Media Informatika, Vol. 3 No. 1, Juni 2005, 11-23
ISSN: 0854-4743

DARI KONSTRUKSI PEMROGRAMAN BPEL KE NOTASI GRAFIS ISDL

Teduh Dirgahayu
Jurusan Teknik Informatika, Fakultas Teknologi Industri, Universitas Islam Indonesia
Jl. Kaliurang Km. 14 Yogyakarta 55501
Telp. (0274) 895287 ext. 122, Faks. (0274) 895007 ext. 148
E-mail: teduh.dirgahayu@fti.uii.ac.id

ABSTRAK

Tulisan ini menyajikan hasil pemetaan beberapa konstruksi pemrograman BPEL (Business Process Execution Language for Web Services) ke notasi grafis ISDL (Interaction Systems Design Language). Pemetaan telah mencakup baik konstruksi pemrograman umum maupun konstruksi pemrograman khusus BPEL. Konstruksi-konstruksi pemrograman tersebut adalah penugasan, pengambilan keputusan, perulangan, pengurutan eksekusi, penerimaan permintaan, pengembalian hasil, permintaan layanan, penantian, dan eksekusi paralel. Dengan pengetahuan hasil pemetaan ini, suatu model ISDL yang bebas teknologi implementasi dapat lebih dirinci untuk diimplementasikan pada teknologi Web services dengan BPEL sebagai bahasa pemrogramannya.

Kata kunci: *konstruksi pemrograman, notasi grafis, pemetaan, BPEL, ISDL*

1. PENDAHULUAN

Web services adalah salah satu teknologi pendukung komputasi berorientasi layanan. Dengan teknologi ini, suatu penyedia layanan dapat menyediakan layanan-layanan di web yang siap dipakai oleh berbagai aplikasi. Contoh-contoh layanan web antara lain: layanan pembayaran dengan kartu kredit, layanan pemesanan barang di toko maya, layanan pencarian informasi dengan mesin pencari, dan layanan prakiraan cuaca. Layanan-layanan yang disediakan biasanya relatif bebas konteks sehingga satu layanan dapat digunakan oleh banyak aplikasi. Teknologi *Web services* telah pula digunakan untuk penyediaan layanan komputasi ilmiah yang kerap disebut komputasi Grid (BEA System et al., 2003).

Sebelum dapat menggunakan suatu layanan web, pengembang aplikasi perlu mengetahui antarmuka pemrograman aplikasi dan protokol untuk mengakses layanan tersebut (Alonso et al., 2004). Kemudian pengembang aplikasi bisa membangun aplikasi dengan memakai teknologi dan bahasa pemrograman pilihannya. Si penyedia layanan bisa menyediakan layanannya juga memakai teknologi dan bahasa pemrograman pilihannya selama antarmuka pemrograman aplikasi dan protokol layanan web tersebut dipatuhi.

Salah satu bahasa pemrograman untuk membangun layanan berteknologi *Web services* adalah BPEL4WS (*Business Process Execution Language for Web Services*) (Quartel, 1998). BPEL4WS sering singkat disebut BPEL. Eksekusi program BPEL

dinamakan *proses bisnis*, atau cukup disebut *proses*. Sebagaimana bahasa-bahasa pemrograman lain, BPEL mempunyai konstruksi-konstruksi pemrograman umum dan konstruksi-konstruksi pemrograman khusus sesuai dengan tujuan keberadaannya. BPEL bertujuan menyediakan suatu layanan web yang berinteraksi dengan layanan-layanan web lain yang telah tersedia terlebih dahulu. Suatu contoh layanan web yang dapat dikembangkan dengan BPEL adalah layanan pemesanan jasa wisata yang merupakan hasil interaksi dengan layanan pemesanan hotel, layanan pemesanan tiket perjalanan, dan layanan pembayaran dengan kartu kredit.

Langkah awal dalam pengembangan suatu sistem, termasuk layanan web, adalah pembuatan model yang mewakili struktur dan fungsi sistem tujuan. Model dinyatakan dalam suatu bahasa pemodelan, yang bisa menggunakan notasi teks ataupun notasi grafis. Notasi grafis lazim disukai karena lebih mudah dipahami; sedangkan notasi teks menjamin bahwa suatu model bertafsiran tunggal. ISDL (*Interaction Systems Design Language*) (Foster dan Kesselman, 2003) adalah salah satu bahasa pemodelan yang mempunyai notasi teks dan notasi grafis.

Tulisan ini menyajikan hasil pemetaaan beberapa konstruksi pemrograman BPEL ke notasi grafis ISDL. Pemetaan ini perlu untuk membentuk suatu pengetahuan implementasi yang bermanfaat dalam pemodelan suatu layanan web. Dengan pengetahuan ini, suatu layanan dapat dimodelkan dengan notasi grafis ISDL untuk kemudian diimplementasikan dengan BPEL. Pengetahuan ini membuat pemodel layanan web dapat mempertimbangkan efisiensi dan kemudahan implementasi. Manfaat lebih jauh ialah bahwa program BPEL dapat dibangkitkan secara otomatis dari model bernotasi grafis ISDL.

Bagian 2 memberikan beberapa konstruksi pemrograman BPEL yang hasil pemetaannya disajikan dalam tulisan ini. Karena keterbatasan ruang penulisan, tidak semua hasil pemetaan disajikan. Bagian 3 membahas beberapa konsep dasar dan notasi grafis ISDL. Bagian 3.1 menyajikan hasil pemetaan beberapa konstruksi pemrograman BPEL ke notasi grafis ISDL. Bagian 4 menutup tulisan ini dengan suatu simpulan umum.

2. BPEL

BPEL ialah suatu bahasa pemrograman berdasar XML (*Extended Markup Language*) (W3C, 2004). Sebagai suatu bahasa pemrograman, BPEL mempunyai dua macam konstruksi pemrograman, yakni konstruksi pemrograman umum dan konstruksi pemrograman khusus sesuai tujuan keberadaannya. Konstruksi-konstruksi pemrograman BPEL dinamakan *aktivitas*. Setiap aktivitas dinyatakan sebagai suatu elemen XML yang dapat mengandung satu atau lebih atribut elemen.

Konstruksi pemrograman umum adalah konstruksi pemrograman yang juga terdapat pada bahasa-bahasa pemrograman lain. Konstruksi atau aktivitas pemrograman umum tersebut adalah:

- **Penugasan**, dinyatakan dengan elemen *assign*.
  Aktivitas ini memberikan suatu nilai kepada suatu variabel. Nilai dapat berupa suatu nilai literal atau suatu nilai pada variabel lain. Sebagai bahasa

yang berdasar XML, BPEL dapat mengakses elemen dan atribut XML menggunakan XPath (W3C, 1999).

- **Pengambilan keputusan**, dinyatakan dengan elemen *switch*.
  Aktivitas ini memilih satu dari beberapa pilihan percabangan apabila syarat salah satu cabang terpenuhi. Untuk menghindari kebuntuan, BPEL menyediakan suatu pilihan *default* yang akan diambil jika semua syarat cabang tidak bisa terpenuhi.
- **Perulangan**, dinyatakan dengan elemen *while*.
  Aktivitas ini mengerjakan ulang satu atau lebih aktivitas lain selama syarat perulangan terpenuhi.
- **Pengurutan eksekusi**, dinyatakan dengan elemen *sequence*.
  Aktivitas ini menyatakan secara eksplisit urutan eksekusi aktivitas-aktivitas lain. Pada beberapa bahasa pemrograman lain, konstruksi semacam ini biasanya dinyatakan secara implisit, yaitu berdasarkan urutan pernyataan konstruksi.

Konstruksi atau aktivitas khusus BPEL adalah aktivitas yang berkenaan dengan penyediaan layanan atau permintaan penggunaan layanan lain. Karena penggunaan layanan bisa berlangsung dalam jangka waktu yang lama, BPEL mempunyai suatu aktivitas yang berkenaan dengan pengelolaan waktu. Selain itu, BPEL mempunyai aktivitas untuk mengeksekusi beberapa aktivitas secara paralel dan menjaga sinkronisasi antara aktivitas-aktivitas tersebut. Aktivitas khusus BPEL tersebut adalah:

- **Penyediaan layanan**, dinyatakan dengan elemen *receive*.
  Aktivitas ini menyediakan layanan bagi pengguna. Aktivitas ini bertugas menerima permintaan penggunaan layanan. Dalam keadaan ini, eksekusi proses dihentikan hingga suatu permintaan datang dari pengguna. Ketika ada permintaan, eksekusi proses dilanjutkan lagi.
- **Pengembalian hasil**, dinyatakan dengan elemen *reply*.
  Aktivitas ini mengembalikan hasil proses kepada pengguna layanan. Susunan aktivitas *receive* yang diakhiri dengan aktivitas *reply* membentuk operasi permintaan-tanggapan (*request-response*) sinkron.
- **Permintaan layanan**, dinyatakan dengan elemen *invoke*.
  Aktivitas ini meminta layanan web lain. Aktivitas ini bisa membentuk operasi satu-arah (*one-way*) asinkron ataupun operasi permintaan-tanggapan sinkron.
- **Penantian**, dinyatakan dengan elemen *wait*.
  Aktivitas ini menunggu selama rentang waktu tertentu atau hingga suatu waktu terlampaui.
- **Eksekusi paralel**, dinyatakan dengan elemen *flow*.
  Aktivitas ini mengeksekusi beberapa aktivitas sekaligus. Sinkronisasi antara aktivitas-aktivitas tersebut dapat dilakukan dengan menggunakan sub-elemen *link*.

Selain aktivitas-aktivitas yang telah disebutkan di atas, BPEL masih mempunyai beberapa aktivitas lain, yaitu: *terminate*, *empty*, *pick*, *throw*, *scope*, dan *compensate*. BPEL juga mempunyai beberapa konstruksi lain yang tidak

digolongkan sebagai aktivitas, antara lain: *correlation*, *compensation handler*, *fault handler*, dan *event handler*. Pemetaan aktivitas-aktivitas dan konstruksi-konstruksi tersebut tidak disajikan dalam tulisan ini.

3. ISDL

ISDL adalah bahasa pemodelan struktur dan fungsi sistem terdistribusi. ISDL terdiri dari tiga elemen, yakni: model entitas, model perilaku, dan hubungan penugasan (Foster dan Kesselman, 2003). Model entitas menyatakan bagian-bagian penyusun sistem beserta hubungan antara mereka. Model perilaku mewakili fungsi bagian-bagian pembentuk sistem beserta interaksi antara mereka. Hubungan penugasan memetakan hubungan antara model entitas dan model perilaku.

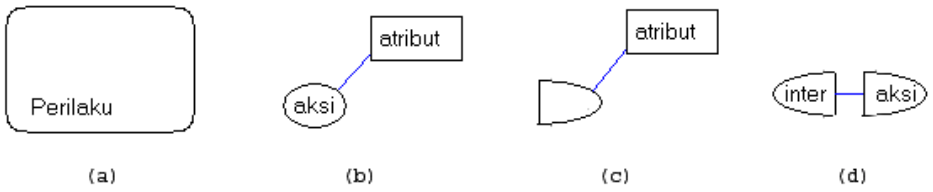
Tulisan ini hanya membahas beberapa konsep pemodelan perilaku yang berkenaan dengan pemetaan konstruksi pemrograman BPEL ke notasi grafis ISDL. Notasi tekstual yang diwakili oleh notasi grafis tidak disertakan. Pembaca yang berkeinginan untuk mempelajari ISDL lebih lanjut dapat mengacu ke Foster dan Kesselman (2003), ISDL Home (n.d.), dan Visser et al. (2002).

Beberapa konsep dasar dalam model perilaku dipaparkan sebagai berikut. Perhatikan bahwa penggunaan istilah *aktivitas* pada bagian ini tidak mengacu ke aktivitas BPEL, tetapi ke aktivitas secara umum.

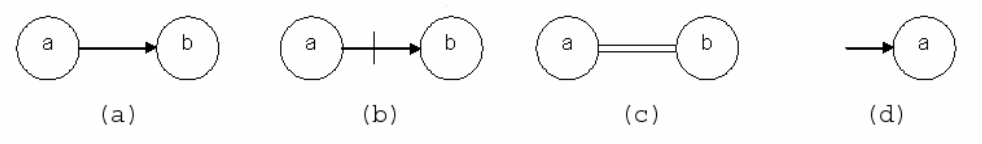
- **Perilaku**
  Perilaku menyatakan eksekusi fungsi yang dimodelkan. Perilaku dapat berupa *perilaku terstruktur*, yaitu perilaku yang terdiri dari beberapa sub-perilaku, atau *perilaku monolitik*, yaitu perilaku yang mendefinisikan perilaku dasar yang tidak tersusun dari sub-perilaku. Notasi grafis perilaku adalah suatu segiempat bersudut lengkung dengan nama perilaku dituliskan di dalam segiempat tersebut. Notasi ini disajikan pada Gambar 1(a).
- **Aksi**
  Aksi menyatakan suatu aktivitas yang dieksekusi secara utuh. Aksi selalu berada di dalam suatu perilaku. Tiap aksi adalah atomik, yang berarti bahwa suatu aksi adalah aktivitas tunggal yang tidak dapat dibagi menjadi satuan-satuan yang lebih kecil. Suatu aksi hanya dapat dieksekusi secara utuh atau tidak sama sekali. Selain itu, aksi tidak pernah memberikan hasil sementara aktivitas yang sedang dilakukannya. Notasi grafis aksi adalah suatu elips dengan nama aksi dapat diberikan di dalam elips tersebut. Notasi ini disajikan pada Gambar 1(b).
- **Interaksi**
  Interaksi menyatakan suatu aktivitas yang dieksekusi secara utuh sebagai kerjasama antara dua atau lebih *perilaku peserta*. Suatu interaksi terjadi hanya jika semua perilaku pesertanya terlibat. Seperti halnya aksi, interaksi hanya dapat dieksekusi secara utuh atau tidak sama sekali. Bagian interaksi yang ada di suatu perilaku peserta disebut *sumbangan interaksi*. Notasi grafis sumbangan interaksi adalah suatu elips yang terpotong. Suatu interaksi terdiri dari beberapa sumbangan interaksi yang saling terhubung. Notasi ini disajikan pada Gambar 1(c) dan (d).

- **Atribut**
  Atribut menampilkan informasi yang diperlukan atau dihasilkan oleh suatu aktivitas. Atribut dapat pula menyatakan waktu dan lokasi aktivitas. Atribut bisa mengandung kekang informasi, waktu, ataupun tempat. Notasi grafis atribut adalah suatu segiempat yang dihubungkan dengan suatu garis ke aktivitas yang dijelaskannya. Notasi ini disajikan pada Gambar 1(b) dan (c).
- **Relasi Kausal**
  Relasi kausal menyatakan syarat eksekusi suatu aktivitas. Atribut dapat diberikan pada relasi kausal sebagai kekang relasi yang ditempuh. Terdapat empat syarat dasar relasi kausal sebagai berikut. Perhatikan bahwa pada syarat-syarat berikut, aksi dipakai untuk mewakili baik aksi maupun interaksi.
  - *membolehkan*. Eksekusi suatu aksi membolehkan suatu aksi lain dieksekusi.
  - *mencegah*. Eksekusi suatu aksi mencegah suatu aksi lain dieksekusi.
  - *sinkronisasi*. Dua atau lebih aksi harus dieksekusi dalam waktu bersamaan.
  - *mulai*. Suatu aksi dapat selalu dieksekusi.

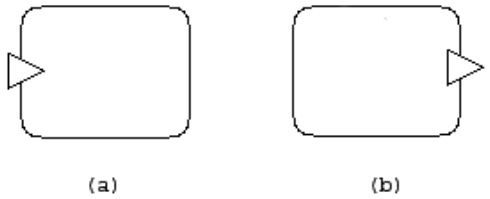
  Notasi keempat syarat dasar tersebut disajikan pada Gambar 2.
- **Titik Masuk**
  Titik masuk menyatakan awal suatu perilaku dieksekusi. Apabila suatu perilaku membutuhkan informasi tertentu, maka informasi tersebut dapat dilewatkan sebagai atribut titik masuk. Notasi grafis titik masuk adalah suatu segitiga pada tepi segiempat notasi perilaku dengan salah satu sudutnya menunjuk ke dalam segiempat. Notasi ini disajikan pada Gambar 3(a).
- **Titik Keluar**
  Titik keluar menyatakan akhir suatu perilaku. Apabila perilaku tersebut ingin melewatkan informasi bagi perilaku lain, maka informasi tersebut dapat dilewatkan sebagai atribut titik keluar. Perlu diketahui bahwa suatu perilaku dapat berlangsung terus-menerus sehingga tidak mempunyai titik keluar. Suatu perilaku dapat berakhir tanpa menyatakan titik keluar, yaitu setelah eksekusi aktivitas terakhir dalam perilaku tersebut. Notasi grafis titik keluar adalah suatu segitiga pada tepi segiempat notasi perilaku dengan salah satu sudutnya menunjuk ke luar segiempat. Notasi ini disajikan pada Gambar 3(b).



Gambar 1. Notasi (a) perilaku, (b) aksi dengan atribut, (c) sumbangan interaksi dengan atribut, dan (d) interaksi



Gambar 2. Relasi kausal (a) membolehkan, (b) mencegah, (c) sinkronisasi, dan (d) mulai



Gambar 3. Notasi (a) titik masuk dan (b) titik keluar

3.1 Pemetaan dari BPEL ke ISDL

Perhatikan bahwa penggunaan istilah *aktivitas* pada bagian ini mengacu semata pada aktivitas BPEL. Istilah *atribut* mengacu pada atribut elemen XML, sedangkan atribut pada ISDL disebutkan sebagai *atribut aksi*, *atribut interaksi*, atau *atribut relasi kausal*.

BPEL mendefinisikan beberapa atribut baku bagi semua aktivitas, yaitu: *name*, *joinCondition*, dan *suppressJoinFailure*. Atribut-atribut tersebut boleh tidak disertakan. Atribut *name* menyatakan nama aktivitas. Penamaan aktivitas tidak mempengaruhi eksekusi proses, tetapi membuat program lebih mudah dimengerti. Atribut ini dipetakan ke ISDL sebagai nama aksi atau nama sumbangan interaksi. Dua atribut lainnya berkenaan dengan penanganan kegagalan eksekusi. Karena pemetaan penanganan kegagalan tidak dicakup dalam tulisan ini, kedua atribut tersebut tidak dibahas.

Suatu aktivitas dapat pula mengandung elemen baku, yakni *link,* untuk sinkronisasi antara aktivitas. Perhatikan bahwa konsep sinkronisasi di BPEL berbeda dengan konsep sinkronisasi di ISDL. Di ISDL, sinkron berarti dua atau lebih aksi atau interaksi dilakukan pada saat yang sama. Di BPEL, sinkron berarti suatu aktivitas *target* hanya boleh dieksekusi setelah aktivitas *source* dieksekusi. Dengan demikian, konsep sinkronisasi di BPEL dipetakan sebagai konsep relasi kausal *membolehkan*.

Berikut ini adalah hasil pemetaan aktivitas-aktivitas BPEL ke notasi grafis ISDL. Untuk kejelasan pemetaan, elemen XML aktivitas disajikan sebelum notasi grafis ISDL hasil pemetaan. Hasil pemetaaan dijelaskan secara singkat. Pada Subbagian 3.3, 3.4, 3.5, dan 3.10, aktivitas dinyatakan sebagai aksi untuk mewakili baik aksi maupun sumbangan interaksi.

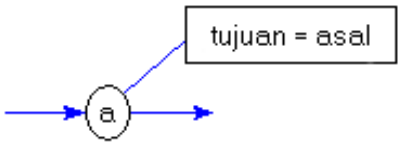
Perhatikan bahwa pada hasil pemetaan hampir selalu terdapat relasi kausal menuju atau meninggalkan suatu aksi. Semua relasi kausal yang menuju aksi tersebut diasumsikan sebagai relasi kausal *membolehkan* dari sembarang aksi,

sumbangan interaksi, atau titik keluar. Semua relasi kausal yang meninggalkan aksi tersebut juga diasumsikan sebagai relasi kausal *membolehkan* ke sembarang aksi, sumbangan interaksi, atau titik masuk.

3.2 Penugasan

```
<assign atribut-baku>
  elemen-baku
  <copy>+
    asal
    tujuan
</assign>
```

Tanda plus "+" berarti beberapa sub-elemen dapat dideklarasikan sekaligus dalam satu elemen. Hasil pemetaan diperlihatkan pada Gambar 4. Diandaikan bahwa atribut `name` memberi nama *a* ke suatu aksi. Jika terdapat beberapa elemen `copy` dalam satu elemen `assign`, maka terdapat beberapa penugasan pada atribut aksi.

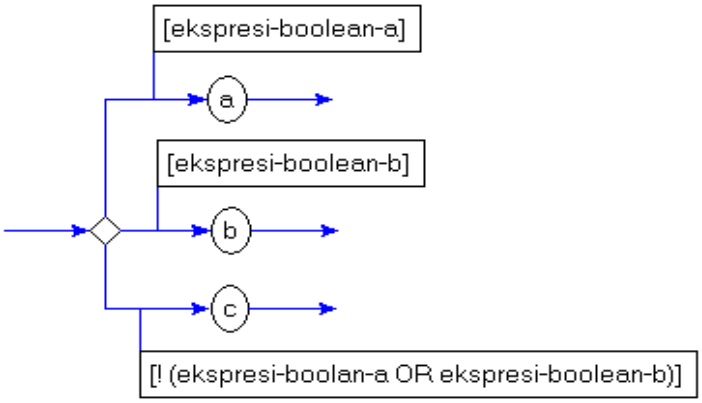


Gambar 4. Notasi ISDL untuk aktivitas penugasan

3.3 Pengambilan Keputusan

```
<switch atribut-baku>
  element-baku
  <case condition="ekspresi-boolean">+
    aktivitas
  </case>
  <otherwise>
    aktivitas
  </otherwise>
</switch>
```

Hasil pemetaan diperlihatkan pada Gambar 5. Diandaikan terdapat dua element `case` yang masing-masing mengandung aktivitas berupa aksi *a* dan *b*, masing-masing dengan syarat *ekspresi-boolean-a* dan *ekspresi-boolean-b* pada atribut `condition`. Diandaikan pula terdapat aktivitas berupa aksi *c* pada elemen `otherwise`. Salah satu cabang akan ditempuh apabila kekang relasi kausal terpenuhi.



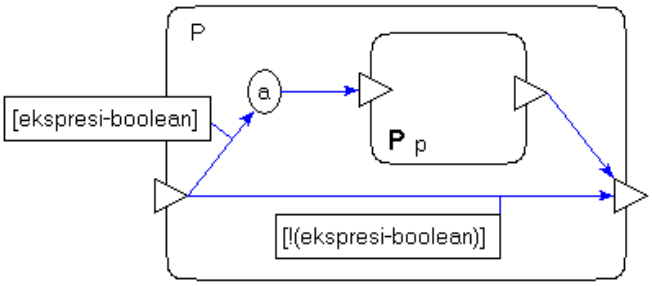
Gambar 5. Notasi ISDL untuk aktivitas pengambilan keputusan

Notasi belahketupat adalah notasi pintas yang merangkum beberapa relasi kausal hingga pada suatu titik. Pada titik itu, relasi-relasi yang terangkum terurai ke tujuan masing-masing sesuai kekang relasi kausal. Tanpa notasi pintas ini, tiap relasi kausal harus digambarkan dari asal hingga ke tujuannya. Penggunaan notasi pintas ini bermanfaat untuk mengurangi keruwetan dan mempermudah pemahaman terhadap model.

3.4 Perulangan

```
<while condition="ekspresi-boolean" atribut-standar>
  elemen-baku
  aktivitas
</while>
```

Hasil pemetaan diperlihatkan pada Gambar 6. Diandaikan atribut `name` memberi nama *P* kepada perilaku. Diandaikan pula aksi *a* adalah aktivitas yang akan dieksekusi berulang. Selama syarat *ekspresi-boolean* pada atribut `condition` terpenuhi, yang berarti kekang relasi kausal ke arah aksi *a* terpenuhi, perilaku *P* dieksekusi secara rekursif.



Gambar 6. Notasi ISDL untuk aktivitas perulangan

3.5 Pengurutan Eksekusi

```
<sequence atribut-baku>
  elemen-baku
  aktivitas+
</sequence>
```

Hasil pemetaan diperlihatkan pada Gambar 7. Diandaikan `aktivitas+` mendefinisikan dua aktivitas, yakni aksi *a* dan *b*, yang akan dieksekusi secara sekuensial. Urutan pernyataan aktivitas menyatakan urutan eksekusi aksi-aksi tersebut.

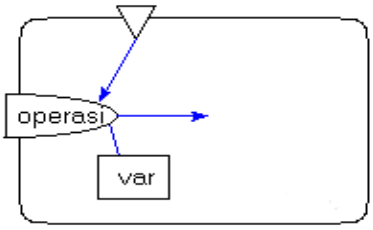
Gambar 7. Notasi ISDL untuk aktivitas pengurutan eksekusi

3.6 Penyediaan Layanan

```
<receive partnerLink="plink" portType="port"
         operation="operasi" variable="var"?
         createInstance="yes|no"?
  atribut-baku>
  elemen-baku
</receive>
```

Tanda tanya "?" berarti atribut boleh tidak digunakan. Garis vertikal "|" berarti satu nilai atribut harus dipilih dari nilai-nilai yang dipisahkan oleh garis tersebut. Hasil pemetaan diperlihatkan pada Gambar 8. Atribut `partnerLink` dan `portType` tidak dapat dipetakan ke notasi grafis ISDL. Keduanya merupakan informasi yang tidak menentukan bagaimana suatu program dieksekusi, tetapi merupakan informasi yang berkenaan dengan rekan dalam berinteraksi. Sumbangan interaksi dapat diberi nama sesuai operasi yang dilakukan.

Apabila atribut `createInstance` bernilai `yes`, yang berarti aktivitas ini adalah aktivitas pertama yang memulai eksekusi proses, maka harus ada relasi kausal *membolehkan* dari titik masuk ke sumbangan interaksi yang mewakili aktivitas ini. Apabila bernilai `no`, maka tidak boleh ada relasi kausal dari titik masuk ke sumbangan interaksi tersebut. Nilai atribut `variable` diwakili oleh informasi pada atribut interaksi.

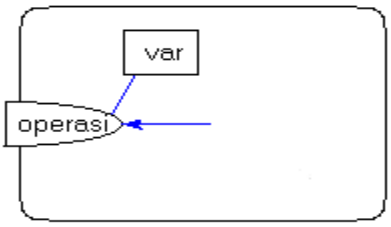


Gambar 8. Notasi ISDL untuk aktivitas penyediaan layanan

3.7 Pengembalian Hasil

```
<reply partnerLink="plink" portType="port"
       operation="operasi" variable="var"?
       faultName="kegagalan"?
  atribut-baku>
  element-baku
</reply>
```

Hasil pemetaan diperlihatkan pada Gambar 9. Sebagaimana pada aktivitas penyediaan layanan, atribut `partnerLink` dan `portType` tidak dapat dipetakan ke notasi grafis ISDL. Karena pemetaan tidak mencakup penanganan kegagalan, atribut `faultName` tidak dibahas. Sumbangan interaksi diberi nama sesuai operasi yang dilakukan. Nilai atribut `variable` diwakili oleh informasi pada atribut interaksi.



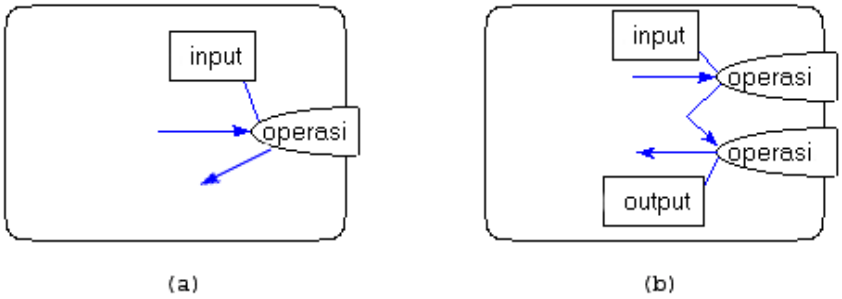
Gambar 9. Notasi ISDL untuk aktivitas pengembalian hasil

3.8 Permintaan Layanan

```
<invoke partnerLink="plink" portType="port"
        operation="operasi"
        inputVariable="input"? outputVariable="output"?
  atribut-baku>
  element-baku
</invoke>
```

Hasil pemetaan diperlihatkan pada Gambar 10. Sebagaimana pada aktivitas penyediaan layanan, atribut `partnerLink` dan `portType` tidak dapat dipetakan ke notasi grafis ISDL.

Terdapat dua macam layanan yang dapat diminta, yaitu layanan asinkron dan layanan sinkron. Pada permintaan layanan asinkron, eksekusi proses berlanjut setelah permintaan dikirimkan; sedangkan pada permintaan layanan sinkron, eksekusi proses dihentikan hingga hasil permintaan diberikan oleh penyedia layanan. Dengan demikian, ada dua bentuk pemetaan aktivitas ini ke notasi grafis ISDL.



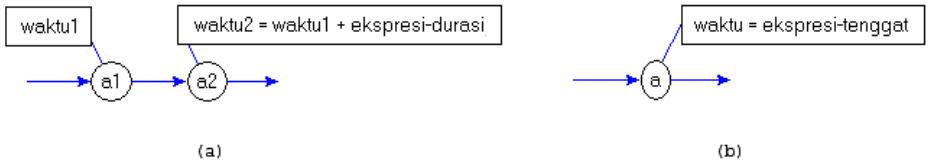
Gambar 10. Notasi ISDL untuk aktivitas permintaan layanan (a) asinkron dan (b) sinkron

Macam layanan yang diminta diketahui dari penggunaan atribut `outputVariable`. Jika atribut ini digunakan, maka layanan yang diminta adalah layanan sinkron dan hasil layanan yang diminta tersebut ditempatkan pada variabel yang disebutkan pada atribut ini.

3.9 Penantian

```
<wait (for="ekspresi-durasi" | until="ekspresi-tenggat")
  atribut-baku>
  elemen-baku
</wait>
```

Aktivitas ini harus menggunakan salah satu dari atribut `for` atau `until`, tetapi tidak keduanya secara bersamaan. Dengan demikian, terdapat dua bentuk pemetaan ke notasi grafis ISDL. Hasil pemetaan diperlihatkan pada Gambar 11.



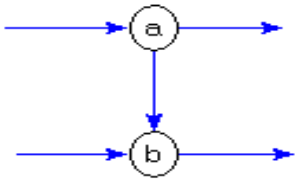
Gambar 11. Notasi ISDL untuk aktivitas penantian (a) beratribut for dan (b) beratribut until

Diandaikan atribut `name` memberi nama *a* pada aksi yang mewakili aktivitas ini. Pemodelan atribut `for` memerlukan dua aksi *a1* dan *a2*. Aksi *a1* mencatat waktu awal saat durasi mulai dihitung, sedangkan aksi *a2* menunggu hingga durasi waktu terlampaui. Kedua aksi tersebut memerinci aksi *a*. Pada pemodelan atribut `until`, aksi *a* dikerjakan setelah waktu tenggat terlampaui.

3.10 Eksekusi Paralel

```
<flow atribut-baku>
  elemen-baku
  <links>?
    <link name="nama-link">+
  </links>
  aktivitas+
</flow>
```

Hasil pemetaan diperlihatkan pada Gambar 12. Diandaikan `aktivitas+` mendefinisikan dua aksi *a* dan *b* yang akan dieksekusi secara paralel. Diandaikan pula ada sinkronisasi antara aksi *a* sebagai aktivitas *source* dan aksi *b* sebagai aktivitas *target*. Karena aktivitas *target* hanya dapat dikerjakan setelah aktivitas *source*, ada relasi kausal *membolehkan* dari aksi *a* ke aksi *b*.



Gambar 12. Notasi ISDL untuk aktivitas eksekusi paralel

4. SIMPULAN

Tulisan ini menyajikan hasil pemetaan beberapa konstruksi pemrogramaan BPEL ke notasi grafis ISDL. Hasil pemetaan tersebut merupakan suatu pengetahuan yang bermanfaat untuk mengimplementasikan model ISDL ke program BPEL. Pengetahuan ini juga dapat dipakai sebagai pendukung dalam pembangkitan program BPEL secara otomatis dari model ISDL.

Layanan-layanan web dapat dimodelkan menggunakan ISDL. Model awal suatu layanan, yang biasanya bebas teknologi implementasi, dirinci sebelum diimplementasikan pada teknologi dan bahasa pemrograman tertentu. Hasil pemetaan ini dapat digunakan sebagai acuan dalam memerinci model yang akan diimplementasikan pada teknologi *Web services* dengan BPEL sebagai bahasa pemrogramannya.

5. UCAPAN TERIMAKASIH

Penulis berterima kasih kepada Dr. Ir. Dick Quartel dan Ir. Remco Dijkman. Diskusi-diskusi yang penulis lakukan bersama mereka sangat bermanfaat dan memberi masukan bagi penyelesaian pemetaan ini. Penulis juga berterimakasih kepada Universitas Islam Indonesia atas kesempatan yang diberikan kepada penulis untuk melakukan penelitian ini di University of Twente, The Netherlands.

PUSTAKA

Alonso, G., Casati, F., Kuno, H., dan Machiraju, V. (2004). *Web Services: Concepts, Architectures and Applications*. Springer.

BEA Systems, IBM, Microsoft, SAP, dan Siebel System. (2003). *Business Process Execution Language for Web Services Version 1.1*. http://www-106.ibm.com/developerworks/webservices/library/ws-bpel/.

Foster, I., dan Kesselman, C., eds. (2003). *The Grid 2: Blueprint for a New Computing Infastructure*. Morgan Kaufmann.

ISDL Home. (n.d.) *Interaction System Design Language*. http://isdl.ctit.utwente.nl.

Quartel, D. (1998). Action Relation: Basic Design Concepts for Behaviour Modeling and Refinement. *PhD. Thesis*. University of Twente.

Visser, C.A., Ferreira P. L., dan Quartel, D. (2002). *Architectural Design of Distributed Systems*. University of Twente.

W3C. (1999). *XML Path Language (XPath) Version 1.0*. http://www.w3c.org/TR/xpath/.

W3C. (2004). *Extended Markup Language (XML) Version 1.0 (Third Edition)*. http://www.w3.org/TR/2004/REC-xml-20040204/.