Jurnal Ilmu Komputer Agri-Informatika

Tersedia secara online di:
http://journal.ipb.ac.id/index.php/jika

Volume 2 Nomor 2 halaman 47 - 54
ISSN: 2089-6026

Pengembangan Sistem Manajemen Pembelajaran Pemrograman Bahasa Scheme, Java, PHP, dan Perl

Development of Programming Learning Management System for Scheme, Java, PHP, and Perl Language

ALFA NUGRAHA PRADANA, JULIO ADISANTOSO*

Abstrak

Secara konvensional proses pembelajaran dilakukan dalam bentuk diskusi ataupun penyampaian materi di dalam suatu kelas atau perkuliahan. Proses pembelajaran seperti ini sulit diterapkan pada bidang pemrograman terutama untuk kelas besar. Keterampilan dalam membuat suatu algoritme, logika, bahasa pemrograman, dan pengetahuan-pengetahuan lain seperti matematika juga sangat diperlukan dalam membuat suatu program komputer. Pada kondisi ini, sistem manajemen pembelajaran atau *learning management system* (LMS) memiliki peranan penting dalam melengkapi dan memperbaiki proses pembelajaran konvensional di bidang pemrograman. Tim Olimpiade Komputer Indonesia (TOKI) telah mengembangkan aplikasi SMP pemrograman yang disebut TOKI *Learning Contest* (LC) untuk bahasa pemrograman *Pascal*, C, dan C++. Penelitian ini menganalisis TOKI LC dan mengembangkan sistem manajemen pembelajaran agar dapat mendukung dan melengkapi proses pembelajaran di bidang pemrograman untuk mahasiswa bidang studi ilmu komputer atau informatika dengan menambahkan sistem penilaian bahasa *Scheme*, *Java*, *PHP*, dan *Perl* dan menguji kinerja masing-masing bahasa. Penelitian ini telah menambahkan *grader* otomatis untuk empat bahasa pemrograman yaitu Scheme, Java, PHP, dan Perl, yang memenuhi konsep *asynchronous e-learning*. Hasil menunjukkan bahwa kinerja *run time* pada bahasa pemrograman Java menggunakan waktu yang lebih lama. Selain itu penggunaan *memory space* pada bahasa Java menggunakan memori yang cukup besar untuk menjalankan setiap program dibandingkan bahasa pemrograman lain.

Kata kunci: *grader*, pemrograman, sistem manajemen.

Abstract

In conventional manner, learning process is done in the form of discussion or direct delivery of learning materials in a class or lecture. Programming is not easily applied to a conventional learning process. In this condition, LMS has an important role to improve the process of learning for programming. TOKI LC has been developed to identify the programming in Pascal, C, and C++ languages. However, this system has not been equipped with a programming language that is often used by student majoring in computer science or informatics. Therefore, this research aims to analyze and develop four other programming languages into learning management systems in the field of programming. LMS for programming was analyzed according to the concepts of e-learning, storyboard, and grading model. The system was developed with the prototype model on method of development grader, and then tested. Analysis shows that the system is adopting asynchronous learning with several main functional components, namely portal, baseline LMS features, and LCMS. Development of grader generates the Java programming language as a language with run time performance and memory space usage greater than the other programming languages, namely Pascal, C, C++, Scheme, PHP, and Perl.

Keywords: grader, learning management, programming.

Departemen Ilmu Komputer, Fakultas Matematika dan Ilmu Pengetahuan Alam, Institut Pertanian Bogor, Bogor 16680
*Penulis korespondensi: Tel/Faks: 0251-8625584; Surel: julioipb@gmail.com

PENDAHULUAN

Learning management system (LMS) atau sistem manajemen pembelajaran adalah sistem yang memudahkan proses administrasi, pengelolaan, dokumentasi, penelusuran, dan pelaporan untuk suatu program pembelajaran seperti *e-learning* (Hall 2000). Secara konvensional proses pembelajaran dilakukan dalam bentuk diskusi ataupun penyampaian materi di dalam suatu kelas atau perkuliahan. Proses pembelajaran seperti ini sulit diterapkan pada bidang pemrograman terutama untuk kelas besar karena membutuhkan komunikasi antara dosen dan mahasiswa yang lebih intensif. Membuat program komputer untuk pemecahan masalah sangat memerlukan pemahaman di bidang algoritme, logika, dan sintaks bahasa pemrograman. Pada kondisi ini, LMS memiliki peranan penting dalam proses pembelajaran di bidang pemrograman melalui latihan terstruktur dan diskusi *online*.

Terdapat banyak aplikasi LMS pemrograman, antara lain sphere online judge (SPOJ), TopCoder, international olympiad in informatics (IOI), dan TOKI LC. TOKI LC dikembangkan pertama kali oleh Barus (2009) untuk memfasilitasi kontes pemrograman bagi pelajar sekolah menengah di Indonesia.

Colton *et al*. (2006) mengembangkan *WebBot*, *grader* atau sistem penilai program otomatis berbasis web (Colton *et al*. 2006). Sistem ini membuat antarmuka untuk mengurangi interaksi penggunaan *e-mail* dan *grader* yang berjalan secara otomatis. Akan tetapi, penambahan fitur sistem tersebut tidak memberikan laporan kepada pengajar mengenai kemajuan proses belajar mahasiswa dan tidak memenuhi konsep LMS yang menghubungkan *learner* dengan sumber pembelajaran (Peer3 2001).

Patil (2010) mengadopsi konsep LMS pada sistem *grader* otomatis untuk tugas pemrograman, *Javabrat*. Sistem ini tidak hanya memfasilitasi pelajar dalam mengakses sumber pembelajaran, tetapi juga memfasilitasi pengajar untuk melihat perkembangan mahasiswa dalam menguasai suatu bahasa pemrograman. Kelemahan *Javabrat* ialah hanya mengenal bahasa pemrograman Java dan Scala.

TOKI LC merupakan salah satu *platform grader* yang telah dikembangkan dengan fitur: antarmuka yang lebih baik dibanding *WebBot*, penilaian program secara otomatis melalui sistem, registrasi, manajemen *user*, pelaporan, dan dapat mengenali bahasa pemrograman *Pascal*, C, dan C++. Akan tetapi, sistem ini belum dilengkapi dengan bahasa pemrograman yang sering digunakan untuk mahasiswa bidang studi ilmu komputer atau informatika, antara lain *Scheme* pada aspek pemrograman fungsional, *Java* pada aspek pemrograman berorientasi objek, PHP pada aspek pemrograman prosedural, dan *Perl* sebagai bahasa yang dikembangkan dengan mencampur fitur terbaik dari beberapa bahasa menjadi satu (Al-Qahtani *et al*. 2010).

Oleh karena itu, penelitian ini akan menganalisis TOKI LC dan mengembangkan sistem manajemen pembelajaran agar dapat mendukung dan melengkapi proses pembelajaran di bidang pemrograman untuk mahasiswa bidang studi ilmu *komputer* atau informatika dengan menambahkan sistem penilaian bahasa *Scheme*, *Java*, *PHP*, dan *Perl* dan menguji kinerja masing-masing bahasa.

METODE

LMS pemrograman selanjutnya disebut LMSP. Pelaksanaannya memililiki tiga tahapan, yaitu analisis, implementasi, dan pengukuran kinerja.

Analisis

Tahap analisis yang dilakukan ini bertujuan mengadopsi konsep *e-learning* pada LMSP dengan menunjukkan dan mengoptimalkan konsep *e-learning* yang telah dibangun pada penelitian sebelumnya dan yang belum dikembangkan.

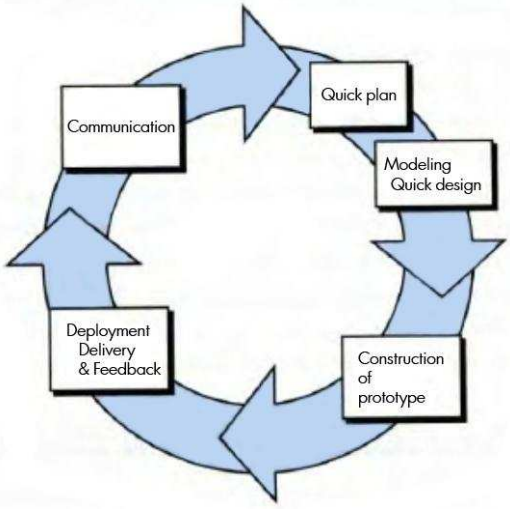
E-learning merujuk kepada penggunaan jaringan teknologi informasi secara intensional dalam pengajaran dan pembelajaran (Naidu 2006). LMSP termasuk ke dalam *asynchronous learning*, dimana proses pembelajaran tidak memerlukan komunikasi satu sama lain. Pada tahap ini, dilakukan analisis terhadap komponen fungsional *asynchronous learning* menurut Peer3 (2001), antara lain portal, akses penghubung *learner* dengan lingkungan *e-learning*; LMS, penghubung antara *learner* dan sumber atau isi pembelajaran yang terdiri atas fitur yang mendukung proses belajar mengajar; *learning content management systems* (LCMS), lingkungan tempat pengembang dapat membuat, menyimpan, menggunakan kembali, dan mengirim sumber pembelajaran dari pusat penyimpanan objek, biasanya *database*. LCMS umumnya bekerja dengan isi atau sumber berdasarkan model *learning object* (Abazi-Bexheti 2008).

Analisis *selanjutnya* ialah *learning objects* dan *storyboard*. *Learning objects* adalah unit-unit terpisah yang berisi objek pembelajaran, isi, penilaian objektif, fasilitas pencarian, pengindeksan dan penggunaan kembali objek pembelajaran (Peer3 2001). *Storyboard* sebagai salah satu bagian dalam pengembangan *learning objects* harus mengandung instruksi dan deskripsi secara detail, spesifik, dan terurut (Mustaro *et al.* 2005). Kedua analisis tersebut berhubungan dan dilakukan dengan mengidentifikasi ciri-ciri tersebut.

Analisis terakhir dilakukan dengan cara survei melalui kuesioner *online* berbasis web yang disebar di situs jejaring sosial dengan target responden mahasiswa yang masih aktif dan pernah mengikuti perkuliahan pemrograman. Tujuan survei ini adalah untuk mengetahui dan mengevaluasi sejauh mana mahasiswa mengenal, memahami dan menguasai pemrograman yang diberikan pada perkuliahan.

Implementasi

Sistem penilai program bahasa Scheme, Java, PHP, dan Perl pada LMSP dikembangkan menggunakan model *prototyping*. Pada tahap *communication*, metode diskusi dilakukan dengan dosen pemrograman sebagai pengguna untuk mendefinisikan tujuan dan mengidentifikasi kebutuhan yang diperlukan pada perangkat lunak. Hasil dari tahap ini digunakan sebagai deskripsi umum sistem dan analisis kebutuhan sistem (Gambar 1).



Gambar 1 Model pengembangan *prototyping* (Pressman 2005)

Pada tahap *quick plan and quick design modeling*, perancangan model *grader* tahap awal dilakukan dan diperbaiki sesuai permintaan pengguna. Implementasi dan pengembangan model *grader* dilakukan pada tahap *construction of prototype* sebagai penerjemahan rancangan analisis sistem ke dalam suatu bahasa pemrograman yang dapat dikenali oleh komputer.

Pengukuran Kinerja

Pengujian terhadap *grader* bahasa pemrograman *Scheme*, *Java*, *PHP*, dan *Perl* dilakukan menggunakan metode *black-box* atau *behavioral testing* yang memungkinkan pengembang perangkat lunak menurunkan sekumpulan kondisi *input* yang dapat memenuhi seluruh kebutuhan fungsional (Pressman 2005). Terdapat dua kinerja yang diuji, yaitu kinerja program komputer, dan perbandingan eksekusi program komputer terhadap tujuh bahasa pemrograman pada *grader* LMSP. Kinerja program komputer dilakukan pada program yang menggunakan bahasa pemrograman *Scheme*, *Java*, *PHP*, dan *Perl*. Program tersebut telah dikumpulkan oleh *learner* dan dinilai oleh *grader* dan juga oleh sistem *grader online Ideone* (IDEONE 2010). Hasil kinerja program komputer digunakan untuk membandingkan konsumsi memori dan waktu eksekusi program pada *grader* dan Ideone. Pada pengujian ini digunakan lima soal yang masing-masing telah dibuat solusi menggunakan bahasa pemrograman *Pascal*, *C*, *C++*, *Scheme*, *Java*, *PHP*, dan *Perl*. Selanjutnya *run time* (detik) dan penggunaan memori (MB) untuk setiap bahasa pemrograman diukur untuk setiap solusi.

HASIL DAN PEMBAHASAN

Analisis

Pada hasil analisis tahap awal, untuk mengakses lingkungan sistem ini telah dikembangkan suatu portal berupa halaman login. Learner yang telah terdaftar dalam suatu perkuliahan pemrograman diberi *username*/*email* dan *password* untuk mengikuti proses pembelajaran yang telah tersedia.

Hasil perbandingan keempat fitur LMS (TOKI LC, WebBot, Javabrat, dan Ideone) dengan LMSP yang telah dikembangkan tercantum pada Tabel 1. Hasil analisis komponen LMSP menunjukkan bahwa fitur pelaporan dalam bentuk statistik telah dikembangkan, dan *LMS-to-Content API*, serta LMS sebagai alat pembelajaran kolaboratif belum dikembangkan.

Analisis LCMS dilakukan dengan mengevaluasi komponen dasar dan beberapa kriteria evaluasi LCMS menurut Peer3 (2001), untuk menunjukkan sejauh mana konsep LCMS dapat digunakan untuk mengadopsi konsep *e-learning* pada LMSP. Hasil evaluasi untuk tiga komponen dasar LCMS yang dimiliki LMSP tercantum pada Tabel 2.

Storyboard pada LMSP diterapkan pada setiap soal pembelajaran pemrograman. Materi pada soal pemrograman dibuat sesuai skenario. Skenario tersebut digambarkan ke dalam situasi yang sering terjadi kehidupan nyata berbentuk suatu cerita atau narasi yang memiliki konteks, plot, karakter, dan parameter yang saling berhubungan. Cerita ini berisi kasus atau masalah yang memiliki suatu solusi yang dapat diselesaikan oleh *learner*. Pada akhir cerita, *learner* diminta memberikan solusi dengan membuat suatu program dengan format masukan, keluaran, dan bahasa pemrograman tertentu.

Dari hasil kuesioner didapatkan sebanyak 98 variasi jawaban. Persentase hasil yang diperoleh dari beberapa pertanyaan yang diberikan, antara lain bahwa persentase tertinggi untuk pertanyaan bahasa pemrograman yang pernah digunakan oleh mahasiswa adalah bahasa pemrograman C++ sebesar 81% dan yang paling jarang digunakan adalah bahasa pemrograman Perl sebesar 12%. Bahasa lainnya yaitu Visual Basic, Python, Matlab, Clips, Prolog, ASP .NET, dan ActionScript. Hasil ini menunjukkan bahwa diperlukan pembelajaran lebih lanjut dalam pengenalan beberapa bahasa pemrograman yang kurang populer. Selain itu, diperlukan sistem penilai program terhadap bahasa yang banyak digunakan oleh mahasiswa.

Tabel 1 Hasil perbandingan fitur LMS

Fitur LMS	TOKI LC (2009)	WebBot (2006)	Javabrat (2010)	Ideone (2010)	LMSP (2012)
Menurut Peer3 (2001):					
Registrasi	√	√	√	√	√
Manajemen *user*	√	√	√	-	√
Rencana Pembelajaran	√	√	√	-	√
Pelaporan	√	-	√	√	√
LMS-to-Content API	-	-	-	√	-
Menurut Naidu (2006):					
Pengiriman materi perkuliahan					
Manajemen transaksi kelas *online*	√	√	√	-	√
Penilaian hasil pembelajaran	√	√	√	-	√
Penelusuran dan pelaporan perkembangan *learner*	√	√	√	√	√
Pelaporan prestasi dan kemajuan penyelesaian tugas	√	-	√	-	√
Alat pembelajaran kolaboratif (perangkat portabel dan *mobile*)	-	-	-	√	-
Mengumpulkan, mengatur, dan melaporkan seluruh aktivitas pembelajaran *learner*	-	-	√	√	√

Tabel 2 Hasil evaluasi komponen dasar LCMS

Komponen dasar	Hasil evaluasi
Pusat penyimpanan *learning object*	*Database* terpasang pada komputer *server* yang tersedia di Gedung Rektorat IPB
Alat pembuatan dan penyusunan *learning object*	Media pembelajaran dibuat dalam format HTML
Database pada lingkungan pembelajaran berbasis web	*Database* sudah dikembangkan pada lingkungan berbasis web dan disesuaikan dengan materi

Pemahaman materi pemrograman rata-rata untuk jawaban "Tidak Mengerti" dan "Mengerti" memperoleh rata-rata 42%. Hasil ini menunjukkan bahwa proses pembelajaran konvensional tidak memberikan dukungan lebih dalam pembelajaran pemrograman sehingga diperlukan LMS pemrograman. Persentase tertinggi terhadap perilaku mahasiswa untuk belajar menguasai dan mendalami suatu bahasa pemrograman adalah mengerjakan latihan materi dan soal yang diberikan pada perkuliahan secara mandiri sebesar 80%, selanjutnya membaca buku pemrograman 71%, belajar mandiri dengan mengikuti tutorial *online* 54%, dan lainnya seperti mencari solusi di forum diskusi, dan bertanya antar sesama mahasiswa sebesar 26%. Proses belajar mandiri dalam pemrograman secara tidak langsung kurang membantu untuk pembelajaran pemrograman. Persentase mahasiswa yang pernah mengikuti kegiatan pembelajaran pemrograman secara *online* adalah 63% dan yang tidak pernah sebesar 37%.

Dari hasil persentase jawaban secara keseluruhan, terlihat bahwa dalam proses pembelajaran pemrograman secara konvensional, sebagian besar mahasiswa kurang menguasai pemrograman dan lebih banyak belajar secara mandiri dengan mengerjakan soal dan latihan, membaca buku pemrograman maupun mengikuti tutorial secara online. Selain itu, penjelasan konsep, contoh-contoh, materi, dan latihan pemrograman serta *feedback* merupakan kebutuhan yang sangat diperlukan untuk menguasai pemrograman. Oleh karena itu, LMS pemrograman dikembangkan sebagai solusi dengan sistem penilai bahasa yang banyak digunakan dan fitur yang dapat membantu mahasiswa dalam mengikuti proses pembelajaran pemrograman.

Implementasi *Grader Otomatis*

Beberapa variabel diperlukan dalam implementasi *grader*. Salah satu variabel tersebut adalah variabel yang berisi *path file evaluator*. *File evaluator* ini adalah *file* biner berekstensi *.exe yang memiliki fungsi mengatur batas waktu *compile* dan eksekusi, mengatur batas memori, memasukkan *file input* untuk menguji program, menulis hasil eksekusi program ke dalam *file output*, dan menulis hasil *compile* yang tidak sesuai atau *error*.

Untuk menjalankan seluruh fungsi tersebut secara lengkap, *file evaluator* memiliki pilihan yang digunakan sebagai parameter, antara lain `-a <level>`, mengatur tingkat akses *file* (0 = tidak ada, 1 = cwd, 2 = /etc, /lib, ..., 3 = seluruh akses, 9 = tidak diperiksa), `-e`, menurunkan seluruh lingkungan *compile* bahasa pemrograman ke dalam proses, `-i <file>`, mengalihkan standar *input* ke suatu *file*, `-m <size>`, batas ruang alamat memori dalam satuan KB, `-o <file>`, mengalihkan standar *output* ke suatu *file*, `-r <file>`, mengalihkan standar *error* ke suatu *file*, dan `-w <time>`, mengatur batas waktu yang disediakan.

Variabel lain yang diperlukan dalam proses implementasi, yaitu variabel pengatur batas waktu dan memori, variabel yang berisi *file temporary compile error* yang berfungsi untuk mencatat standar *error* ke dalam *file*, dan variabel perintah *compile*. Seluruh variabel untuk proses implementasi *grader* secara lengkap ialah sebagai berikut:

```
$moeval_path $parameter -r $tmperr -- $command
$moeval_path      = file evaluator,
$parameter  = parameter batas waktu  dan memori compile,
$tmperr     = file temporary compile error,
$command    = perintah compile.
```

Implementasi keseluruhan untuk mengkompilasi dan menjalankan program Scheme, Java, PHP, dan Perl secara berurut mengikuti tahap:

```
/* IMPLEMENTASI SCHEME */
/bin/mo-evalbox -e -f -a 2 -w [batas waktu] -m [batas memori] –i [file input] -o
[file output] -r [file temporary]  -- /usr/bin/guile [nama file program]

/* IMPLEMENTASI JAVA */
```

Perintah *compile*:

```
/bin/mo-evalbox -e -m [batas memori] -w [batas waktu] -r [file temporary] --
/usr/bin/javac [nama file program]
```

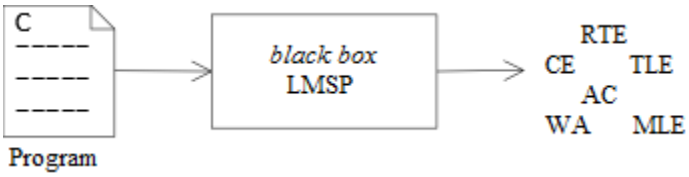
Perintah eksekusi:

```
/bin/mo-evalbox -e -a 2 -w [batas waktu] -m [batas memori] –i [file input] -o [file
output] -r [file temporary] -- /usr/bin/java –classpath /tmp/ [file hasil
kompilasi]

/* IMPLEMENTASI PHP */
/bin/mo-evalbox -e -f -a 2 -w [batas waktu] -m [batas memori] –i [file input] -o
[file output] -r [file temporary]  -- /usr/bin/php [nama file program]

/* IMPLEMENTASI Perl */
/bin/mo-evalbox -e -f -a 2 -w [batas waktu] -m [batas memori] –i [file input] -o
[file output] -r [file temporary]  -- /usr/bin/perl [nama file program]
```

Setelah *grader* diimplementasikan, pengujian dilakukan untuk mengetahui kecocokan sistem dan fungsi *compiler* yang telah dibuat. Pengujian dilakukan dengan metode *black-box* dengan memberikan *input* dan melihat *output*. *Input* yang digunakan adalah program dengan bahasa Scheme, Java, PHP, dan Perl. Program tersebut diuji coba untuk menghasilkan seluruh kriteria status penilaian pada sistem yang telah disebutkan sebelumnya, yaitu *compile error* (CE), *run time error* (RTE), *time limit exceeded* (TLE), *memory limit exceeded* (MLE), *wrong answer* (WA), dan *accepted* (AC). Kriteria status CE, RTE, TLE, MLE, dan WA dibandingkan dengan kriteria status AC sebagai perbandingan terhadap program yang berhasil dengan program yang gagal dinilai oleh *grader*. Skema pengujian *grader* tercantum pada Gambar 2 dan hasilnya secara *black-box* disajikan pada Tabel 3 .



Gambar 2 Skema pengujian *grader*.

Tabel 3 Hasil pengujian sistem secara *black-box*

Status	*Compiler*			
	Scheme	Java	PHP	Perl
CE	RTE	CE	RTE	RTE
RTE	RTE	RTE	RTE	RTE
TLE	TLE	TLE	TLE	TLE
MLE	MLE	MLE	MLE	MLE
WA	WA	WA	WA	WA
AC	AC	AC	AC	AC

Kriteria status CE tidak berlaku untuk program yang dikumpulkan dengan bahasa Scheme, PHP, dan Perl karena penilaian program oleh *grader* langsung dieksekusi oleh *interpreter* tanpa perlu dikompilasi terlebih dahulu seperti bahasa Java sehingga hasil uji program tersebut menjadi RTE. Untuk hasil pengujian pada kriteria status yang lain didapatkan bahwa seluruh fungsi telah berjalan sesuai kebutuhan baik pada *compiler* Scheme, Java, PHP maupun Perl. Hasil *output* tersebut sesuai dengan *input* program yang dinilai. Pada kode program yang diuji coba, kriteria status MLE didapatkan hanya dengan memberikan batas memori yang rendah tanpa harus membandingkan dengan status AC (Tabel 3).

SIMPULAN

Berdasarkan penelitian yang telah dilakukan, dapat disimpulkan bahwa telah ditambahkan *grader* otomatis untuk empat bahasa pemrograman yaitu Scheme, Java, PHP, dan Perl. Disamping itu, LMSP telah memenuhi konsep *asynchronous e-learning* didukung oleh komponen fungsional *Portal*, LMS, dan LCMS.

Rata-rata konsumsi memori dan rata-rata waktu eksekusi program pada Ideone lebih kecil dibandingkan dengan penilaian pada *grader* LMSP. Jika dilihat dari jenis bahasa pemrograman, kinerja *run time* pada bahasa pemrograman Java menggunakan waktu yang lebih lama dan memori yang cukup besar dibandingkan bahasa pemrograman lain.

DAFTAR PUSTAKA

Abazi-Bexheti L. 2008. Development of a learning content management systems. *Int J Sys App Eng Dev.* 2:1-5.

Al-Qahtani SS, Arif R, Guzman LF, Pietrzynski P, Tevoedjre A. 2010. Comparing selected criteria of programming languages Java, PHP, C++, Perl, Haskell, AspectJ, Ruby, COBOL, Bash Scripts and Scheme. [Internet] http://arxiv.org/ftp/arxiv/papers /1008/1008.3434.pdf . [diakses 6 Juli 2012].

Barus PN. 2009. TOKI Learning Center: sistem pelatihan kompetisi pemrograman komputer [skripsi]. Bandung: Institut Teknologi Bandung.

Colton D, Fife L, Winters R. 2005. Building a computer program grader. *Inf Sys Edu J*. 3(6).

Colton D, Fife L, Thompson A. 2006. A web-based automatic program grader. *Inf Sys Edu J*. 4(114):1-9.

Hall B. 2000. *Learning Management Systems: How to Choose the Right System for Your Organization.* Sunnyvale(US): Brandon-hall.

[IDEONE]. Ideone Contributors. 2010. Online IDE & debugging tool for programming language. [Internet]. http://ideone.com/ [diakses 12 Mei 2012].

Mustaro PN, Silveira IF, Omar N, Stump SMD. 2007. Structure of storyboard for development of interactive learning objects. Di dalam: *Proceedings of the 2005 Informing Science and IT Education Joint Conference*; Flagstaff, 16-19 Juni 2007. Sao Paulo, Brazil: Universidade Presbiteriana Mackenzie.

Naidu S. 2006. *E-Learning: A Guidebook of Principles, Procedures, and Practices*. Ed ke-2. New Delhi(IN): Commonwealth Educational Media Center for Asia.

Pressman RS. 2005. *Sofware Engineering: A Practitioner's Approach*. Ed ke-6. Singapura(SG): McGraw-Hill.

Patil A. 2010. Automatic grading of programming assignments [tesis]. San Jose: The Faculty of the Departement of Computer Science, San Jose State University.

[Peer3] Peer3 Author. 2001. Evaluating learning content management systems. http://www.jeffcaton.com/PDF_files/Peer3_LCMS.pdf [12 Mei 2012].