

Analisis Requirement Perangkat Lunak untuk Menjalankan *Process Based Assessment* (PBA) dalam Pembelajaran Pemrograman Dasar

Jonner Hutahaean^a, Ani Rahmani^b dan Joe Lian Min^c

^{abc}Jurusan Teknik Komputer dan Informatika, Politeknik Negeri Bandung, Bandung 40012
E-mail : jonnerh@jtk.polban.ac.id, anirahma@jtk.polban.ac.id, danjoelianmin@jtk.polban.ac.id,

ABSTRAK

Tulisan ini menguraikan tentang penerapan prinsip *Process Based Assessment* (PBA) yang tepat dijalankan dalam pembelajaran pemrograman dasar, serta hasil analisis *requirement* perangkat lunak untuk mendukung PBA tersebut. Kedua aspek tersebut merupakan luaran dari penelitian tahap pertama dari rencana 3 tahap penelitian, melalui skim penelitian Hibah Bersaing 2014. Telaah mengenai metode pembelajaran pemrograman dasar telah banyak dilakukan. Pelbagai strategi pengajaran telah pula diuji coba dengan pelbagai pendekatan, namun pengajaran pemrograman dasar dipandang selalu menarik untuk diteliti. Para akademisi di pelbagai negara telah berhasil mengangkat isu pengajaran pemrograman dasar dari sudut pandang yang semakin beragam. Dalam kaitannya dengan proses evaluasi, dikenal adanya tes formatif dan tes sumatif. Kedua mekanisme evaluasi ini, tepatnya dikaitkan dengan waktu pelaksanaan evaluasi. Formatif ditujukan untuk mengevaluasi siswa saat proses pembelajaran berlangsung, sedangkan summatif dilaksanakan saat proses pembelajaran berakhir. *Process based assessment* (PBA) adalah suatu implementasi dari evaluasi formatif. Dalam PBA, proses evaluasi dan penilaian berlangsung pada proses belajar siswa, dan bukan melakukan evaluasi terhadap hasil akhir siswa. Berdasarkan analisis dan eksplorasi teknologi maka untuk menjalankan PBA diperlukan sistem perangkat lunak yang mampu menginformasikan proses mikro dari suatu aktivitas pemrograman, misalnya *create file/program*, Editing (*insert, edit, copy-paste, cut, move, delete*) file; mampu mengidentifikasi siswa yang sedang masuk ke sistem dan melakukan aktivitas pemrograman, mampu menyimpan informasi mengenai *micro-process* dari suatu aktivitas pemrograman, setiap siswa; mampu memberikan *feedback* berdasarkan problem yang teridentifikasi; serta mampu menampilkan secara *real time* (event-streaming) aktivitas pemrograman siswa, dalam pengajaran formal.

Kata Kunci : *Process based assessment, micro process, event streaming*

1. PENDAHULUAN

Dalam pengajaran pemrograman, salah satu aspek yang perlu mendapat perhatian adalah proses evaluasi dan penilaian [1]. Proses tersebut dipandang serius, karena penugasan dalam pengajaran pemrograman perlu dilakukan secara berkesinambungan karena proses tersebut bukanlah hal yang mudah dan sederhana, memerlukan waktu yang tidak sebentar, menuntut dilakukannya tahapan yang mungkin dapat membosankan [2], serta harus memberi *feedback* (umpan balik) secepat mungkin terhadap hasil kerja siswa [3], [1].

Seandainya proses evaluasi dan penilaian dilakukan terhadap seluruh program dengan segera, dan siswa mendapat *feedback* dengan segera pula, maka siswa akan merespon hasil yang diperoleh dengan mengerjakan latihan dan tugas berikutnya. Dengan ini, motivasi

siswa dapat terus meningkat dan semakin *enjoy* dalam mengerjakan latihan-latihan berikutnya, sehingga hasil pengajaran dapat lebih maksimal [4].

Process Based Assessment (PBA) atau sering disebut sebagai *formative assessment* adalah suatu metode evaluasi dengan memanfaatkan berbagai 'alat' dan strategi untuk mengetahui dan menentukan tingkat pemahaman siswa, mengidentifikasi kendala, serta merencanakan langkah-langkah tindakan "intervensi" untuk meningkatkan pemahaman siswa selama proses belajar [5].

Dalam pemrograman, pemantauan terhadap aktivitas siswa selama pemrograman, perlu dilakukan agar pengajar dapat mengetahui kebiasaan, sikap, kemampuan, serta perilaku siswa, sehingga kendala yang dihadapi dapat

ditelusuri berdasarkan pada 'rekaman' hasil pemantauan tersebut[6],[7].

1.1 PBA dalam Pengajaran Pemrograman

Dalam merancang PBA untuk pengajaran pemrograman, mengacu pada 4 (empat) esensi dari PBA yang dikemukakan Pinchok, N and Brandt, W.P, 2009, yaitu bahwa konsep PBA akan dimanfaatkan untuk[5] :

1.1.1. Identifying the learning gap

Yakni upaya untuk mengetahui 'jarak' atau kesenjangan antara hal (dalam hal ini pengetahuan) yang siswa pahami dengan hal perlu siswa pahami; dan strategi apa yang harus dilakukan pengajar untuk memperkecil kesenjangan tersebut;

1.1.2. Feedback flow

Yakni bagaimana alur *feedback* yang harus dirancang dari pengajar untuk merespon setiap situasi dan menentukan strategi ke depannya, untuk meningkatkan pemahaman siswa;

1.1.3. Student actively

Siswa perlu menyadari aktivitas pembelajaran yang dijalannya, berikut proses evaluasi yang akan dilakukan terhadapnya. Dalam hal ini, siswa dan pengajar perlu melakukan sharing untuk membangun '*metacognitif skill*', dimana siswa dapat memahami kapan sesungguhnya dia memerlukan 'bantuan';

1.1.4. Learning progression

Yakni upaya mem-*breakdown* tujuan pembelajaran secara umum, kemudian turun menjadi sub-sub tujuan pada setiap materi ajar. Pengajar perlu memahami 'posisi' pemahaman siswa pada kurun waktu tertentu, pada setiap sub tujuan yang seharusnya dicapai, sehingga tujuan akhir pembelajaran dapat tercapai.

Dalam pengajaran pemrograman, PBA dipandang penting mengingat pemrograman merupakan skil yang harus terus diasah dan memerlukan pemantauan selama proses pembelajaran berlangsung[8]. Tabel 1 merupakan problem dalam pengajaran pemrograman, dan upaya solusi yang dapat diterapkan.

Tabel 1 .Problem Pengajaran Pemrograman

No	Identifikasi Masalah	Rumusan Arah Penelitian
1	problematika siswa dalam proses belajar pemrograman, sering tidak diketahui sejak dini, melainkan diketahui pada akhir semester atau pada saat yang tidak tepat, setelah tidak ada waktu untuk meluruskan, atau untuk membentuk <i>skill</i> pemrograman yang benar	menerapkan konsep <i>process base assessment</i> (PBA) agar persoalan siswa selama belajar dapat diketahui sesegera mungkin, dan pengajar dapat memberi arahan yang tepat dengan segera pula,
2	Arahan serta <i>feedback</i> dari	merumuskan satu mekanisme untuk

	pengajar bersifat adhock, situasional, potensi tidak konsisten, dan sulit menjangkau seluruh siswa secara bersamaan	dapat memantau aktivitas detil seluruh siswa selama belajar pemrograman, sehingga problem yang ditemui siswa dapat dideteksi sejak dini dan pengajar dapat memberi arahan dan bimbingan dengan tepat dan segera (<i>formatifassessment</i>)
3	Akan sulit bagi pengajar untuk dapat memantau aktivitas detil seluruh siswa secara langsung, setiap saat, dan bersama-sama, serta memberi arahan untuk kasus yang potensi sangat beragam pada setiap siswa	membuat <i>tools</i> untuk pemantauan dan pemberian <i>feedback</i> secara otomatis

Manfaat *process based assessment* (PBA) secara otomatis dalam pengajaran pemrograman dapat meningkatkan efektifitas dan efisiensi waktu secara signifikan. Efektifitas diukur dari tingkat capaian siswa terhadap *learning outcome* suatu materi, sedangkan efisiensi diukur dari berapa lama siswa memahami suatu materi;

Model *event stream processing* (ESP) dapat mendukung implementasi PBA dalam pengajaran pemrograman. Hal ini karena PBA menuntut *feedback* yang segera serta proses pemantauan secara *real time*. Oleh karena itu perlu dilakukan:

1. Analisis konsep dan penerapan *process based assessment* (PBA) untuk pengajaran pemrograman
2. Analisis dan perumusan *user requirement* untuk otomatisasi penerapan PBA dalam pengajaran pemrograman

Analisis tentang Process Based Assessment dilakukan untuk memantau aktivitas programming siswa secara secara *real time* dan berkala (minute to minute, harian, mingguan, triwulan, semester dan tahunan) sehingga diketahui proses secara mikro.

1.2 Micro Process

Konsep *micro-process* dalam pemrograman diartikan sebagai upaya untuk merekam aktivitas memrogram secara mikro (detil). Aktivitas yang dimaksud adalah mulai dari membuka file, mengeksekusi program, pengubahan nama identifier, aktivitas copy and paste, dan lain-lain. Konsep ini diperlukan untuk mendukung pengumpulan dan penyimpulan dari *primitive event* selama aktivitas [9].

2. METODE

Dalam pengembangan perangkat lunak untuk otomatisasi PBA, akan menerapkan konsep *micro-process* dan *event-stream-processing*. Hal ini ditujukan agar perangkat lunak yang dibuat, dapat menyediakan informasi secara detil (mikro) tentang aktivitas siswa dalam memrogram.

2.1. Event Stream Processing

Konsep *event stream processing*, digunakan untuk menunjang proses ‘perekaman’ aktivitas tersebut, sehingga pengajar memiliki informasi yang utuh tentang perilaku siswa dalam memrogram.

Konsep *event stream processing* (ESP) adalah suatu model untuk melakukan pemantauan secara langsung (*real time*). Isu dalam model ini adalah bagaimana data diakuisisi, direpresentasikan, dan kemudian ditransformasi sehingga memiliki makna yang utuh. Konsep ini banyak digunakan untuk bidang yang terkait dengan sistem terdistribusi, dan banyak pula digunakan untuk kasus-kasus terapan pada bidang transportasi, yakni untuk memantau peta situasi transportasi secara

langsung [10].

2.2. Analisis Problem Siswa

Kegiatan analisis problem siswa dalam aktivitas pemrograman, merupakan tahap awal mengenali kebutuhan perangkat lunak untuk mengotomasi pemberian *feedback*. Problem pemrograman dapat berupa problem logik, yakni problem dalam ranah problem solving yang secara umum memerlukan kemampuan matematika dan teknis analisis tertentu. Kedua adalah problem teknis, yakni dalamnya adalah pemahaman terhadap sintaks, perilaku, serta kemampuan siswa dalam mengatasi situasi “panik”. terkait dengan perilaku pemrograman, termasuk instrument untuk pencatatan hasil evaluasi dan penilaian, untuk siswa dan pengajar melihat hasil pembelajaran, baik per tugas maupun keseluruhan kasus;

2.3. Analisis Requirement Perangkat Lunak

Analisis requirement perangkat lunak dilakukan sebagai langkah awal untuk melakukan menuju langkah berikutnya dalam perancangan perangkat lunak. Requirement perangkat lunak, diangkat dari hasil analisis aspek-aspek yang terkait, baik aspek teknis maupun non teknis.

Requirement yang dihasilkan meliputi :

- a. User Requirement
- b. Requirement fungsional sistem
- c. Requirement user interface

2.4. Analisis Bahan Pengajaran dan Bahan Evaluasi Pemrograman

Luaran dari penelitian sebelumnya tentang Metode Pengajaran Pemrograman dengan bantuan Sistem Automatic Grading[11], salah satunya berupa rumusan mengenai materi evaluasi yang dirancang dengan mengacu pada taxonomi kognitif dari Bloom. Luaran dari penelitian tersebut dimanfaatkan pada pengembangan perangkat lunak ini, sebagai materi penugasan yang harus dikerjakan siswa selama proses belajar pemrograman.

Rancangan mengenai pengajaran praktik pemrograman

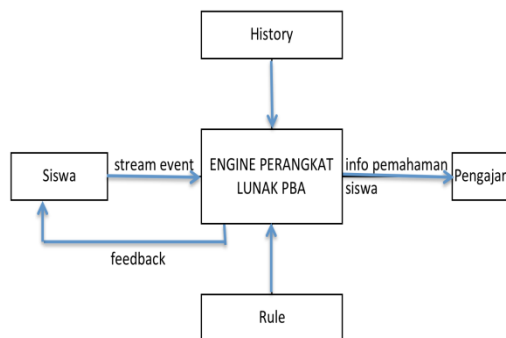
yang digunakan meliputi :

- proses pembelajaran praktik yang dijalankan (jumlah pertemuan, materi pada setiap pertemuan dan lain-lain);
- rumusan mengenai materi pengajaran terhadap level pembelajaran menurut taxonomi Bloom;
- soal-soal penugasan untuk setiap materi, menurut acuan tertentu, beserta kriteria penilaian yang akan diterapkan;

3. HASIL DAN PEMBAHASAN

3.1. Model Diagram PBA yang akan Dijalankan

PBA yang akan dijalankan merupakan sebuah siklus interaksi antara siswa, pengajar dan sistem perangkat lunak. Gambar 1 memperlihatkan model interaksi tersebut, yang terdiri atas beberapa komponen yaitu siswa, pengajar, history (data aktivitas siswa memrogram), dan rule.



Gambar 1 merupakan model global perangkat lunak yang akan dibuat.

3.2. Requirement untuk Otomasi PBA

Dari analisis yang telah dilakukan, berikut adalah *requirement* yang didefinisikan

3.3. User Requirement

- User* dalam hal ini didefinisikan sebagai pengajar yang akan memanfaatkan semua informasi aktivitas pemrograman siswa untuk kebutuhannya dalam menjalankan fungsinya sebagai pengajar.
- User* memerlukan informasi tentang perilaku siswa dalam memrogram
- User* memerlukan media untuk melakukan pemantauan terhadap siswa selama aktivitas pemrograman berlangsung *User* memerlukan media untuk melakukan “komunikasi” dengan siswa selama aktivitas pemrograman berlangsung
- User* memerlukan media untuk memantau perolehan nilai siswa baik nilai latihan harian, maupun nilai evaluasi (quiz), UTS maupun UAS.

3.4. Pendefinisian Requirement Perangkat Lunak

Requirement fungsional sistem diperoleh dari hasil analisis terhadap problem-problem siswa serta kebutuhan *user*.

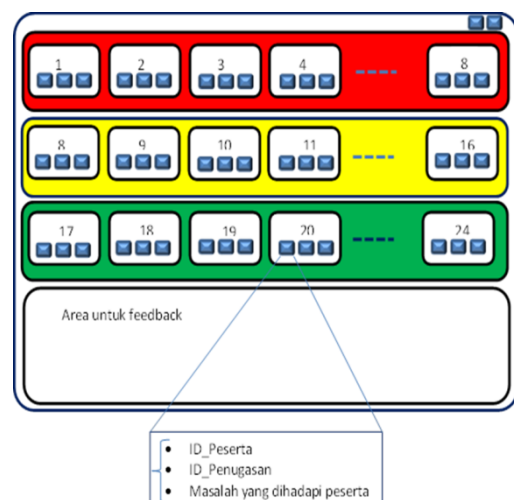
Berikut rumusan *requirement* fungsional sistem perangkat lunak yang akan dikembangkan :

- Sistem dapat menginformasikan proses mikro dari suatu aktivitas pemrograman. Proses mikro (*micro-process*) yang dimaksud adalah berupa *event-event* yang terjadi selama siswa beraktivitas, terdiri dari [4],[12]:
 - create file/program
 - Editing (insert, edit, copy-paste, cut, move, delete)

- Save file
 - Proses kompilasi
 - Keluar masuk lingkungan IDE
 - Penekanan key-stroke lain yang tidak terkait dengan aktivitas pemrograman, tetapi sering dilakukan oleh siswa
- Sistem dapat mengidentifikasi siswa yang sedang masuk ke sistem dan melakukan aktivitas pemrograman
 - Sistem dapat menyimpan informasi mengenai micro-process dari suatu aktivitas pemrograman, setiap siswa.
 - Sistem dapat memberikan *feedback* berdasarkan problem yang teridentifikasi. Jenis dan karakteristik *feedback* yang diberikan, akan dirumuskan setelah kegiatan uji coba tahun pertama dijalankan, yakni dengan merekam aktivitas riil pengajar dalam memberikan *feedback* kepada siswa
 - Sistem dapat menampilkan secara *real time* (*event-streaming*) aktivitas memrogram siswa, dalam pengajaran formal.

3.5. Requirement User Interface

- User Interface* (UI) untuk memantau aktivitas siswa selama proses pembelajaran formal praktikum. UI ini akan memberi “tempat khusus” bagi siswa yang secara khusus akan dipantau (siswa yang menjadi sample)
- UI untuk menampilkan *feedback*
- UI untuk pengajar mengintervensi siswa yang memiliki problem (lihat Gambar 2)



Gambar 2. Model GUI PBA Dalam Pembelajaran Pemrograman Dasar

3.6. Hasil Eksplorasi Teknologi untuk Merekam Micro-Process

Teknologi yang digunakan untuk merekam *micro-process* diputuskan Eclipse, yaitu sebuah *tools open source* yang dapat berfungsi sebagai IDE (*Integrator Development Environ-ment*), sehingga aktivitas

pemrograman dapat dilakukan menggunakan Eclipse. Selain sebagai IDE, Eclipse memungkinkan untuk dilakukan Plugin modul-modul dari luar.

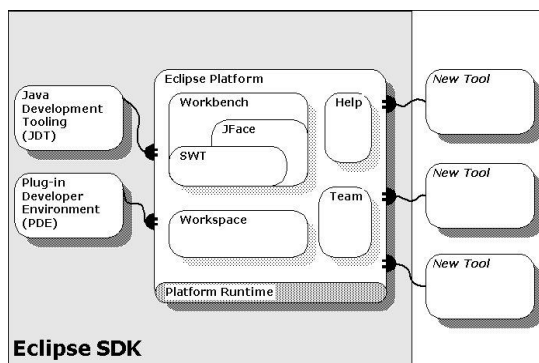
Eclipse adalah platform yang dirancang dari dasar untuk membangun tool pengembangan perangkat lunak. Secara disain, platform tidak menyediakan banyak fungsi untuk programmer aplikasi. Melainkan untuk pengembang tool. Pengembang *tools* dapat menambahkan berbagai hal terhadap platform melalui komponen yang dinamakan dengan plug-ins.

Eclipse menyediakan model umum antarmuka pengguna (UI) untuk bekerja dengan *tools*. Inti dari Eclipse adalah sebuah arsitektur untuk penemuan dinamis, pemuatan, dan menjalankan plug-in. Platform ini menangani logistik untuk menemukan dan menjalankan kode yang tepat. Platform UI menyediakan pengguna model navigasi standar. Dengan ini, plug-in dapat fokus terhadap apa yang secara substansi akan dikerjakan.

Platform Eclipse mendefinisikan arsitektur terbuka sehingga setiap tim pengembangan plug-in dapat fokus pada bidang keahlian mereka

3.7. Arsitektur Platform

Platform Eclipse itu sendiri disusun sebagai subsistem yang dilaksanakan dalam satu atau lebih plug-in (lihat Gambar 3.). Subsistem-subsistem ini dibangun di atas Platform *runtime*.



Gambar 3. Arsitektur Eclipse

Plug-in yang membentuk subsistem menentukan titik ekstensi untuk menambahkan perilaku ke platform. Tabel 2 menjelaskan komponen runtime utama dari platform yang diimplementasikan sebagai satu atau lebih plug-in.

Dalam kaitan ini, *event-event micro-process* pada Eclipse yang akan digunakan untuk merekam *micro-process* pemrograman meliputi hal-hal berikut :

- Identifikasi user name
- Identifikasi nama project
- Aktivasi perubahan kode
- Waktu Aktivasi
- Ada aktivitas terhadap editor. Membuka, hingga menutup editor
- Pengaturan Teks
- Penyalinan Teks (copy teks)
- Menghapus / memotong teks
- Perubahan kode
- Perubahan pada Windows
- Identifikasi Windows yang aktif
- Identifikasi Aktivasi Eclipse (Windows Eclipse tidak aktif)
- Jumlah error dan informasi warning
- Menutup Eclipse
- Membuka bagian selain editor

- Menutup file Melakukan Save (menyimpan data)

Komponen	Penjelasan
Platform runtime	Mendefinisikan model titik ekstensi dan plug-in. Dan secara dinamis menemukan plug-in dan menangani informasi mengenai plug-in dan ekstensi poin dalam registri platform. Plug-in yang dimulai ketika diperlukan sesuai dengan operasi pengguna dari platform. Runtime ini dilaksanakan menggunakan kerangka OSGi.
Resource management (workspace)	Mendefinisikan API untuk menciptakan dan mengelola sumber daya (proyek, file, dan folder) yang dihasilkan oleh Tool dan disimpan dalam sistem file.
Workbench UI	Mengimplementasikan kokpit pengguna untuk menavigasi platform. Ini mendefinisikan poin ekstensi untuk menambahkan komponen UI seperti view atau tindakan menu. Ini memasok toolkit tambahan (JFace SWT dan) untuk membangun antarmuka pengguna. Layanan UI yang terstruktur sehingga subset dari UI plug-in dapat digunakan untuk membangun aplikasi rich clien yang independen terhadap pengelolaan sumber daya dan model workspace. IDE-centric plug-ins mendefinisikan fungsi tambahan untuk navigasi dan memanipulasi sumber daya.
Help system	Mendefinisikan poin ekstensi untuk plug-in untuk memberikan bantuan atau dokumentasi lain dalam bentuk browsable book.
Team support	Mendefinisikan model pemrograman tim untuk mengelola dan versioning sumber daya.
Debug support	Mendefinisikan model debug yang independen terhadap bahasa dan UI kelas untuk membangun debugger dan peluncur.
Other utilities	Plug-in utilitas lain yang memberikan fungsi seperti mencari dan membandingkan sumber daya, melakukan pembangunan menggunakan file konfigurasi XML, dan secara dinamis memperbarui platform dari server.

Tabel 2 . Komponen Runtime untuk Plug-In

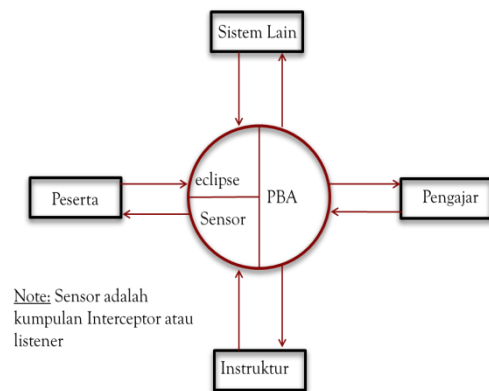
Uraian mengenai *micro-process*, serta kaitannya dengan aktivitas pemrograman dapat dijelaskan seperti berikut :

- Setiap aktifitas *cursor mouse* dan *menu* yang dipilih dalam Eclipse, perlu direkam yang menginformasikan bagian mana pada Eclipse yang sedang aktif. Contoh : ketika user mengakses bagian *project management*, *problem info*, *outlile* atau *task list*.
- Setiap aktifitas mengetik perlu direkam :
 - Proses *save Log* perlu dilakukan setiap selang waktu tertentu, misalnya setiap 3 detik, setelah ada perubahan dalam *text* program.
 - Log yang tersimpan adalah gabungan dari seluruh karakter dalam program yang disusun dalam sebuah *string*.

iii. Log hanya berisi *plain text* atau bisa dikatakan informasi murni , serupa dengan apa yang ada di dalam *text* program secara keseluruhan. Tidak berisi aktifitas *push keyboard*, contoh : melakukan *backspace*, *delete*, *pageup*, *down* dan lain-lain.

- Setiap proses *saving* perlu didokumentasikan dalam sebuah Log.
- Setiap Dialog Box didokumentasikan dalam Log. Begitupun *response* dari *user*. Informasi ini berjenis *<dialogname>*

Seluruh aktifitas umum, diantaranya : *compile*, *clear*, *build* dan lain-lain. Akan dapat didokumentasikan secara langsung. Setelah digabungkan, didapatkan model seperti pada Gambar 4.



Gambar 4. Model Konteks PBA Dalam Pembelajaran Pemrograman Dasar

Gambar 4, menunjukkan bahwa perangkat lunak PBA yang dikembangkan akan diintegrasikan dengan Eclipse.

4. KESIMPULAN

Berdasarkan analisis dan eksplorasi teknologi, berikut adalah rumusan *requirement* fungsional sistem perangkat lunak yang diperlukan untuk menjalankan pemrograman PBA:

- Sistem dapat menginformasikan proses mikro dari suatu aktivitas pemrograman, misalnya create file/program, Editing(insert, edit, copy-paste, cut, move, delete) file
- Sistem dapat mengidentifikasi siswa yang sedang masuk ke sistem dan melakukan aktivitas pemrograman
- Sistem dapat menyimpan informasi mengenai *micro-process* dari suatu aktivitas pemrograman, setiap siswa.
- Sistem dapat memberikan *feedback* berdasarkan problem yang teridentifikasi. Jenis dan karakteristik *feedback* yang diberikan, akan dirumuskan setelah kegiatan uji coba tahun pertama dijalankan, yakni

dengan merekam aktivitas riil pengajar dalam memberikan feedback kepada siswa

- e. Sistem dapat menampilkan secara *real time* (event-streaming) aktivitas memprogram siswa, dalam pengajaran formal.

Sementara berdasarkan kegiatan uji coba *automatic grading* untuk menjalankan PBA ditemukan bahwa ada sembilan kelompok masalah pemrograman yang dihadapi peserta, yaitu:

- a. Kesalahan sintaks
- b. Ketelitian dalam test case
- c. Algoritma
- d. Kesulitan dalam penggunaan tools
- e. Kesalahan dalam menggunakan type data
- f. Kesalahan dalam menampilkan output
- g. Tidak mengetahui rumus/model matematik
- h. Tidak mengetahui konsep dalam bahasa pemrograman yang digunakan
- i. Lain-lain

Secara teknis, kelompok pertanyaan dari peserta tersebut adalah sebagai berikut:

- a. Kompilasi sukses namun *output error*
- b. Sintaks *error*
- c. Salah implementasi rumus
- d. Sudah diperbaiki, namun tetap *error* (tidak di-save setelah edit)

Tidak memahami perintah penugasan sehingga output jadi berbeda.

4. DAFTAR PUSTAKA

- [1] Liem, I., "*Aspek Pedagogi Pengajaran Pemrograman Pertama*", Depdiknas-RI., 2004.--6
- [2] Patil, A. *Automatic Grading of Programming Assignments*. Master's Projects. Paper 51. San Jose State University, 2010.--7
- [3] Gomes, A and Mendes, AJ., "*Studies and Proposals about Initial Programming Learning*" : 40th ASEE/IEEE Frontiers in Education Conference S3F-1, 2010. --3
- [4] Teague, D., "*Pedagogi of Computer Programming*". *Thesis of Master of Information Technology (Research)*. Queensland University of Technology, 2011.--11
- [5] Pinchok, N and Brandt, W.P., "*Connecting Formative Assessment, Research and Practice. An Introduction Guide for Educator*". Learning Point Associates, 2009.--8
- [6] Cordiner, M. "*Guidlines for Good Assessment Practices*". Center for the Advancement of Learning and Teaching, University of Tasmania, 2011. --1
- [7] Truog, N., Roe, P., and Peter, B.P., *Static Analysis of Students' Java Programs*". Proceeding. *The Sixth Australian Computing Education Conference (ACE2004)*, 2004--12
- [8] Schlesinger, F dan Jekutsch, S. , "*ElectroCodeGram : An Environment for Studying*". Freie Universit"at Berlin, Institut f. Informatik, Takustr. 9, 14195 Berlin, Germany, 2008.--10
- [9] Patroumpas, K dan Sellis, T., "*Event Processing and Real Time Monitoring over Streaming Traffic Data*". School of Electrical and Computer Engineering National Technical University of Athens, Hellas, 2011.--9
- [10] Edwards, S. H. *Teaching Software Testing: Automatic Grading Meets Test-first Coding*. Virginia Tech, Dept. of Computer Science: s.n.2003.--2
- [11] Harris, JA., Elizabeth S. Adams,. dan Harris, NA. 2003. "*Making Program Grading Easier (but not Totally Automatic)*". A paper at James Madison University. --4
- [12] Jimmoyiannis, A. Using SOLO Taxonomy to Explore Student's Mental Model of The Programming Variable and the Assignment Statement. Theme in Science & Technology Education, 4(2), 53-74, 2011.--5