DIES NATALIS XVII

FAKULTAS TEKNOLOGI INFORMASI DAN SAINS
UNIVERSITAS KATOLIK PARAHYANGAN

Bandung, Selasa 20 April 2010

Fakultas Teknologi Informasi dan Sains
UNIVERSITAS KATOLIK PARAHYANGAN
BANDUNG

DIES NATALIS XVII

FAKULTAS TEKNOLOGI INFORMASI DAN SAINS
UNIVERSITAS KATOLIK PARAHYANGAN

Bandung, Selasa 20 April 2010

Fakultas Teknologi Informasi dan Sains
UNIVERSITAS KATOLIK PARAHYANGAN
BANDUNG

Pembelajaran Pemrograman di Perguruan Tinggi: Masalah dan Tantangannya

Cecilia E. Nugraheni

Jurusan Teknik Informatika
Fakultas Teknologi Informasi dan Sains
Universitas Katolik Parahyangan

"... as long as there were no machines, programming was no problem at all; when we had a few weak computers, programming became a mild problem, now we have gigantic computers, programming has become an equally gigantic problem."

"The Humble Programmer"- Edsger Dijkstra, 1972

I. Pendahuluan

Tuntutan pasar yang begitu besar akan sumberdaya manusia yang mempunyai kompetensi tinggi di bidang Teknologi Informasi dan Komunikasi (TIK) mendorong banyak perguruan tinggi untuk mendirikan program studi yang bertujuan untuk menghasilkan lulusan dengan kompetensi tersebut. Program studi seperti ini dapat muncul dengan berbagai nama dan strata. Sejumlah nama program studi yang umum digunakan, khususnya di Indonesia, adalah Informatika, Teknik Informatika, Ilmu Komputer, Teknologi Informasi, Teknologi Informasi dan Komunikasi, Sistem Informasi, Sistem Komputer, dsb.

Apapun nama yang dipilih, pada intinya program-program studi tersebut tetap saja dapat dikembalikan pada pengertian dasar dari program studi Informatika yaitu bidang ilmu yang mempelajari bagaimana cara memecahkan permasalahan manusia dengan bantuan komputer. Antara permasalahan dan komputer tentu saja ada jarak yang besar. Agar dapat diselesaikan oleh komputer, permasalahan tersebut harus dianalisis, diformulasikan dengan cara tertentu, dan kemudian dinyatakan dalam bentuk program yaitu serangkaian perintah/instruksi yang dapat dilaksanakan oleh

komputer. Kegiatan dalam menghasilkan sebuah program inilah yang biasa dikenal dengan istilah pemrograman.

Oleh karena itu, tidaklah heran jika pemrograman merupakan kompetensi dasar yang harus dimiliki oleh lulusan program studi Teknik Informatika dan sejenis [1]. Hanya pada kenyataannya tidaklah mudah mengajar maupun belajar pemrograman ini. Mata kuliah pemrograman biasanya dianggap sulit, dan sering banyak yang tidak lulus, bahkan disebut sebagai penyebab tingginya tingkat *dropout*. Ketidaklulusan dan nilai D pada mata kuliah pengantar pemrograman di banyak universitas di AS, Kanada, dan tempat-tempat lain sudah mendekati 50 persen [2]. Situasi seperti ini juga dialami oleh Jurusan Teknik Informatika Unpar dimana tingkat ketidaklulusan mata kuliah pemrograman mencapai 30% pada tahun 2006 dan tingkat *dropout* adalah 20% (seperlima mahasiswa dalam setiap angkatan tidak berhasil menyelesaikan studinya) [3]. Setelah dipelajari lebih lanjut ternyata memang ada korelasi antara tingkat ketidaklulusan mata kuliah pemrograman dengan tingkat *dropout*. Sebagian besar mahasiswa yang gagal studinya adalah mahasiswa yang tidak lulus kuliah pemrograman.

Isu-isu yang berkaitan dengan pembelajaran pemrograman sudah menjadi bahan diskusi sejak munculnya pendidikan informatika atau ilmu komputer [4,5]. Isu-isu tersebut meliputi bagaimana cara terbaik untuk memberikan konsep dasar pemrograman pada mata kuliah dasar pemrograman, pemilihan bahasa pemrograman, materi apa yang harus diajarkan, bagaimana cara mengajar, dan bagaimana menyeimbangkan antara pengetahuan sintaks, kemampuan merancang, dan kreativitas.

Makalah ini membahas isu-isu yang terkait dengan pembelajaran pemrograman di perguruan tinggi, khususnya pada program studi teknik informatika dan sejenisnya. Pembelajaran pemrograman di Jurusan Teknik Informatika Unpar akan diberikan sebagai salah satu studi kasus.

II. Pemrograman, paradigma pemrograman, dan bahasa pemrograman

Terdapat beberapa istilah penting yang berkaitan dengan pemrograman, yaitu program, paradigma pemrograman, bahasa pemrograman, dan algoritma. Berikut ini akan diberikan penjelasan sedikit tentang masing-masing istilah tersebut. Sebagian penjelasan ini diambil dari [6,7,8].

Sebuah program komputer adalah serangkaian instruksi yang dijalankan oleh komputer untuk memecahkan permasalahan tertentu. Sebuah program ditulis dalam suatu bahasa pemrograman dan diinterpretasikan satu per satu oleh penerjemah

bahasa kedalam bahasa mesin dan dikonversikan menjadi energi listrik yang menyebabkan komputer bekerja. Kegiatan yang dilakukan dalam rangka menghasilkan sebuah program disebut sebagai pemrograman.

Algoritma adalah serangkaian langkah-langkah yang tepat, terperinci, dan terbatas untuk menyelesaikan suatu permasalahan. Langkah yang tepat artinya rangkaian langkah-langkah tersebut selalu benar untuk menyelesaikan masalah yang diberikan. Terperinci artinya setiap langkah diberikan secara detail dan dapat dieksekusi oleh komputer. Sedangkan langkah terbatas artinya suatu saat langkah tersebut harus berhenti.

Paradigma pemrograman adalah bagaimana cara pandang kita terhadap penyelesaian masalah pemrograman. Setiap bahasa pemrograman pasti didasarkan pada minimal satu paradigma pemrograman ini. Saat ini terdapat banyak paradigma pemrograman. Empat paradigma pemrograman yang terkenal adalah:

1. Paradigma pemrograman prosedural atau imperatif
   Paradigma ini didasari oleh konsep mesin von Neumann (*stored program concept*): sekelompok tempat penyimpanan (memori) yang dibedakan menjadi memori instruksi dan memori data; masing-masing dapat diberi nama dan harga. Instruksi akan dieksekusi satu per satu secara sekuensial oleh sebuah pemroses tunggal. Beberapa instruksi menentukan instruksi berikutnya yang akan dieksekusi (percabangan kondisional). Data diperiksa dan dimodifikasi secara sekuensial pula. Program dalam paradigma ini didasari pada struktur informasi di dalam memori dan manipulasi dari informasi yang disimpan tersebut. Kata kunci yang sering didengungkan dalam pendekatan ini adalah:

   Algoritma + Struktur Data = Program.

   Keuntungan pemrograman dengan paradigma ini adalah efisiensi eksekusi, karena dekat dengan mesin. Sedangkan kelemahannya adalah pemrograman menjadi sangat tidak "manusiawi" dan tidak alamiah, karena kita harus berpikir dalam batasan mesin (komputer). Contoh bahasa pemrograman yang bersifat prosedural adalah Pascal, Fortran, dan C.

2. Paradigma pemrograman fungsional
   Paradigma ini didasarkan pada konsep pemetaan dan fungsi pada matematika. Fungsi dapat berbentuk sebagai fungsi primitif atau komposisi dari fungsi-fungsi lain yang telah terdefinisi. Diasumsikan bahwa ada fungsi-fungsi dasar yang dapat dilakukan. Penyelesaian masalah didasari atas aplikasi dari fungsi-fungsi tersebut.

Paradigma fungsional ini tidak lagi mempermasalahkan memorisasi dan struktur data, tidak ada pemilahan antara data dan program, serta tidak ada lagi pengertian tentang variabel. Pemrogram tidak perlu lagi mengetahui bagaimana mesin mengeksekusi atau bagaimana informasi disimpan dalam memori, setiap fungsi adalah kotak hitam yang menjadi perhatiannya hanya keadaan awal dan akhir. Kelemahan dari program-program yang menggunakan pendekatan ini adalah waktu eksekusi yang relatif lebih tinggi dibandingkan dengan pendekatan prosedural. Contoh bahasa pemrograman yang bersifat fungsional adalah LISP.

3. Paradigma pemrograman deklaratif, predikatif, atau logik
   Paradigma ini didasari oleh pendefinisian relasi antar individu yang dinyatakan sebagai predikat. Sebuah program logik adalah kumpulan aksioma (fakta dan aturan deduksi). Pemrogram menguraikan sekumpulan fakta dan aturan-aturan (*inference rules*). Ketika program dieksekusi, pemakai mengajukan pertanyaan (*query*) dan program akan menjawab apakah pertanyaan itu dapat dideduksi dari aturan dan fakta yang ada. Program akan memakai aturan deduksi dan mencocokkan pertanyaan dengan fakta-fakta yang ada untuk menjawab pertanyaan. Bahasa pemrograman yang termasuk dalam kelompok bahasa pemrograman logika adalah Prolog.

4. Paradigma pemrograman berorientasi objek
   Paradigma ini didasari oleh kelas dan objek. Objek adalah instansiasi dari kelas. Objek mempunyai atribut (kumpulan sifat), dan mempunyai kelakuan (kumpulan metode). Objek yang satu dapat berkomunikasi dengan objek yang lain lewat pesan, dengan tetap terjaga integritasnya. Kelas mempunyai hierarki, anggota dari sebuah kelas juga mendapatkan turunan atribut dari kelas di atasnya. Paradigma ini menawarkan konsep modularitas, penggunaan kembali, dan kemudahan modifikasi. Paradigma ini masih mengandung paradigma imperatif karena mengkonstruksi program dari objek dan kelas adalah tidak berbeda dengan mengkonstruksi program dari struktur data dan algoritma, dengan cara enkapsulasi menjadi kelas. Bahasa pemrograman Java adalah salah satu contoh bahasa pemrograman yang berorientasi objek.

III. Object first vs imperative first

Di antara empat paradigma pemrograman utama yang dijelaskan sebelumnya, dua paradigma yang paling umum yang diberikan pada kuliah pemrograman adalah

paradigma prosedural dan berorientasi objek. Walaupun diskusi tentang pendekatan mana yang lebih baik diberikan lebih dulu telah banyak dilakukan, namun sampai saat ini belum diperoleh kesepakatan. Hal ini dikarenakan masing-masing paradigma ini mempunyai kelebihan dan kekurangan.

Saat ini memang pendekatan pengajaran pemrograman lebih banyak dengan mengajarkan pemrograman berorientasi objek sebagai konsep pemrograman pertama atau yang disebut sebagai pendekatan *object first*. Alasan dari pemilihan pendekatan ini berangkat dari pernyataan bahwa pendekatan berorientasi objek disebut sebagai cara alami untuk mengkonseptualkan masalah-masalah dunia nyata (*"the natural way of conceptualizing real-world problems"*). Disebutkan bahwa sejak kecil manusia sudah menerapkan kerangka berpikir berorientasi objek ini. Manusia secara sadar atau tidak akan mengelompokkan atau mengklasifikasikan benda-benda yang ada di sekitarnya berdasarkan sifat, properti, atau atribut dari benda tersebut. Diharapkan bahwa dengan pendekatan ini mahasiswa akan lebih mudah mempelajari pemrograman dan untuk selanjutnya dapat mendidik mereka menjadi pemrogram yang handal. Alasan lain mengapa pendekatan *object first* ini diambil adalah untuk menghindari kemungkinan transfer negatif yang diakibatkan oleh proses perubahan dari pendekatan prosedural ke pendekatan objek yang akan mempersulit mahasiswa [9].

Selain itu pemilihan pendekatan *object first* ini juga banyak didorong oleh tuntutan dari pihak industri. Saat ini penggunaan bahasa Java semakin meluas di dunia industri dan sehingga banyak program studi Informatika dan sejenisnya memilih pendekatan *object first* dan mengajarkan bahasa Java sebagai dalam rangka meningkatkan daya saing lulusan mereka.

Tuntutan dunia industri ini menyebabkan diabaikannya paradigma pemrograman prosedural. Namun begitu masih ada beberapa perguruan tinggi yang menggunakan paradigma prosedural pada pengajaran dasar pemrograman yang disebut sebagai pendekatan *imperative first*. Alasan pemilihan pendekatan ini di antaranya adalah karena paradigma prosedural mempunyai keunggulan dalam hal kedekatannya dengan bahasa mesin sehingga lebih mudah bagi pemula untuk memahami cara kerja dari suatu program atau memahami konstruksi dasar dari pemrograman seperti penugasan (*assignment*), pengulangan, dan kondisional.

Pemilihan pendekatan *imperative first* bisa juga dilatarbelakangi oleh bahasa pemrograman yang akan diajarkan. Salah satu bahasa pemrograman prosedural

yang banyak diajarkan adalah bahasa Pascal. Bahasa Pascal ini pada saat awalnya ditujukan untuk pendidikan dan merupakan bahasa yang sederhana dan konsisten. Hal ini sangat kontras dengan bahasa Java yang memang dirancang untuk *professional use*, menggunakan notasi yang tidak selalu konsisten, dan mengandung pustaka yang terlalu kompleks bagi pemula [9].

Saat-saat ini mulai muncul pandangan bahwa tidak cukup hanya mengajarkan salah satu dari paradigma pemrograman tersebut pada mahasiswa di kuliah dasar pemrograman. Banyak pendapat yang mulai menyoroti kelemahan dari pendekatan *object first*. Pendekatan *object first* ternyata tidak sesukses yang diharapkan [10]. Bahkan ada yang secara ekstrim menyatakan bahwa paradigma berorientasi objek hanya menambahkan kompleksitas dari struktur kelas ke sebuah sistem prosedural [11]. Dari pengalaman banyak pengajar pemrograman, sebagian besar setuju bahwa paradigma berorientasi objek hanya dapat diberikan jika mahasiswa telah mempunyai basis pemrograman prosedural yang kuat [9].

Dari kelebihan dan kekurangan kedua paradigma di atas, diusulkan sebuah pendekatan yang disebut sebagai *"variable first"*[9]. Ide utama dari pendekatan ini adalah mengkombinasikan awal yang kuat dalam konstruksi dasar dengan paradigma berorientasi objek. Pendekatan ini disebut sebagai *"variable first"* untuk menekankan bahwa pendekatan ini mulai dengan bagian penting dari paradigma pemrograman imperatif yaitu variabel dan khususnya peran mereka dalam struktur kendali imperatif.

IV. Faktor penyebab kegagalan dalam kuliah pemrograman

Kegiatan pemrograman terdiri atas sedikitnya empat aktivitas, yaitu: (i) analisis masalah dan pembuatan spesifikasi, (ii) perancangan program yang meliputi organisasi data dan algoritma, (iii) penulisan program, dan (iv) pengeksekusian dan pengujian program. Sebagai akibatnya kuliah pemrograman dituntut juga untuk mencakup semua hal tersebut: belajar fitur bahasa, desain program, dan pemahaman program (*program comprehension*), menulis program (*coding*), dan menguji program (*testing* dan *debugging*). Bagi seorang yang baru mulai belajar memrogram, tentu saja akan sulit mempelajari semua hal itu dalam waktu yang bersamaan.

Belajar memrogram dianggap sebuah tugas yang berat bagi sebagian besar mahasiswa. Sebuah studi melaporkan bahwa sedikitnya 25% mahasiswa gagal lulus pada mata kuliah pemrograman di awal studi. Beberapa faktor penyebab kegagalan mereka adalah sbb. [3,11,12,13,14,15]:

1. Pengetahuan pemrograman yang kurang

   Mahasiswa baru yang memilih untuk mengikuti kuliah di jurusan Teknik Informatika dan sejenis tidak diwajibkan untuk memiliki kemampuan dasar komputer sebagai prasyarat dalam mengikuti pendidikannya. Sebagian mahasiswa mungkin telah mengikuti pelajaran komputer selama di sekolah menengah sedangkan yang lainnya bahkan tidak memiliki pelajaran tersebut. Masalah ini tetap muncul pada mhs yg sudah memiliki keterampilan atau sudah memahami penggunaan komputer. Hal ini disebabkan karena mereka memiliki pengalaman pemrograman yang sangat sedikit.

2. Abstraksi yang kurang

   Beberapa masalah lain yang timbul dalam mempelajari pemrograman adalah mahasiswa perlu membayangkan hal-hal yang abstrak yang tidak dapat mereka bayangkan di dunia nyata, misalnya bagaimana membayangkan variabel, tipe data, atau memori jika dihubungkan dengan dunia nyata.

   Selain itu, mahasiswa juga mengalami kesulitan memahami cara kerja dari sebuah program. Du Boulay menyatakan bahwa *"...it takes quite a long time to learn the relation between a program on the page and mechanism it describes"*[13].

3. Kesulitan dalam perancangan program

   Dari banyak hasil studi, dapat disimpulkan bahwa sumber kesulitan utama mahasiswa dalam kuliah pemrograman bukanlah pada saat mempelajari konsep dari sebuah paradigma dan juga pada saat mempelajari sintaks dari sebuah bahasa pemrograman, tetapi lebih pada perencanaan program. Terdapat perbedaan antara pengetahuan tentang pemrograman dengan strategi pemrograman. Seorang mahasiswa dapat belajar menjelaskan konsep pemrograman seperti, misalnya tentang tipe data pointer, tetapi masih sering salah dalam menggunakannya. Mahasiswa tahu sintaks dan semantik dari perintah-perintah bahasa pemrograman, tetapi mereka tidak tahu bagaimana menggabungkan fitur-fitur tersebut kedalam sebuah program yang valid.

4. Tidak mampu melihat hasil komputasi

   Bagi pemula membuat program yang bebas dari kesalahan sintaks saja sudah sulit dan memakan waktu. Setelah bebas dari kesalahan sintakspun kadang mahasiswa masih harus berhadapan dengan *"execution errors"*. Sebagai akibatnya, hasil dari program setelah dijalankan menjadi sesuatu yang sangat lama dicapai. Situasi semacam ini sering menimbulkan keputusasaan bagi siswa.

5. Kesulitan dengan kakas (*tools*) yang digunakan
   Seni pemrograman juga mencakup pengetahuan tentang kakas pemrograman. Saat ini hampir semua mata kuliah pengantar pemrograman berdasarkan pembelajaran sebuah bahasa *general-purpose* seperti Java. Pengalaman menunjukkan bahwa banyak mahasiswa tidak kesulitan dengan bahasanya itu sendiri, tetapi kakas yang mereka gunakan membuat belajar pemrograman menjadi sulit. Misalnya *compiler* sering memberikan pesan kesalahan yang tidak dimengerti. Faktor bahasa juga menjadi salah satu penyebab situasi ini. Meskipun fasilitas bantuan (*help*) tersedia, hanya sebagian kecil mahasiswa yang memanfaatkanya. Mahasiswa yang kemampuan bahasa Inggrisnya kurang cenderung untuk langsung bertanya kepada dosen pengajar tanpa membaca pesan kesalahan itu sendiri. Kebiasaan ini akan menimbulkan masalah pada saat mereka harus bekerja atau belajar secara mandiri.
6. Lemahnya konsep matematika dan logika
   Ada pendapat yang menyatakan bahwa ada faktor intelegensi secara umum dan matematika atau sains juga berpengaruh pada kesuksesan belajar pemrograman. Banyak institusi yang menekankan pada pembentukan logika, konsep matematika dan algoritma pada pengajaran mata kuliah dasar termasuk pemrograman, dan bukan bahasa atau teknologi tertentu.
7. Kurangnya motivasi
   Profesi pemrogram (*programmer*) dipandang oleh banyak mahasiswa, bahkan mahasiswa Teknik Informatika dan sejenis, sebagai sebuah pekerjaan yang membosankan [15]. Pandangan ini berasal dari gambaran bahwa seorang pemrogram duduk di depan komputer, menghabiskan waktu yang lama untuk menghasilkan keluaran atau memperbaiki kesalahan-kesalahan yang kelihatannya tidak terlalu penting.

V. Pendekatan pembelajaran pemrograman

Sebuah studi menyatakan bahwa rata-rata diperlukan waktu sekitar sepuluh tahun bagi pemrogram pemula untuk menjadi seorang yang dianggap pakar dalam pemrograman. Jika mahasiswa yang baru dianggap sebagai pemula maka akan sulit sekali bagi program studi Teknik Informatika dan sejenisnya untuk mendidiknya langsung menjadi seorang pakar. Pendekatan umum pada pengajaran pemrograman adalah memberikan dasar-dasar bahasa pemrograman pada mata kuliah pemrograman dasar dan kemudian mengarahkan mahasiswa kepada strategi yang efektif pada keseluruhan

proses pemrograman. Oleh karena itu, pembelajaran konsep dasar lebih ditekankan yang nantinya akan dikembangkan menuju ketrampilan lebih lanjut.

Terdapat banyak pertanyaan sehubungan dengan bagaimana mengajarkan pemrograman yang terbaik. Hal ini selalu menjadi masalah yang sulit bagi perguruan tinggi, khususnya bagi program studi teknik informatika dan sejenis. Berdasarkan hasil analisis terhadap masalah-masalah yang dihadapi oleh mahasiswa dalam pembelajaran pemrograman, telah diusulkan pendekatan-pendekatan baru dalam pengajaran pemrograman, seperti visualisasi, *game-based learning*, *problem-based learning*, mengajarkan bahasa yang multiparadigm, mengelompokkan mahasiswa berdasarkan tingkat atau kompetensi pemrograman mereka, dsb. Berikut ini akan dibahas dua pendekatan yaitu visualisasi dan *game-based learning*.

5.1 Visualisasi

Visualisasi telah lama digunakan dalam pengajaran ilmu komputer karena visualisasi ini dipandang bermanfaat untuk memahami dan mempelajari konsep abstrak dan kompleks dari bidang ini. Banyak studi membuktikan bahwa visualisasi membantu mahasiswa memahami dan mempelajari konsep pemrograman [11,14,16,17].

Termasuk dalam pendekatan visualisasi dalam pengajaran pemrograman adalah penggunaan diagram untuk merepresentasikan algoritma. Penggunaan notasi-notasi grafis standar seperti *flowchart* untuk merepresentasikan alur kendali program pernah populer pada tahun 80-an [18]. Notasi lain yang terkenal dan banyak digunakan sampai saat ini adalah *data flow diagram* [19].

Saat ini pendekatan visualisasi telah berkembang dengan cukup pesat. Visualisasi tidak hanya berupa diagram-diagram yang bersifat statis. Konsep pemrograman yang direpresentasikan sebagai objek-objek secara visual dan ditampikan dalam bentuk animasi akan lebih menarik minat mahasiswa sehingga mahasiswa akan merasa bahwa belajar pemrograman itu menyenangkan. Penelitian menunjukkan bahwa penggunaan grafik dan animasi sebagai kakas pembelajaran yang efektif [20].

Salah satu contoh kakas lingkungan pemrograman yang mendukung pembelajaran pemrograman yang menggunakan pendekatan visualisasi adalah Alice [21,22]. Alice dikembangkan oleh Carnegie Mellon University yang dirancang secara khusus untuk memperkenalkan konsep pemrograman bagi pemula. Alice mendukung pendekatan *object first* dengan konsep modern, dimana objek-objek yang disediakan seperti orang, binatang, dan objek-objek menarik lainnya dapat ditampilkan dalam bentuk 3 dimensi

dengan mudah. Komputasi ditampilkan melalui animasi sehingga mahasiswa dapat melihat hasil atau jalannya sebuah algoritma secara interaktif. Selain itu, keunggulan dari Alice bagi pemula adalah tidak diperlukannya pengetikan program sehingga mahasiswa tidak dipusingkan dengan kesalahan sintaks.

Meskipun mempunyai banyak keunggulan, pendekatan visual juga mempunyai kelemahan. Kelemahan utama yang banyak disorot adalah waktu yang diperlukan untuk membuat visualisasi, untuk meng-*install* dan mempelajari teknologi yang digunakan, dan tidak mudah untuk mengintegrasikan visualisasi dengan kuliah.

5.2 Pendekatan *game-based learning*[1]

Game-based learning adalah metode pembelajaran yang menggunakan aplikasi permainan/*game* yang telah dirancang khusus untuk membantu dalam proses pembelajaran. Dengan menggunakan *game-based learning* kita dapat memberikan stimulus pada tiga bagian penting dalam pembelajaran yaitu emosi, intelektual, dan psikomotorik. *Game-based learning* adalah salah satu metode pembelajaran yang dirasa cocok dengan kondisi dari generasi dijital sekarang ini karena tiga alasan berikut:

1. menciptakan lingkungan belajar yang menyenangkan dan meningkatkan motivasi belajar
2. kompetisi dan kerjasama tim dalam menyelesaikan misi yang ada dalam aplikasi *game* juga dapat menambahkan komponen motivasi pada siswa
3. umpan balik yang cepat dan spesifik memberikan kemudahan bagi siswa untuk memikirkan cara lain yang tepat menyelesaikan tugasnya

Walaupun begitu pendekatan ini juga dapat menimbulkan dampak buruk, antara lain:

1. mahasiswa menganggap bahwa ini hanya sekedar permainan
2. mahasiswa berprinsip bahwa jika kalah dalam *game* maka tinggal mengulang atau memulainya lagi dari awal
3. mahasiswa cenderung memainkan *game* tanpa menikmati alur yang sudah disiapkan oleh *game* tersebut

Jika situasi itu ditemui, maka meskipun mahasiswa merasa senang belajar pemrograman, mahasiswa cenderung untuk menjadi tidak serius dalam belajar tujuan sehingga tujuan pembelajaran sendiri akan tidak tercapai.

Salah satu *game* yang dapat digunakan untuk mendukung pengajaran pemrograman adalah Ceebot[22]. Ceebot merupakan konsep baru yang memperkenalkan dasar-

1) Bagian ini diambil dari [3]

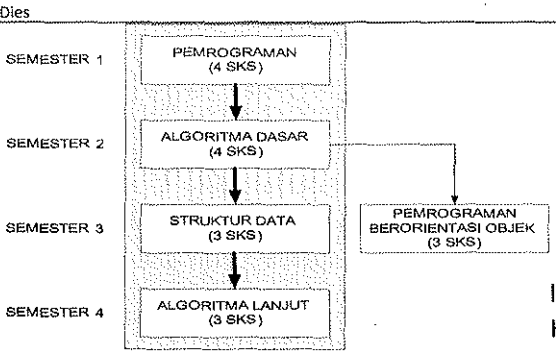
dasar pemrograman kepada penggunanya dengan cara yang menyenangkan. Dengan menggunakan lingkungan tiga dimensi yang cukup menarik, akan membuat pemakai atau dalam hal ini adalah mahasiswa dapat menikmati *game* ini layaknya *game-game* mutakhir. Ceebot memperkenalkan berbagai konsep modern yang dapat ditemukan pada lingkungan pemrograman yang sekarang populer yaitu C++, C#, atau Java. Ceebot menyediakan sejumlah misi-misi yang dirancang dalam bentuk *puzzle*. Dengan menyelesaikan misi-misi tersebut mahasiswa memperoleh gambaran tentang konsep-konsep pemrograman yang abstrak dan belajar tentang *problem solving*.

VI. Pengajaran pemrograman di jurusan Teknik Informatika Unpar

Pada saat pendiriannya, tahun 1996, Jurusan Teknik Informatika Unpar, atau Ilmu Komputer pada saat itu, banyak dibantu oleh dosen-dosen dari Jurusan Teknik Informatika ITB. Bantuan tersebut tidak hanya pada saat penyusunan kurikulum, melainkan juga sebagai tenaga pengajar pada saat perkuliahan sudah berjalan. Sebagai akibatnya, warna dari Jurusan Ilmu Komputer Unpar saat itu sangat-sangat mirip dengan Jurusan Teknik Informatika ITB. Salah satu cirinya adalah kuatnya atau mendominasinya mata kuliah-mata kuliah yang terkait dengan pemrograman, seperti dasar-dasar pemrograman, algoritma dan pemrograman, struktur data, struktur data lanjut, pemrograman logik, pemrograman fungsional, rekayasa perangkat lunak, dan pemrograman berskala besar.

Pada saat pergantian kurikulum, pada tahun 2003, meskipun perbedaan dengan Jurusan Teknik Informatika ITB mulai sedikit tampak, namun ciri tersebut tetap dipertahankan. Bahkan akhirnya ciri ini menjadi salah satu keunggulan lulusan Jurusan Teknik Informatika Unpar, yaitu mempunyai kompetensi tinggi dalam bidang algoritma dan pemrograman. Pengakuan akan hal ini telah datang dari beberapa pemakai lulusan, seperti OCBC NISP, Hewlett-Packard, Unpar (khususnya Biro Teknologi dan Informasi), Microsoft, dll.

Pengajaran pemrograman pada kurikulum 2003 menggunakan pendekatan *imperative first* dan bahasa pemrograman yang digunakan adalah bahasa C. Jalur mata kuliah pemrograman pada Kurikulum 2003 diberikan pada Gambar 1. Terdapat empat mata kuliah pemrograman utama, yaitu Pemrograman, Algoritma Dasar, Struktur Data, dan Algoritma Lanjut. Paradigma pemrograman berorientasi objek diperkenalkan pada mata kuliah Pemrograman Berorientasi Objek di semester ketiga dimana para mahasiswa diajarkan bahasa pemrograman Java. Sedangkan paradigma pemrograman yang lain yaitu fungsional dan logik ditawarkan sebagai mata kuliah pilihan.



Gambar 1. Jalur kuliah pemrograman pada Kurikulum 2003

Untuk mata kuliah Pemrograman digunakan tiga bentuk perkuliahan yaitu kuliah di kelas, praktikum di laboratorium, dan responsi di laboratorium. Urutan dari ketiga bentuk perkuliahan ini diatur sedemikian rupa dimana untuk setiap topik, materi pertama kali diberikan pada saat kuliah di kelas, berikutnya adalah praktikum, diikuti dengan responsi, dan diakhiri kuliah lagi di kelas. Pada kuliah muka di kelas materi dititikberatkan pada teori dan konsep dasar dari setiap topik. Setelah itu pada saat praktikum mahasiswa akan berlatih dengan soal-soal secara terbimbing. Praktikum lebih dititikberatkan pada *problem solving* untuk melihat sejauh mana mahasiswa dapat menggunakan teori dan konsep yang telah dipelajari di kelas untuk memecahkan permasalahan yang diberikan. Pada saat responsi, mahasiswa dituntut untuk mampu mengimplementasi teori dan konsep yang telah dipelajari dalam bentuk program. Selanjutnya, kuliah di kelas yang kedua berisi pembahasan tentang soal yang diberikan pada saat praktikum dan *review* tentang materi dari topik bahasan pada minggu tersebut.

Di satu sisi memang merupakan kebanggaan dan kepuasan tersendiri bagi jurusan karena dapat menghasilkan lulusan yang mempunyai kompetensi tinggi dalam bidang algoritma dan pemrograman sesuai dengan tujuan jurusan. Tapi sisi yang lain ternyata semakin lama kami mengamati bahwa tingkat ketidaklulusan mata kuliah pemrograman ini sangat tinggi. Pada tahun 2006 kami menghadapi masalah dengan tingginya tingkat *drop-out* yang mencapai 20% dan juga tingginya angka ketidaklulusan pada mata kuliah pemrograman yaitu sebesar 30%. Kami mengamati bahwa terdapat korelasi antara tingginya angka ketidaklulusan mata kuliah pemrograman dengan tinggi tingkat *dropout* tadi. Ternyata sebagian besar mahasiswa yang *dropout* adalah mahasiswa yang tidak lulus atau mengalami kesulitan pada mata kuliah pemrograman.

Dari analisis dapat disimpulkan bahwa para mahasiswa menemui masalah-masalah yang seperti yang telah dibahas di depan. Salah satu langkah yang diambil untuk

memperbaiki keadaan tersebut adalah dengan menerapkan pendekatan *game-based learning* dengan menggunakan CeeBot. Ceebot diperkenalkan pada awal kuliah agar mahasiswa dengan mudah dapat memahami konsep dasar pemrograman. Setelah beberapa saat menggunakan Ceebot mahasiswa diajarkan bahasa pemrograman C. Pendekatan ini ternyata membawa hasil yang menggembirakan yaitu persentase ketidaklulusan yang semula berkisar 37,64% berubah drastis menjadi 17,60% pada tahun 2007. Selain itu, dari survey dapat disimpulkan bahwa mahasiswa memberikan komentar positif dengan pendekatan *game-based learning* dan khususnya penggunaan Ceebot ini[2].

Di samping permasalahan umum yang dihadapi mahasiswa, hasil analisis menunjukkan bahwa struktur kurikulum 2003 juga berpengaruh terhadap tingginya tingkat *dropout* di Jurusan Teknik Informatika Unpar. Adanya prasyarat lulus yang diterapkan pada tiga kuliah pemrograman (garis tebal pada Gambar 1) ternyata menyebabkan banyaknya kegagalan mahasiswa melalui evaluasi tahap pertama. Mahasiswa yang memang mengalami kesulitan dalam pemrograman tingkat dasar akan cenderung tidak lulus dan sebagai konsekuensinya mereka harus mengulang mata kuliah ini di tahun berikutnya.

Bagi sebagian mahasiswa pengambilan ulang kuliah pemrograman memberikan keuntungan bagi mereka, karena mereka tinggal mengulang materi yang diberikan, sehingga pemahaman merekan terhadap materi menjadi lebih baik. Hanya ternyata hal ini tidak berlaku bagi semua mahasiswa yang mengulang. Pemahaman mereka pada pengambilan ulang tidak bertambah baik dan sebagai akibatnya mereka merasa tertekan dan menjadi putus asa. Solusi yang diambil saat itu adalah dengan membuka mata kuliah-mata kuliah tersebut di setiap semester, sehingga jika seorang mahasiswa tidak lulus salah satu dari mata kuliah tersebut, maka mahasiswa tidak harus menunggu satu tahun untuk mengulang.

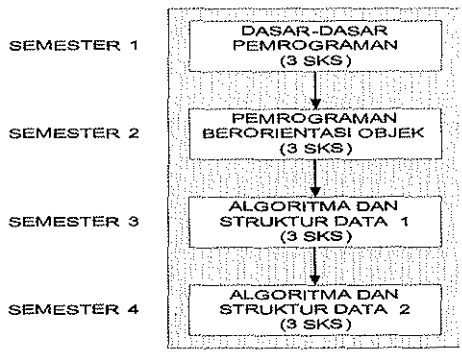
Pengalaman dalam pengajaran pemrograman dengan kurikulum 2003, *trend* saat ini, dan juga masukan dari pihak industri menjadi bahan pertimbangan pada saat penyusunan kurikulum 2008. Sejak semester Ganjil 2008/2009, jurusan Teknik Informatika mulai menerapkan adanya tiga peminatan atau subjurusan, yaitu Ilmu Komputer (*computer science*), Teknologi Informasi Bisnis, dan Telematika. Pada empat semester pertama semua mahasiswa menjalankan empat semester bersama, dan pada semester kelima mahasiswa harus memilih salah satu peminatan tersebut. Terkait

2) Seperti yang dilaporkan di [3]

dengan pemrograman, untuk menjaga agar kompetensi pemrograman minimal yang harus dimiliki oleh lulusan dapat terjaga, maka mata kuliah-mata kuliah pemrogaman dasar diberikan pada empat semester bersama.

Jalur mata kuliah pemrograman pada Kurikulum 2008 diberikan pada Gambar 2. Pendekatan pengajaran pemrograman yang digunakan adalah *object first* dan bahasa pemrograman yang diberikan adalah bahasa Java. Mata kuliah-mata kuliah tersebut meliputi Dasar-dasar Pemrograman (semester 1), Pemrograman Berorientasi Objek (semester 2), Algoritma dan Struktur Data 1, dan Algoritma dan Struktur Data 2 (semester 4). Seperti pada kurikulum 2003, mata kuliah pada semester sebelumnya akan menjadi prasyarat bagi mata kuliah berikutnya. Hanya prasyarat lulus tidak lagi diterapkan pada kurikulum 2008, tetapi cukup prasyarat tempuh. Paradigma pemrograman prosedural diberikan pada semester 5 dan hanya wajib bagi subjurusan Ilmu Komputer dan Telematika. Sedangkan paradigma pemrograman logik dan fungsional diberikan sebagai mata kuliah pilihan.

Mata kuliah Dasar-dasar Pemrograman bertujuan untuk memperkenalkan konsep-konsep dasar pemrograman berorientasi objek dan dasar-dasar dari bahasa pemrograman Java. Untuk pengajarannya digunakan pendekatan visual. Pada setengah semester pertama, mahasiswa diperkenalkan dengan konsep-konsep dasar pemrograman melalui penggunaan Alice. Setelah itu mahasiswa diperkenalkan dengan sintaks bahasa Java. Pengetahuan tentang bahasa Java ini selanjutnya akan diperdalam pada mata kuliah Pemrograman Berorientasi Objek. Sedangkan pengetahuan dan pemahaman struktur data dan algoritma baik yang dasar maupun lanjut diberikan pada mata kuliah Struktur Data dan Algoritma 1 dan Struktur Data dan Algoritma 2.



Gambar 2. Jalur mata kuliah pemrograman pada Kurikulum 2008.

Bentuk dan alur perkuliahan yang digunakan saat ini mirip dengan sebelumnya yaitu kuliah di kelas, praktikum di laboratorium, asistensi di laboratorium, dan kuliah

lagi di kelas. Asistensi di sini sebenarnya sama dengan responsi hanya dilaksanakan dalam kelompok-kelompok yang lebih kecil agar lebih efektif. Setiap asisten akan melayani maksimal sepuluh mahasiswa.

Pemilihan pendekatan visual dan khususnya dengan menggunakan Alice sebagai kakas pendukung perkuliahan adalah untuk mengatasi masalah yang mungkin dihadapi oleh mahasiswa dalam belajar pemrograman. Dengan keunggulan yang dimiliki oleh Alice, yaitu (i) objek-objek ditampilkan dalam bentuk tiga dimensi yang menarik, (ii) komputasi ditampilkan melalui animasi sehingga jalannya sebuah algoritma dapat dilihat secara interaktif (iii) hasil komputasi bisa langsung dilihat, dan (iv) tidak diperlukan penulisan program, diharapkan mengatasi lima faktor penyebab kegagalan dalam kuliah pemrograman seperti yang dibahas di bab IV, yaitu pengetahuan pemrograman yang kurang, abstraksi yang lemah, kesulitan dalam merancang program, ketidakmampuan melihat hasil komputasi, dan kesulitan dengan kakas yang digunakan.

Karena pendekatan baru ini baru dijalankan mulai semester Ganjil 2008/2009, maka belum bisa disimpulkan apakah pendekatan yang digunakan ini lebih baik dibandingkan dengan pendekatan yang sebelumnya. Berdasarkan pengamatan pada mahasiswa angkatan 2008, sekitar 80% dari jumlah mahasiswa tidak mengalami kesulitan dalam menyelesaikan kuliah Dasar-dasar Pemrograman. Mahasiswa yang tidak lulus pada pengambilan yang pertama sebagian besar lulus pada pengambilan yang kedua. Di sini dapat diambil kesimpulan sementara bahwa pendekatan pengajaran untuk mata kuliah Dasar-dasar Pemrograman berhasil baik. Khususnya untuk kakas pembelajaran pemrograman yang digunakan, yaitu Alice, sebagian besar mahasiswa memberi tanggapan positif. Komentar umum yang diberikan oleh mahasiswa adalah Alice cukup membantu mereka dalam pengenalan dasar pemrograman.

Masalah mulai muncul pada mata kuliah Pemrograman Berorientasi Objek. Tingkat kelulusan untuk mata kuliah ini tidak setinggi mata kuliah Dasar-dasar Pemrograman yaitu hanya sekitar 50%, bahkan seperlima dari mahasiswa yang lulus hanya lulus dengan nilai D. Hasil analisis sementara terhadap situasi ini adalah mahasiswa cukup kaget dengan perubahan pendekatan pengajaran yang cukup drastis. Pada mata kuliah ini Alice tidak digunakan lagi sebagai kakas pendukung perkuliahan. Di sini mahasiswa diperkenalkan dengan bahasa pemrograman yang sesungguhnya, dimana mereka harus benar-benar menulis (mengetik) program sesuai dengan sintaks Java dan menguji kebenaran program, terutama dari logiknya. Sebagai akibatnya kesulitan terkait dengan kakas yang digunakan muncul di sini, yaitu mahasiswa harus berhadapan dengan

pesan-pesan kesalahan yang cukup sulit untuk dipahami. Lebih jauh lagi, masalah ini sebenarnya dipicu oleh tujuan awal Alice yang ingin membebaskan mahasiswa dari kompleksitas sintaks bahasa pemrograman Java pada saat awal belajar pemrograman.

Mahasiswa yang telah lulus Pemrograman Berorientasi Objek sesuai dengan kurikulum akan menempuh mata kuliah Algoritma dan Struktur Data 1. Mahasiswa yang tidak lulus Pemrograman Berorientasi Objek cenderung untuk mengulang dan tidak mengambil mata kuliah Algoritma dan Struktur Data 1 meskipun secara kurikulum diperbolehkan. Tingkat kelulusan mata kuliah Algoritma dan Struktur Data 1 cukup tinggi yaitu sekitar 70 %. Hal ini menunjukkan bahwa mahasiswa yang mampu untuk lulus mata kuliah-mata kuliah pemrograman awal dengan baik cenderung untuk tidak mengalami kesulitan pada mata kuliah-mata kuliah pemrograman selanjutnya.

Secara umum dari data yang sudah terkumpul dapat diambil kesimpulan awal bahwa pendekatan visual pada pengajaran pemrograman dapat mengurangi rasa frustasi yang sering muncul pada saat mahasiswa pertama kali beradaptasi dengan bahasa pemrograman. Perlu diingat bahwa kesimpulan awal ini dibuat berdasarkan data yang baru berasal dari satu angkatan dan masih harus dilakukan pengumpulan data lebih lanjut sebelum dapat diambil kesimpulan yang valid. Selain itu, data-data yang dipertimbangkan dalam pengambilan kesimpulan ini baru mencakup distribusi nilai dari mata kuliah pemrograman.

Faktor-faktor lain yang juga dapat mempengaruhi hasil dari pendekatan visual adalah kemampuan tenaga pengajar untuk menjelaskan sesuai dengan kakas yang digunakan. Tenaga pengajar dituntut untuk mampu mengajarkan konsep-konsep dasar pemrograman yang berlaku umum tanpa terjebak menjadi hanya mengajarkan mengenai kakas tersebut. Selain itu, tenaga pengajar juga harus dapat melakukan proses transisi antara kakas visual dengan bahasa pemrograman dengan memanfaatkan konsep-konsep dasar yang telah diajarkan dengan menggunakan kakas visual tersebut. Hal ini diperlukan agar tidak ada konsep-konsep yang seakan-akan terputus antara kakas visual dengan bahasa pemrograman.

VII. Penutup

Pengajaran pemrograman di perguruan tinggi, khususnya pada program studi Teknik Informatika dan sejenis, mempunyai masalah yang cukup kompleks. Banyak kesulitan yang dihadapi oleh mahasiswa, mulai dari kurangnya bekal yang diperoleh di sekolah menengah sampai dengan kurangnya motivasi. Masalah-masaah yang berhasil dihimpun tersebut menjadi tantangan bagi banyak pihak khususnya bagi

perguruan tinggi penyelenggara program studi Teknik Informatika untuk mencari cara agar pengajaran pemrograman bisa berjalan dengan lebih baik sehingga tercapai tujuan memberikan kompetensi pemrograman yang tinggi. Berbagai pendekatan telah diusulkan untuk mengatasi permasalahan tersebut.

Selain apa yang telah dipaparkan pada makalah ini, tentu saja, masih terdapat isu-isu lain yang terkait dengan pengajaran pemrograman, misalnya peninjauan pengajaran pemrograman dari aspek pedagogi, peninjauan pengajaran pemrograman dari sisi dosen pengajar, masalah lain yang terkait dengan fasilitas yang diperlukan untuk mendukung pengajaran (laboratorium), dan juga isu tentang plagiarisme dalam konteks pemrograman.

Acuan:

1. Jackei O'Kelly and J. Paul Gibson. RoboCode & Problem-based Learning: A non-prescriptive approach to teaching programming. ITiCSE'06, Univ. Of Bologna, Italy.
2. Qusay H. Mahmoud, et.al. Making Computer Programming Fun and Accessible. IEEE Computer 37(2): 106-108 (2004).
3. Lucky Adhie dan Cecilia E. Nugraheni. Pembelajaran Pemrograman dengan Game-based Learning. Prosiding Seminar Nasional Ilmu Komputer (SNIKA) 2008, Jurusan Teknik Informatika Unpar, 2008.
4. Samer Al-Imamy, Javanshir Alizadeh, Mohamed A. Nour. On the Development of a Programming Teaching Tool: The Effect of Teaching by Templates on the Learning Process. Journal of Information Technology Education Volume 5, 2006.
5. Scott Leutenegger dan Jeffry Edgington. A Games First Approach to Teaching Introductory Programming. Proc. of SIGCSE 2007.
6. Josh Tenenberg. The Meaning of Computer Programs. M. Beynon, C.L. Nehaniv, and K. Dautenhahn (Eds.): CT 2001, LNAI 2117, pp. 165-174, 2001. Springer-Verlag Berlin Heidelberg 2001.
7. Konsep Pemrograman Prosedural. http://kur2003.if.itb.ac.id/file/Konsep%20 Pemrograman %20 Prosedural.pdf (tanggal akses 17 Februari 2010).
8. Dasar Pemrograman. http://kur2003.if.itb.ac.id/file/ IF1281Dasar Pemrograman. pdf (tanggal akses tanggal akses 17 Februari 2010)
9. Jorma Sajaniemi and Chenglie Hu. Teaching Programming: Going beyond "Object First". In Romera, J. Good, E. Acosta Chaparro & S. Bryant (eds). Proceeding in 18th Workshop of the Psychology of Programming Interest Group, University of Sussex, September 2006.

10. Lukasz Muziol. Teaching programming: Modern approaches using tools dan dynamic languages. http://gride.googlecode.com/files/lmuziol-teaching-summary.pdf (tanggal akses 5 Maret 2010).
11. Kirsti Ala-Mutka. Problems in Learning and Teaching Programming. http://www.cs.tut.fi/~edge/literature_study.pdf (tanggal akses 1 Maret 2010).
12. Esteban Walter Gonzalez Clua. A Game Oriented Approach for Teaching Computer Science. SBC Anals of XXVIII Congresso da SBC.
13. Du Boulay. Some difficulties of learning to program. In Soloway & Sphorer: Studying the Novie Programmer, pp. 290-294.
14. Andrew Rudder, et.al. Teaching Programming using Visualization. Proceeding of Web Based Education, 2007.
15. Myungsook Klassen. Visu Azad Ali, Frederick G. Kohun, and Robert Morris. Considerations for Selecting Programming Language to Teach Prespective Teachers. Proceeding of Alice Symposium 2009al Approach for Teaching Programming Concepts. Proceeding of 9th International Conference on Engineering Education.
16. Lynne P. Baldwin and Jasna Kuljis. Visualisation Techniques for Learning and Teaching Programming. Journal of Computing and Information Technology –CIT 8, 2000, 4, 285 – 291.
17. L. L. Tripp. A Survey of graphical notations for program design: an update. ACM SIGSOFT Software Engineering Notes. 13(4) (1988), 39-44.
18. J.L. Diaz-Herrera and R.C. Flude. Pascal/HSD: A graphical programming system. IEEE Proceedings COMSAC (1980), pp. 723-728.
19. S. Rodger. Introducing Computer Science Through Animation and Virtual Worlds. SIGCSE Technical Symposium on Computer Science Education. 2002.
20. Alice. http://www.alice.org (tanggal akses 29 Juli 2008).
21. Ceebot. http://www.ceebot.com (tanggal akses 1 Agustus 2008).

Ucapan terima kasih

Dalam penyusunan makalah ini, saya banyak dibantu oleh rekan-rekan dosen Jurusan Teknik Informatika Unpar. Untuk itu saya mengucapkan terima kasih sebesar-besarnya kepada tim pengajar pemrograman: Lucky Adhie,Skom., Beatrix,Skom, dan Kristopher D. Harjono, Skom. atas *sharing* pengalaman mengajar pemrograman dan data-data statistik yang telah diberikan. Juga kepada dua rekan senior saya, Dr. Oerip S. Santosa dan Dr. Veronica S. Moertini, terima kasih atas masukan yang sangat berguna bagi penyempurnaan makalah ini.

CURRICULUM VITAE

IDENTITAS DIRI

Nama	:	Dr.rer.nat. Cecilia Esti Nugraheni, ST, MT
Tempat dan Tanggal Lahir	:	Solo, 27 November 1969
Jenis Kelamin	:	Perempuan
Status Perkawinan	:	Kawin
Agama	:	Katolik
Golongan / Pangkat	:	Penata / III C
Jabatan Akademik	:	Lektor
Alamat Rumah	:	Jl. Taman Dirgantara no. 6 Bandung 40295
Telp./Faks.	:	022-70797467
Alamat e-mail	:	cheni@home.unpar.ac.id

RIWAYAT PENDIDIKAN PERGURUAN TINGGI

Tahun Lulus	Program Pendidikan	Perguruan Tinggi	Jurusan
1993	Sarjana	Institut Teknologi Bandung	Teknik Informatika
1997	Magister	Institut Teknolog Bandung	Teknik Informatika
2004	Doktor	Ludwig-Maximilians Universität, Munich, Jerman	Informatika

PELATIHAN PROFESIONAL

Tahun	Jenis Pelatihan(Dalam/ Luar Negeri)	Penyelenggara	Jangka waktu
2000	International Summer School on Engineering Theories of Software Construction	Technische Universitaet Muenchen, Germany	25 Juli – 6 Agt 2000
2001	Course on Formalware Engineering: Formal Methods for Engineering Software	CISM, Udine, Italy	24-28 Sep 2001
2002	Summer school on MOdélisation et VÉrification des processus Parallèles	University of Nantes, Nantes, Perancis	17-21 Jun 2001
2002	Course on "Formal methods for the Design of Computer Communication and Software System: Model Checking"	Centro Residenziale Univ., Bertinoro, Italy	8-14 Sep 2002
2009	Pelatihan Oracle 10g: Introduction to SQL	InformIT, Bandung	16 Jan – 3 Feb 2009
2009	Pelatihan Oracle 10g: Administration Workshop I	InformIT, Bandung	10-24 Mar 2009
2010	Pelatihan Java Programming	InformIT, Bandung	4-8 Jan 2010

PENGALAMAN MENGAJAR

Mata Kuliah	Program Pendidikan	Institusi/Jurusan/ Program Studi	Sem/Tahun Akademik
Struktur Diskret	Sarjana	Unpar / Teknik Informatika	Genap/03-04
Metode Formal	Magister	ITB / Teknik Informatika	Ganjil/04-05
Struktur Diskret	Sarjana	Unpar / Teknik Informatika	Genap/04-05
Metode Formal	Sarjana	Unpar / Teknik Informatika	Genap/04-05
Pengantar Sistem Cerdas	Sarjana	Unpar / Teknik Informatika	Ganjil/05-06
Teori Bahasa dan Kompilasi	Sarjana	Unpar / Teknik Informatika	Ganjil/05-06
Metode Formal	Magister	ITB / Teknik Informatika	Ganjil/05-06
Logika Formal Dasar	Sarjana	Unpar / Teknik Informatika	Genap/05-06
Model Checking	Sarjana	Unpar / Teknik Informatika	Genap/05-06
Teori Bahasa dan Kompilasi	Sarjana	Unpar / Teknik Informatika	Ganjil/06-07
Metode Formal	Magister	ITB / Teknik Informatika	Ganjil/06-07
Logika Formal Dasar	Sarjana	Unpar / Teknik Informatika	Genap/06-07
Metode Formal	Sarjana	Unpar / Teknik Informatika	Genap/06-07
Pemrograman Logik	Sarjana	Unpar / Teknik Informatika	Genap/06-07
Teori Bahasa dan Kompilasi	Sarjana	Unpar / Teknik Informatika	Ganjil/07-08
Jaringan Syaraf Tiruan	Sarjana	Unpar / Teknik Informatika	Ganjil/07-08
Logika Formal Dasar	Sarjana	Unpar / Teknik Informatika	Genap/07-08
Bahasa Pemrograman	Sarjana	Unpar / Teknik Sipil	Genap/07-08
Metode Formal	Sarjana	Unpar / Teknik Informatika	Genap/07-08
Struktur Diskret	Sarjana	Unpar / Teknik Informatika	Ganjil/08-09
Desain dan Analisis Alg.	Sarjana	Unpar / Teknik Informatika	Ganjil/08-09
Teori Bahasa dan Otomata	Sarjana	Unpar / Teknik Informatika	Ganjil/08-09
Logika Formal Dasar	Sarjana	Unpar / Teknik Informatika	Genap/08-09
Teknik Kompilasi	Sarjana	Unpar / Teknik Informatika	Genap/08-09
Bahasa Pemrograman	Sarjana	Unpar / Teknik Sipil	Genap/08-09
Metode Formal	Magister	ITB / Teknik Informatika	Genap/08-09
Struktur Diskret	Sarjana	Unpar / Teknik Informatika	Ganjil/09-10
Teori Bahasa dan Otomata	Sarjana	Unpar / Teknik Informatika	Ganjil/09-10

KARYA ILMIAH

A. Buku/Bab Buku/Jurnal

Tahun	Judul	Penerbit/Jurnal
2004	Predicate Diagrams as Basis for the Verification of Reactive Systems	Hieronymus, Munich, Germany
2007	Diagram-based Verification of Real-time Systems using Timed Predicate Diagrams	International Journal of Computer Science and Network Security (IJCSNS)
2008	Diagram-based Verification of Parameterized Systens	Journal of Combinatorial Mathematics and Combinatorial Computing (JCMCC)
2008	Spesifikasi dan Verifikasi Bounded	Jurnal Ilmu Komputer
	Retransmission Protocol Secara Formal dengan Temporal Logic of Actions	
2009	Verification of Ring-based Leader Election Protocol using Predicate Diagrams*	International Journal of Computer Science and Network Security (IJCSNS)

B. Makalah/Poster

Tahun	Judul	Penyelenggara
2002	Predicate Diagrams as Basis for the Verfication of Reactive Systems	University of Nantes, Nantes, Perancis
2002	Prediag: A Tool for the Generation of Predicate Diagrams	Charles University, Praha, Cheko
2004	Verifikasi Sistem Berparameter dengan Diagram Predikat Berparameter	Institut Pertanian Bogor, Bogor
2005	Universal Properties Verification of Parameterized Parallel Systems	Institute of High Performance Computing (IHPC) Singapore
2005	Formal Specification of Real-time Systems using TLA+	Universitas Mercu Buana, Jakarta
2005	Pengembangan Perangkat Lunak Secara Formal dengan Metodologi Raise	Universitas Satya Wacana, Salatiga
2006	Mutual Verification of Parameterized Reader-Writer Algorithm: A Case Study	Universitas Islam Indonesia, Yogyakarta
2006	Penerapan Algoritma Simulated Annealing untuk Penjadwalan Sidang Seminar	STIKOM, Surabaya
2006	Pembuatan Spesifikasi Protokol Secara Formal dengan TLA+	STTTelkom, Bandung
2007	Formal Specification of Reactive Systems	Universitas Ahmad Dahlan, Yogyakarta
2007	Formal Software Development using RAISE Methodology	Universitas Ahmad Dahlan, Yogyakarta
2008	Penyelesaian Penjadwalan Ujian dengan SAT	STIKOM Denpasar, Bali
2008	Pembelajaran Pemrograman dengan Pendekatan Game-based Learning	Teknik Informatika Unpar

KONFERENSI/SEMINAR/LOKAKARYA/SIMPOSIUM

Tahun	Judul Kegiatan	Penyelenggara	Panitia/ peserta/ pembicara
2004	Seminar "New Era of Mastering Science and Technology of Database: a Good News for IT Education and Practice"	Institut Teknologi Bandung	Peserta
2004	Seminar Nasional Ilmu Komputer dan Teknologi Informasi (SNIKTI)	Institut Pertanian Bogor (IPB)	Pembicara
2004	Workshop "Frontier in Information Communication Technology"	Institut Teknologi Bandung	Peserta
2005	Workshop "Indonesia .Net Curriculum"	Microsoft Indonesia- ITB	Peserta
2005	International Conference on Computational Science and Its Applications - ICCSA 2005	Institute of High Performance Computing (IHPC), Singapore	Pembicara

Tahun	Judul Kegiatan	Penyelenggara	Panitia/ peserta/ pembicara
2005	Seminar Nasional Matematika FMIPA-Unpar 2005	Jurusan Matematika – Unika Parahyangan	Peserta
2005	International Conference on "Information and Communication Technology (ICCT – UMB 2005)"	Universitas Mercu Buana, Jakarta	Pembicara
2005	Seminar Nasional Ilmu Komputer dan Teknologi Informasi VI	Universitas Satya Wacana, Salatiga	Pembicara
2005	Seminar Nasional Software Vaganza 2005	Jurusan Teknik Informatika – ITB	Peserta
2006	Universities' Leadership Forum & International Seminar "The Roles of Higher Education in National Development and Its Impacts on strategies and Sustainability"	DIKTI – UNESCO	Peserta
2006	Seminar Ilmu Komputer "e-Learning"	Jurusan Teknik Informatika – Unika Parahyangan	Pembicara
2006	Penataran dan Lokakarya Manajemen Berkala Ilmiah	DP2M DIKTI	Peserta
2006	Workshop on Data Mining Technology	Jurusan Teknik Informatika – ITB	Peserta
2006	Seminar Nasional Aplikasi Teknologi Informasi (SNATI) 2006	Universitas Islam Indonesia, Yogyakarta	Pembicara
2006	Seminar Nasional Sistem & Teknologi Informasi (SNASTI) 2006	STIKOM Surabaya	Pembicara
2007	International Seminar on Natural Sciences and Applied Natural Sciences	FMIPA - Universitas Ahmad Dahlan, Yogyakarta	Pembicara
2007	International Conference on Graph Theory and Information Security	Jurusan Matematika – Institut Teknologi Bandung	Pembicara
2008	Seminar Nasional Aktualisasi Website sebagai Sarana Komunikasi antara Perguruan Tinggi dengan Stakeholder	Politeknik POS Bandung dan Asosiasi Perguruan Tinggi Informatika dan Komputer (APTIKOM)	Peserta
2008	APTIKOM 2008 National General Meeting	APTIKOM	Peserta
2008	International Seminar "Research and Curricula Trend in Informatics and Computer Science"	APTIKOM	Pembicara
2008	Musyawarah Nasional APTIKOM	APTIKOM	Peserta
2008	Konferensi Nasional Sistem & Informatika 2008	STIKOM, Bali	Pembicara

KEGIATAN PROFESIONAL/PENGABDIAN KEPADA MASYARAKAT

Tahun	Jenis/Nama Kegiatan	Tempat
2005	Pengamat/ Seminar Guru-guru SMA	SMA St. Aloysius Bandung
2006	Koordinator presentasi / Open House Unpar	Unpar
2003-2004	Dewan Penyunting / Majalah Ilmiah Integral	Unpar
2005	Juri / Lomba Presentasi Karya Ilmiah INAYS	Unpar
2006	Juri / Lomba Presentasi Karya Ilmiah INAYS	Unpar
2006	Dewan Penyunting / Seminar Nasional Ilmu Komputer dan Aplikasinya (SNIKA) 2006	Unpar
2006	Moderator / Seminar Penjaminan Mutu Perguruan Tinggi dan Sistem Pangkalan Data Pendukungnya	Unpar
2006	Panitia / Seminar "The collaboration of University, Government, and Industry to socialize Networking Technology"	Unpar
2006	Panitia / Musyawarah Nasional APTIKOM	Unpar
2007	Pembicara / Seminar "Meningkatkan Kesadaran Dosen dalam Mengembangkan Kualitas Pembelajaran"	Universitas Pasundan Bandung
2007	Dewan Penyunting / Seminar Nasional Ilmu Komputer dan Aplikasinya (SNIKA) 2007	Unpar
2007	Panitia / Pelatihan Pengelolaan Data Berbasis Penjaminan Mutu	Unpar
2007	Anggota / Pemantau Independen Ujian Nasional SMA Tingkat Kota Bandung Th. Pelajaran 06/07	SMA Negeri 24 Bandung
2007	Panitia / Seminar Nasional Pembelajaran Algoritma dan Pemrograman di Perguruan Tinggi	Unpar – Galeri Ciumbuleuit Bandung
2008	Program Committee / Software Engineering and Formal Methods (SEFM) 2008	Capetown, South Africa
2008	Reviewer / Seminar Ilmiah Ilmu Komputer Nasional (SILICON) 2008	Universitas Pelita Harapan
2008	Dewan Penyunting / Seminar Nasional Ilmu Komputer dan Aplikasinya (SNIKA) 2008	Unpar
2009	Reviewer / Asia Modelling Symposium (AMS)	Unpar
2009	Dewan Penyunting / Seminar Nasional Ilmu Komputer dan Aplikasinya (SNIKA) 2009	Unpar

JABATAN DALAM PENGELOLAAN INSTITUSI

Peran/Jabatan	Institusi(Univ,Fak,Jurusan,Lab,studio, Manajemen Sistem Informasi Akademik dll)	Tahun ... s.d. ...
Kepala Lab.	Lab. Komputasi FMIPA Unpar	1996 – 1998
Ketua Jurusan	Jurusan Ilmu Komputer / Teknik Informatika Unpar	2004 – sekarang
Anggota	Senat Fakultas	2004 – sekarang
Anggota	Senat Universitas	2004 – sekarang
Wakil Dekan I	Fakultas Teknologi Informasi dan Sains Unpar	2009 – sekarang

PENGHARGAAN/PIAGAM

Tahun	Bentuk Penghargaan	Pemberi
1995-1997	Beasiswa URGE	DIKTI
1999-2004	Beasiswa	DAAD
2007	Insentif Publikasi di Jurnal Internasional	DIKTI
2008	Insentif Publikasi di Jurnal Internasional	DIKTI
2009	Insentif Publikasi di Jurnal Internasional	DIKTI

ORGANISASI PROFESI/ILMIAH

Tahun	Jenis/ Nama Organisasi	Jabatan/jenjang keanggotaan
2007-skg	Aptikom	Anggota
1999-skg	IEEE	Anggota


FAKULTAS TEKNOLOGI INFORMASI DAN SAINS
UNIVERSITAS KATOLIK PARAHYANGAN
BAKUNING HYANG MRIH GUNA
SANTYAYA BHAKTI