model-Based Design, página Web,
http://www.mathworks.com/products/simulink/

[29] Koranne, Sandeep. Handbook of Open Source Tools. Springer. ISBN: 978-1-4419-7718-2.

[30] OCTAVE, GNU Octave, página Web,
http://www.gnu.org/Software/octave/

[31] Diferentes Tipos de Licencias para el Software, página Web,
http://www.gnu.org/licenses/license-list.html

[32] FreeMat, FreeMat Opén Source for rapid engineering and scientific prototyping and data processing, página Web,
http://freemat.sourceforget.net/

[33] MPI, página Web,
https://computing.llnl.gov/tutorials/mpi/

[34] SCILAB, Scilab Open Source for Numerical Computation, página Web,
http://www.scilab.org/

[35] Scicos Block Diagram Modeler/Simulator, página Web,
http://www.scicos.org/

[36] SCIPY Open Source Library of Scientific Tools, página Web,
http://www.scipy.org/

[37] Norton, Peter; Samuel, Alex; Foster-Johnson, Eric; Richardson, Leonard; Diamond, Jason; Parker, Aleathea; Roberts, Michael. Beginning Python. Canada: Wiley Publishing, Inc. ISBN-10: 0-7645-9654-3.

[38] Python Programming Language, página Web,
http://www.python.org/

[39] The R Proyect for Statistical Computing, página Web,

http://www.r-project.org/

[40] Maple, página Web,
http://www.maplesoft.com/

[41] Mathematica, página Web,
http://www.wolfram.com/mathematica/

[42] Maxima, página Web,
http://maxima.sourceforge.net

[43] Eitel, D; J., Paul; Deitel, Harvey M. Cómo Programar En Java. Séptima edición. México: Pearson Educación, 2008. ISBN: 978-970-26-1190-5.

[44] Java, página Web,
https://www.java.com

[45] KOctave, página Web,
http://sourceforge.net/projects/koctave/

[46] QTOctave, página Web,
http://www.ohloh.net/p/qtoctave

[47] IDLE is the Python IDE, página Web,
http://docs.python.org/2/library/idle.html

[48] IPython Interactive Computing, página Web,
http://ipython.org/

[49] Appcelerator PyDEV, página Web,
http://pydev.org/

[50] Eclipse, página Web,
http://www.eclipse.org/

[51] The Eric Python IDE, página Web,
http://eric-ide.python-projects.org/

[52] MatLab to Scilab conversion tips, página Web,
https://www.scilab.org/product/man/section_cd3c3e3c2fbd0d1fa2f8f04052b20b91.html

[53] IBM SPSS Software, página Web,
https://www-01.ibm.com/software/mx/analytics/spss/

[54] SAS, Business Analytics and Business Intelligence Software, página Web,
http://www.sas.com/

[55] GNU PSPP, página Web,
http://www.gnu.org/Software/pspp/

[56] EViews, página Web,
http://www.eviews.com/home.html

[57] Gretel, página Web,
http://gretl.sourceforge.net/

[58] Stata, página Web,
http://www.stata.com/

[59] Statgraphics, página Web,
http://statgraphics.softonic.com/

[60] Statistica, página Web,
http://www.statsoft.com/support/download/statistica-Software-updates/

[61] Systat, página Web,

http://www.systat.com/

[62] Vensim, página Web,
http://vensim.com/vensim-Software/

[63] PSPPIRE Data Editor for PSPP, página Web,
http://www.softpedia.com/get/Office-tools/Other-Office-Tools/PSPP.shtml

[64] Kopka, Helmut; W. Daly Patrick. A Guide to LATEX (Tools and Techniques for Computer Typesetting). USA: Pearson Education . ISBN-10: 0321173856.

[65] LaTeX, A Document Preparation System, página Web,
http://www.latex-project.org/

[66] RWeka, página Web,
http://cran.r-project.org/web/packages/RWeka/index.html

[67] Tinn-R Edit code and run it in R, página Web,
https://sourceforge.net/projects/tinn-r/

[68] RStudio Software, Education, and Services for the R community, página Web,
http://www.rstudio.com/

[69] Microsoft Office, página Web,
http://office.microsoft.com/

[70] LibreOffice the Document Foundation, página Web,
http://www.libreoffice.org

[71] Grant, Rickford. Linux for Non-Geeks: A Hands-On, Project-Based, Take-It-Slow Guidebook. No Starch Press. ISBN 10: 1593270348.

[72] OPEN OFFICE, Apache OpenOffice, página Web,
http://www.openoffice.org

[73] CALLIGRA The Integrated Work Applications Suit, páginaWeb,
http://www.calligra.org/

[74] Google Docs, página Web,
http://docs.google.com/

[75] Lotus Sympony, página Web,
https://www.ibm.com/developerworks/ssa/downloads/ls/symphony/learn/

[76] Mathtype, página Web,
http://www.dessci.com/en/products/mathtype/

[77] Scientific WorkPlace, página Web,
http://www.mackichan.com/

[78] Gummi LaTeX Editor, página Web,
http://dev.midnightcoding.org/projects/gummi

[79] Kile LaTeX Editor, página Web,
http://kile.sourceforge.net/

[80] Led LaTex Editor, página Web,
http://www.latexeditor.org/

[81] Lyx LaTeX Editor, página Web,
http://www.lyx.org/

[82] Texmaker LaTeX Editor, página Web,
http://www.xm1math.net/texmaker/

[83] TeXnicCenter LaTeX Editor, página Web,
http://www.texniccenter.org/

[84] TextPad LaTeX Editor, página Web,
http://www.textpad.com/

[85] TeXstudio LaTeX Editor, página Web,
http://texstudio.sourceforge.net/

[86] WinEdt LaTeX Editor, página Web,
http://www.winedt.com/

[87] Repositorio de LaTeX en la Facultad de Ciencias, UNAM, página Web,
http://tezcatl.fciencias.unam.mx/tex-archive/

[88] Medina Serrano, Santiago. SQL Server 2014. Soluciones Prácticas de Administración. RA-MA, 2015. ISBN 9788499645179.

[89] Microsoft SQL Server, página Web,
http://www.microsoft.com/en-us/sqlserver/default.aspx

[90] PostgreSQL, página Web,
http://www.postgresql.org/

[91] MySQL Oracle, página Web,
http://www.mysql.com/

[92] MongoDB, página Web,
http://www.mongodb.org/

[93] LaTeX Beamer Class, página Web,
https://bitbucket.org/rivanvx/beamer/wiki/Home

[94] Martin, Jim (Editor). Linux: The Complete Manual Management. Magbook. MagBooks Manager. ISBN 1-907779-19-1.

[95] Hertzog, Raphäel; Mas, Roland. El libro del administrador de Debian. [s.n.], 2012. ISBN: 979-10-91414-00-5 (papel) ISBN: 979-10-91414-01-2 (electrónico).

[96] Negus, Christopher. Linux® Bible 2008 Edition: Boot Up to Ubuntu®, Fedora®, KNOPPIX, Debian®, openSUSE®, and 11 Other Distributions. Wiley Publishing, Inc. ISBN: 978-0-470-23019-0.

[97] Negus, Christopher. Linux ® Bible 2010 Edition Boot Up to Ubuntu ® , Fedora ® , KNOPPIX, Debian® , openSUSE ® , and 13 Other Distributions. Wiley Publishing, Inc. ISBN: 978-0-470-48505-7.

[98] Andrew S., Tanenbaum; Steen, Maarten Van. Sistemas operativos modernos. Tercera edición. México: Pearson Educación , 2009. ISBN: 978-607-442-046-3.

[99] Figgins, Stephen; Siever, Ellen; Weber, Aaron. Linux in a Nutshell, 4th Edition. O'Reilly, June 2003. ISBN: 0-596-00482-6.

[100] Andrés Rosique. Clonación automática de equipos con FOG Linux. Revistalinux.net, LINUX+DVD 02/2010 (62) http://lpmagazine.org/es.

[101] Speake, Graham; Barber, Brian; Happel, Chris; Lillard, Terrence. Eleventh Hour Linux+ 1st Edition Exam XK0-003 Study Guide. Syngress, 16th November 2009. eBook ISBN: 9781597494984. Paperback ISBN: 9781597494977.

[102] Nitesh Dhanjani Hacknotes - Linux And Unix Security Portable Reference (2003). McGraw-Hill/Osborne, 2003. ISBN - 0-07-222786-9.

[103] Raya Cabrera, Jose Luis. Máquinas virtuales. Guía de Campo. RA-MA, 2010. ISBN: 9788478979493 329.

[104] FSF, Free Software Fundation, página Web,
http://www.fsf.org/

[105] GNU Operating System, página Web,
http://www.gnu.org/

[106] Software Libre, página Web,
http://es.wikipedia.org/wiki/Software_libre

[107] Software Libre, página Web,
http://www.hispalinux.es/SoftwareLibre

[108] ¿Qué es Software Libre?, página Web,
http://www.gnu.org/philosophy/free-sw.es.html

[109] El economista, página Web,
http://eleconomista.com.mx/tecnociencia/2013/01/22/clausuraran-negocios-mexico-uso-ilegal-Software

[110] PCworld, página Web,
http://www.pcworld.com.mx/Articulos/4627.htm

[111] Máquinas Virtuales, página Web,
https://es.wikipedia.org/wiki/Máquina_virtual

[112] Algunos usos de máquinas Virtuales, página Web,
http://www.configurarequipos.com/doc747.html

[113] KVM, página Web,
http://www.linux-kvm.org/page/Main_Page

[114] QEMU, página Web,
http://wiki.qemu.org/Main_Page

[115] Romero, Alfonso. Virtualbox 3.1. Packt Publishing (15 de abril de 2010. ISBN-10: 1847199143 ISBN-13: 978-1847199140.

[116] VirtualBox, página Web,
https://www.virtualbox.org

[117] VirtualBox – Free VirtualBox® Images, página Web,
https://virtualboxes.org/images/

[118] VirtualBox – Free VirtualBox® Images, LUBUNTU, página Web,
https://virtualboxes.org/images/lubuntu/

[119] VirtualBox – Free VirtualBox® Images, LUBUNTU 12.10, página Web,
http://sourceforge.net/projects/virtualboximage/files/Lubuntu/12.10/lubuntu1210.7z/download