Jurnal Penelitian dan Pengembangan Komunikasi dan Informatika Vol. 4 No. 3 (Maret 2014 - Juni 2014)

ANALISIS KONTEKS PROSES BISNIS BERDASARKAN "*EVENT LOG*"

BUSINESS PROCESS CONTEXT ANALYSIS BASED ON "EVENT LOG"

Airlangga Adi Hermawan

TU Eindhoven
Den Dolech 2 Eindhoven The Netherlands
Email a: .airlangga.hermwan@studenthermawan@yahoo.tue.nl.com

diterima: 12 November 2013 | direvisi: 15 Januari 2014 | disetujui: 10 Februari 2014

Abstract

Nowadays, information systems in organizations logged many kinds of event logs. These logs can be analyzed with process mining. The goal of process mining is to extract information from event logs and use it as an insights into the business process of the organizations. Unfortunately, most process mining techniques only consider process instance in isolatkategorialnfluence the analysis of the organizations. In this paper, we developed an approach to get insights in to context-related information based on event logs, such that useful insights can be easily obtained. Our approach has been evaluated through two case studies. The evaluation outcome confirms that most of the context-related information from event logs can be analyzed.

Keywords : *Big Data, Business Process, Process Mining.*

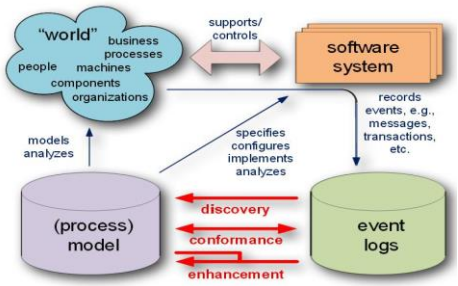
Abstrak

Sekarang ini, sistem informasi di organisasi dapat merekam log dari berbagai macam *event log*. Log ini dapat dianalisis dengan teknik *process mining*. Tujuan dari *process mining* adalah mendapatkan informasi yang berguna dari *event log*, dan menggunakannya sebagai panduan organisasi untuk menilai proses bisnisnya. Sedangkan konteks yang luas tersebut mungkin dapat mempengaruhi hasil analisis. Pada paper ini telah dibuat pendekatan untuk mendapatkan wawasan tentang *event logs* yang juga meliahat dari sisi konteksnya, sehinngga informasi yang didapat tidak ada yang hilang. Pendekatan yang telah dibuat dievaluasi menggunakan 2 kasus studi. Hasil dari evaluasi mengkonfirmasi bahwa sebagian besar dari informasi *event logs* berbasis konteks dapat dianalisis.

Kata kunci *: Big Data, Proses Bisnis, Process Mining.*

Pendahuluan

Process mining merupakan disiplin yang relatif baru. *Process mining* berada pada diantara computational intelligence dengan data mining, dan diantara pemodelan proses dan analisis proses . Inti dari *process mining* adalah menemukan proses bisnis, memonitor, dan melengkapi proses bisnis yang sudah tersedia pada sistem informasi. Gambar 1 mengilustrasikan skema dari *process mining*. Mengekstrak informasi berbasis konteks dapat dianggap sebagai upaya untuk melengkapi *event logs* yang sudah tersedia. Hasil dari proses tersebut baru kemudian dapat dianalisis menggunakan *process mining*. Ada tiga macam tipe dari *process mining*, yaitu: *discovery*, *conformance*, dan *enhancement.*



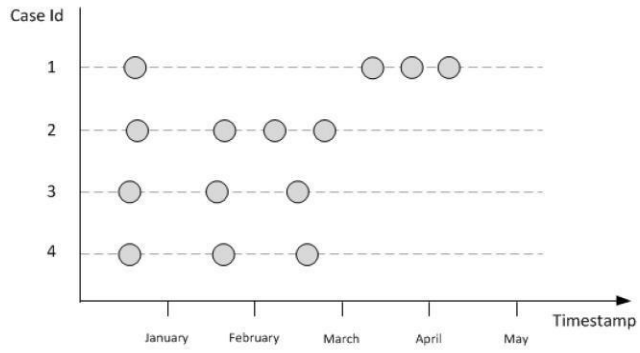
Gambar 1.
Skema "Process mining" (Aalst et Al, 2012)

Cara kerja teknik *discovery* adalah menganalisis suatu *event log*, kemudian dari *event log* tersebut teknik

ini akan membuat sebuah model tanpa menggunakan informasi yang apriori. Teknik ini merupakan teknik yang paling menonjol dari *process mining*(Aalst et al, 2012).

Di tipe kedua dari *process mining* ini, model proses yang sudah ada dibandingkan dengan *event log* yang dihasilkan pada proses yang sama. *Conformance* checking dapat digunakan untuk memeriksa apakah kejadian nyata yang terjadi sesuai dengan model yang ada, dan sebaliknya. Perlu dicatat bahwa aplikasi *conformance* checking sangatlah luas, dapat dilakukan pada model prosedural, model organisasi, kebijakan bisnis, dan lain-lain (Aalst et al, 2012).

Tipe ketiga dari *process mining* adalah *enhancement*. Tipe ini adalah memperluas atau menambah proses model yang sudah tersedia dengan informasi yang didapat dari *event log* di dunia nyata. Pada *conformance* lebih menekankan pada pengukuran perbedaan antara proses model dengan *event log* di dunia nyata. Sedangkan pada tipe ketiga ini bertujuan untuk mengganti atau menambah model a-priori. Sebagai contoh: menggunakan *timestamp* pada *event log* dapat memperluas model sehingga dapat menunjukkan dimana terjadi *bottleneck*, kepuasan pelayanan, waktu eksekusi, dan frekuensi (Aalst et al, 2012). Sekarang ini kebanyakan teknik *process mining* dilakukan hanya melihat satu *instance* saja, tanpa melihat *instance* yang lain.



Gambar 2
Contoh dari analisis log
(Hermawan, 2013)

Masalah Penelitian

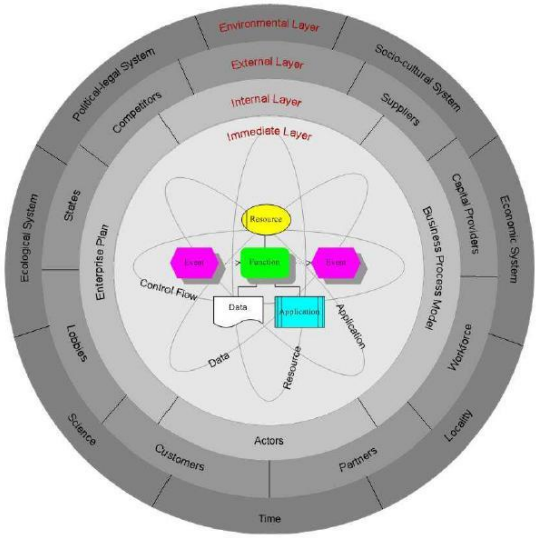
Misalkan saja ada 4 *instance* dalam sebuah proses, diilustrasikan pada gambar 2 terdapat 4 *instance* dalam sebuah proses. Axis x menunjukkan nilai dari *timestamp* dari sebuah *event log*, sedangkan axis y menunjukkan *instance*(*case*). Lingkaran kecil pada gambar tersebut merepresentasikan *event log*. Dalam *case* ini bisa dilihat bahwa *case* id mempunyai performa yang lebih buruk, karena membutuhkan waktu eksekusi yang lebih lama dibandingkan dengan *instance* yang lain. Tetapi jika kita lihat akar permasalahannya, kita tidak bisa melihat dari *case* pertama saja. Contohnya jika ada situasi dimana *case* 1 dieksekusi oleh beberapa orang, dan ada orang yang bertanggung jawab di semua *case*. Sedangkan orang tersebut sibuk di *case* yang lain sehingga *case* pertama tidak dikerjakan dalam beberapa waktu. Setelah orang itu tidak sibuk, maka orang itu dapat kembali mengerjakan *case* pertama. Jadi alasan lamanya eksekusi *case* pertama dikarenakan bukan dari *case* pertama tersebut, tetapi ada situasi dimana orang yang bertanggung jawab harus mengerjakan *case* yang lain. Untuk menganalisis hal ini sangatlah susah jika kita melihat dari *case* pertama saja. Oleh karena itu paper ini akan meneliti bagaimana pendekatan analisa log dengan konteks yang lebih luas.

Tinjauan Pustaka

Pada disiplin sistem informasi, istilah atau yang biasa disebut dengan context-aware diperkenalkan oleh Schilit & Theimer (1994). Definisi lainnya diperkenalkan oleh Dey(2001) sebagai konteks semua informasi yang dapat digunakan untuk melihat lebih dalam situasi dari sebuah *case*. Pada Schmidt (2010) konteks didefinisikan sebagai kombinasi dari semua keadaaan baik implisit ataupun eksplisit yang dapat mempengaruhi situasi dalam sebuah proses. Dapat dikatakan bahwa konteks mempengaruhi *case-case* yang terjadi di dalam sebuah proses. Dalam lingkup proses bisnis, *context-awareness* dapat dikatakan sebagai sesuatu yang baru. Walaupun demikian, ada beberapa peneilitan yang telah dilakukan yang berhubungan dengan konteks. Pada Rosemann et al(2008), sang pengarang mempertimbangkan untuk menggunakan fleksibilitas sebuah proses dilihat dari konteksnya, dan memperkenalkan *framework* untuk mengklasifikasi konteks. Pada Ploesser(2009), *event logs* yang lampau digunakan untuk memperbaiki proses bisnis. Walaupun ada beberapa penelitian tentang *context awareness* tetapi hanya beberapa peneliti saja yang meniliti dari data mentah dan fokus mendapatkan informasi yang dilihat dari sisi konteksnya, seperti yang dilakukan oleh 4, sang pengarang memperkaya *event logs* dari sumber eksternal dengan informasi tambahan untuk memprediksi masa depan. Dapat dikatakan bahwa sampai saat ini sebagian besar analisis data dilakukan secara tradisional tanpa mempertimbangkan konteks.

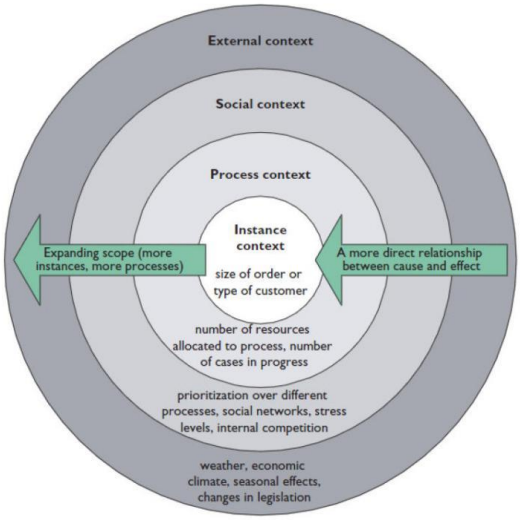
Sekarang ini banyak sekali metode untuk meningkatkan proses bisnis. Salah satu ide adalah menggunakan konteks untuk mengetahu lebih dalam tentang proses bisnis. Di Schmidt (2010) dan Rosemann et al(2008) fleksibilitas dalam proses bisnis mulai dikenalkan. Pengarang dalam kedua paper tersebut memaparkan bahwa dengan menggabungkan antara aspek kontekstual dengan tujuan dari model proses bisnis, fleksibilitas yang diperlukan untuk mengatasi perubahan yang terjadi karena faktor lingkungan, dapat dimodelkan sebagai "onion model". Model ini terbagi menjadi 4 lapisan konteks: *immediate*, internal, eksternal, dan *environmental*. Konteks *immediate information* dapat ditangkap dengan metode tradisional misalkan menggunakan *flow data*(misal: urutan dari activity, *event log*s, dan juga transisinya). Konteks internal mempunyai

hubungan yang kurang spontan terhadap proses dibandingkan dengan *immediate.* Konteks ini bisa berupa informasi dalam suatu organisasi yang mempengaruhi proses. Konteks eksternal bisa berupa elemen yang berada di luar organisasi atau perusahaan, tetapi masih bersinggungan dengan jaringan perusahaan. Konteks environmental berada pada lapisan terluar, di luar jaringan perusahaan, tetapi masih mempengaruhi proses bisnis. Model ini dapat dilihat pada Gambar 3.



Gambar 3.
Model "Onion" (Schmidt (2010)
dan Rosemann et al(2008)

Pada paper ini, kami membatasi ruang lingkup penelitian untuk mendapatkan informasi yang juga memperhatikan konteks pada *event log*. Dalam proses untuk mendapatkan informasi berbasis konteks, kami menggunakan definisi konteks yang telah dilakukan pada penelitian sebelumnya (Aalst, 2012), dan meng-ekstrak informasi berbasis konteks dari data mentah *event log* proses bisnis. Dari konteks yang telah diekstrak kami berusaha untuk mendapatkan kesimpulan dari informasi tersebut. Pada Aalst(2012) , sang pengarang berargumen bahwa ketika proses ekstrasi dilakukan pada data mentah untuk *process mining*, konteks harus diperhatikan. Gambar 4 menunjukkan bahwa ada 4 macam konteks yang berhubungan dengan aplikasi proses bisnis. Setiap tipe konteks diilustrasikan sebagai lapisan. 4 lapisan konteks dari dalam hinggal luar sebagai berikut: konteks *instance*, proses, sosial, dan eksternal. Proses *instance* mempunyai beberapa tanda dalam mempengaruhi proses secara keseluruhan. Misalkan saja ada sebuah proses bisnis. Tipe dari konsumen(contoh: eksekutif, reguler) dapat mempengaruhi alur eksekusi proses bisnis(Aalst & Dustdar, 2012). Hanya melihat sebuah *instance* kecil tanpa melihat *instance* lainnya belum cukup untuk melihat prsoes keseluruhan. Teknik *process mining* juga harus memperhatikan konteks proses. Misalkan saja status pemesanan tidak cukup, tetapi juga harus dengan melihat faktor lain seperti *workload* atau sumber daya manusia yang tersedia. Konteks proses mencakup semua faktor yang berhubungan langsung dengan proses dan *instance*-nya(Aalst & Dustdar, 1012).



Gambar 4
Empat lapisan konteks
menurut Aalst & Dustdar(2012)

Konteks sosial melihat faktor-faktor yang lebih luas dibandingkan konteks proses, seperti: level stress pekerja, hubungan sosial antar pekerja, dan lain-lain. Sedangkan konteks eksternal dapat berawal dari luar lingkungan organisasi(Aalst & Dustdar, 1012). Untuk mengidentifikasi konteks *instance* ataupun konteks proses, sangatlah mudah. Tetapi susah untuk mengidentifikasi konteks sosial, dan eksternal, karena keterbatasan variabel pengukuran. Kami menggunakan kategori-kategori ini dalam menganalisis log. Untuk mempermudah pemahaman dari istilah-istilah yang ada di dalam paper ini, kami paparkan definisi sebagai berikut:

Definisi Event log dan Atribut

E adalah semesta dari *event log*s, misal: kumpulan *event log* yang mungkin ada di dalam log. *Event log*s dapat memiliki atribut. Æ adalah semesta dari nama atribut. Untuk setiap *event log* e Є E dan nama n Є Æ, #ae (e) adalah untuk nilai atribut ae dalam e. Jika *event log* e tidak mempunyai nama atribut n, maka #ae (e) = (tidak mempunyai nilai). Beberapa tipe atribut yang mungkin terdapat dalam *event log* sebagai berikut:

1. #act (e) adalah aktivitas dalam *event log* e.
2. #time (e) adalah timestamp dari *event log* e.
3. #resource (e) adalah sumber daya manusia dari *event log* e.

Nilai dari atribut ae Є AE dapat berupa numerik atau kategorial.

1. #ae (e) merupakan numerik jika dapat diurutkan berdasarkan dari nilai yang terkecil hingga terbesar.
2. #ae (e) adalah kategorial jika tidak dapat diurutkan dari yang nilai terkecil hingga terbesar.

Setiap *event log* dalam *event log* pasti termasuk dalam satu case. Dapat dilihat pada Tabel 1, *event log* id 112 termasuk ke dalam case id 1.

Definisi Log

Log merupakan data yang terurut dalam satu tingkatan L = (E, C, β, ») dimana E ε merupakan set dari *event logs*, C sebuah set dari *cases*, β : E C adalah pemetaan dari *event logs* ke *cases*, dan » E x E adalah sebuah *total ordering* antara *event logs* , misal: berurutan berdasar *timestamp*. Untuk semua (e1, e2) Є » e1, e1 diurutkan hingga yang terbesar hingga e2, dengan notasi e1 » e2.L adalah semesta log. Ć adalah semesta "*cases*". Mirip dengan *event logs*, *case* juga mempunyai atribut. ÀĆ adalah semesta dari nama atribut "*case*". Untuk semua *case* c є C dan nama atribut *case* ac є AC, #ae(c) mendefinisikan nilai dari atribut ac di c. Jika *case* c tidak mempunyai atribut ac, maka #ae(c) = Semua *cases* di log mempunyai spesial atribut yang harus ada. trace. Untuk semua *cases* c Є C dari L, #trace(c) = δ Є E* sehingga:

1. Untuk semua 1 i < j |δ| : δ(i) δ (j) and δ (i) » δ (j), misal: *event log* adalah unik dan diurutkan berdasar total ordering
2. Untuk semua e Є E dimana β(e) = c, terdapat 1 i |δ| sehingga δ(i) = e, misal: semua *event log* dari *case* muncul dalam δ

Jika tabel 1 diformulasikan dengan metode formal, maka:

Tabel 1.
Potongan dari event log, setiap baris merepresentasikan satu buah event

Case id	Event id	Properti			
		Timestamp	Aktivitas	Sumber daya manusia	...
1	112	⊥	Register	Budi	...
	113	31-12-2012:11:02	Teliti	Adi	...
	114	1-1-2013:10:06	Cek tiket	Wawan	...
	115	1-1-2013:15:20	Memutuskan	Sheila	...
	116	5-1-2013:10:32	Menolak	Budi	...
					...
2	117	6-1-2013:10:07	Register	Budi	...
	118	7-1-2013:15:02	Teliti	Adi	...
	119	8-1-2013:13:34	⊥	Cinta	...
	120	10-1-2013:12:09	Memu-tuskan	⊥	...
	121	11-1-2013:15:02	Meneri-ma	Budi	...
...	...	...	...	...	...

Log L= {1,2…} mempunyai

1. #trace(1)= <112, 113, 114, 115, 116> merupakan trace dari case 1.
2. #act(112)= <Cek tiket> merupakan aktivitas dari suatu *event log*.
3. #time(112)= < > tidak ada nilai waktu yang memenuhi di *event log* 112, dst.

Metode Penelitian

Untuk mendapatkan hasil yang objektif dan menjawab pertanyaan riset langkah-langkah berikut ini harus dilakukan

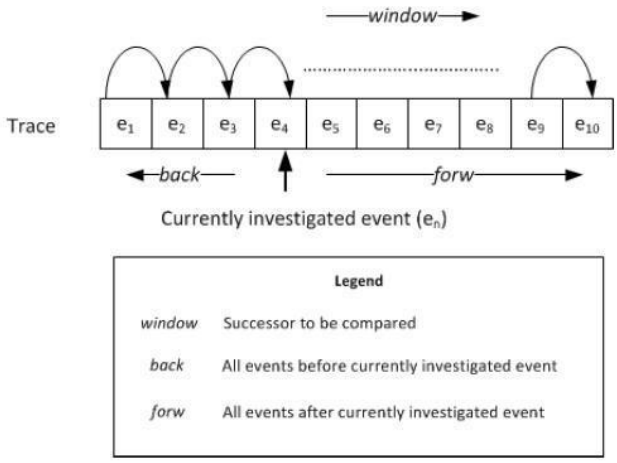
1. Melalukan studi literatur analisis performa proses bisnis.
2. Identifikasi konteks yang mungkin memberikan informasi tentang proses keseluruhan.
3. Membuat visualisasi yang dapat menganalisis informasi yang berhubungan dengan konteks.
4. Implementasi pendekatan pada ProM *framework*
5. Mengevaluasi hasil metode dengan log *event log* dari dunia nyata Menganalisis hasil yang didapat dari visualiasi.

Hasil Penelitian dan Pembahasan

Memperbaiki Atribut "Timestamp"

Dalam dunia nyata, *event log* mungkin mengandung nilai yang tidak benar. Kasusnya dapat berbagai macam, misal: ada kesalahan dalam penulisan ketika memasukkan data, server yang merekam data rekam medis tidak sinkron dengan kenyataan, dan lain-lain. Sebuah rumah sakit mungkin menggunakan metode *paper-based* rekam medis untuk menyimpan data. Suatu ketika ada seorang pasien dalam kondisi kritis datang, data pasien kemudian ditulis dengan pena dan kertas. Dalam kondisi ternyata tidak semua data dimasukkan secara benar, ada kesalahan dalam penulisan di *timestamp*. Ketika pasien sudah dalam keadaan stabil data informasi yang dimasukkan ke dalam basis data sistem informasi juga mengandung kesalahan. Efisiensi *storage* juga dapat membuat hilangnya atribut dalam suatu log. Misalnya saja ada sebuah perusahaan yang mempunyai beberapa karyawan dengan proses bisnis yang sederhana. Kemudian suatu ketika perusahaan tersebut memutuskan bahwa ingin menghilangkan atribut *timestamp* di dalam proses yang tidak begitu penting. Karena tidak semua informasi mengenai *timestamp* terekam dalam basis data. Hal ini dapat menyebabkan hilangnya informasi tersebut. Adanya kesalahan dan hilangnya nilai dari *event log* dapat menyebabkan tingkat kualitas informasi yang bias didapat dari *event log*. Dalam bagian ini akan dijelaskan bagaimana mengatasi kedua hal ini dengan menggunakan pendekatan heuristik. Dengan asumsi bahwa *event log* yang berada di dalam log diurutkan berdasar atributnya. Pendekatan yang dilakukan membatasi panah nilai yang bersifat numerikal saja, misal: *timestamp*. Untuk memperbaiki *timestamp* memulai dengan memeriksa awal dari sebuah trace apakah mempunyai nilai yang salah, kemudian setiap *event log* akan diperiksa dengan cara membandingkan

nilai ke nilai sebelum dan sesudahnya. Setelah langkah ini selesai, maka langkah selanjutnya adalah mengisi nilai *timestamp* yang hilang dari suatu *event log*. Jika *event log* terletak pada awal dari sebuah trace, *timestamp* yang baru akan disalin ke suksesor berikutnya. Jika nilai *event log* yang hilang berada pada trace yang ditengah-tengah.Maka nilai yang akan diberikan berasal dari predecessor. Andaikan semua *event log* yang berada pada trace tidak mempunyai *timestamp*, maka nilai yang diberikan kepada semua *event log* merupakan masukan dari pengguna. Gambar 5 mengilustrasikan bagaimana proses memperbaiki nilai timestamp.

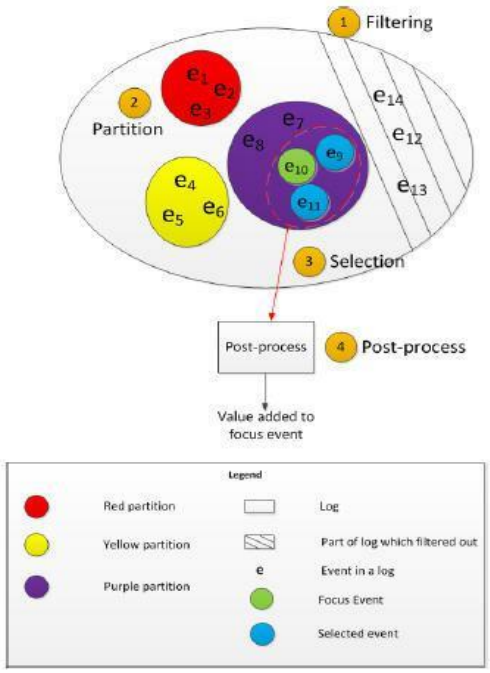


Gambar 5
Algoritma perbaikan timestamp
(Hermawan, 2013)

Mendapatkan Konteks

1. Mendapatkan konteks dari sumber internal. Untuk mendapatkan konteks dari sumber internal, beberapa langkah harus dilakukan, yaitu : *filter, partition, selection*, dan *post-process*.
   Filter. Langkah ini menghilangkan *event log* yang tidak diinginkan. Pada contoh ini e12, e13, dan e14.
   Partition. Setelah semua *event log* yang tidak diinginkan dihilangkan, *Event logs* akan tergabung dalam satu partisi, dan hanya bisa menjadi anggota tidak lebih dari satu partisi. Gambar 6 menunjukkan bahwa ada 3 partisi: merah, kuning, dan ungu.
   Selection. Untuk setiap *event log*, akan diseleksi sebuah kumpulan *event log* yang cocok dengan kriteria yang telah ditentukan pada partisi yang sama. Gambar 6 menunjukkan bahwa *event log* e12, e13, dan e14 telah dipilih, dan e10 merupakan fokus *event log*. Fokus *event log* adalah *event log* yang informasinya akan dilengkapi. Pada akhirnya setiap *event log* akan menjalani proses ini dan akhirnya semua *event log* informasinya akan dilengkapi.
   Post-process. Event log yang telah dipilih dan fokus *event log* kemudian dapat diproses lebih lanjut:

   - Jumlah dari nilai sebuah atribut: mendapatkan jumlah nilai dari atribut *event log* yang telah dipilih, yang kemudian dimasukkan ke dalam *event log* fokus.
   - Nilai minimum: mendapatkan minimum nilai dari atribut *event log* yang telah dipilih, yang kemudian dimasukkan ke dalam *event log* fokus.
   - Nilai maksimum: mendapatkan maksimum nilai dari atribut *event log* yang telah dipilih, yang kemudian dimasukkan ke dalam *event log* fokus.
   - Rata-rata: mendapatkan rata-rata nilai dari atribut *event log* yang telah dipilih, yang kemudian dimasukkan ke dalam *event log* fokus.



Gambar 6
Skema untuk mendapatkan konteks
dari sumber internal (Hermawan, 2013)

Sublog

L = (E, C, β, ») adalah sebuah log. L' = (E', C', β',»') adalah sebuah sublog dari L, dilambangkan dengan L' L, jika hal ini benar, maka:
E' E, contoh: *event logs* dari L' adalah subset dari L, C' = {β(e) | e Є E' }, contoh: *case* dari L' adalah subset dari L
β' : E' → C' adalah fungsi pemetaan untuk semua e Є E', β'(e) = β(e).

»' = {(e1, e2) Є E' x E' |(e1, e2) Є » }, contoh: *total ordering* antara *event log* tetap dijaga.

Definisi Partitioning

Asumsi P adalah sebuah himpunan partisi. P adalah pemetaan *prt* : L x P → L sehingga untuk semua log (E, C, β, ») Є L, hal ini berlaku:

prt(L, p) L, contoh: log dipetakan ke sublog,

Event log e Є E, terdapat p 2 P sehingga prt(L, p) = (E', C', β,' »') dan e Є E', sehingga semua *event log* merupakan bagian dari sublog

Untuk semua partisi p1, p2 Є P, *prt*(L, p1) = (E1, C1, β1,»1), prt(L, p2) = (E2, C2, β2,»2) : E1 E2 6= → p1 = p2, contoh: sebuah *event log* hanya bias menjadi anggota satu sublog.

Definisi Selection

Selection sel : є x L L fungsi parsial yang mengembalikan sublog berdasarkan *event log* dan log, sehingga pada semua *event log* (E, C, β, ») Є L, e Є E : sel(e, (E, C, β, »)) , (E, C, β, »)

Definisi Post-process

Asumsi æ Є Æ adalah sebuah nama atribut, dan asumsi bahwa AV adalah set set dari semua nilai yang mungkin untuk atribut æ. Untuk setiap *event log* e Є E, maka untuk setiap nilai atribut æ diturunkan dari fungsi *selection* sel dan *post pro-process* pst æ adalah:

$$\#_{æ}) = \quad (\text{sel}(e, \quad (e)))$$

Untuk mempermudah pemahaman dari definisi diatas, maka berikut ini adalah contoh dari pendekatan yang digunakan dalam salah satu kasus yang diteliti pada paper ini. Untuk lebih jelasnya diberikan contoh sebagai berikut:

Partitioning - originator values

Fungsi partitioning $prt_{resource}$: L x P → L membagi log menjadi beberapa bagian tergantung originator(sumber daya manusia) (E, C, β, ») Є L, *originator* r Є {#resource (e) Є E}, prtresource (E, C, β, ») = (E', C', β', »') dimana

E' = {e Є E | #resource (e)}, the sub log mengandung semua *event log*s yang dikerjakan oleh sumber daya manusia r

C' = {(e) |e Є E'}, contoh: semua case de ngan *event log* E

β : E' | C' adalah fungsi pemetaan dari e Є E', β'(e) = (e)

»= {(e1; e2) Є E' x E' (e1; e2) Є » }, urutan tiap *event log* di dalam log tidak berubah

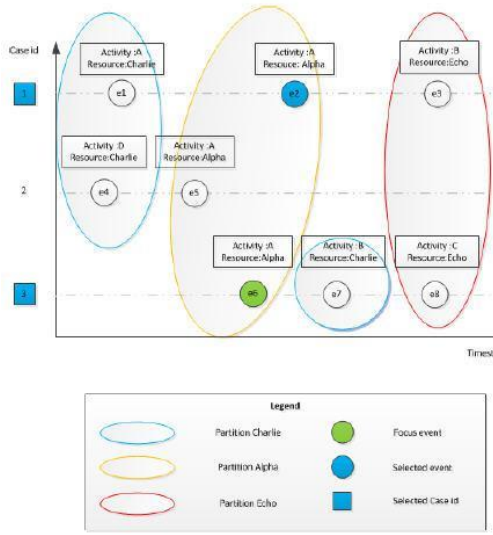
Selection - active cases

Fungsi *selection* sebagai berikut: sel_{act_case} : E x L L sebuah log mengandung *event log*s yang berada pada active cases, seumpama ada log (E, C, β, ») Є L, e Є E : sel_{act_case}(e, (E, C, β, »)) = (E', C', β', »') dimana

E' = {e' Є E |#time(#trace(β(e'))(1) #time(e) #time(#trace(β(e'))(|#trace(β(e'))|) }

C' = {β(e') | e' Є E'}, contoh: semua cases di *event log* E

»' = {(e1,e2) Є E' x E' | (e1,e2) Є »}, contoh: urutan di *event log* tetap dijaga



Gambar 7
Contoh aplikasi dari mendapatkan konteks dari sumber internal (Hermawan, 2013)

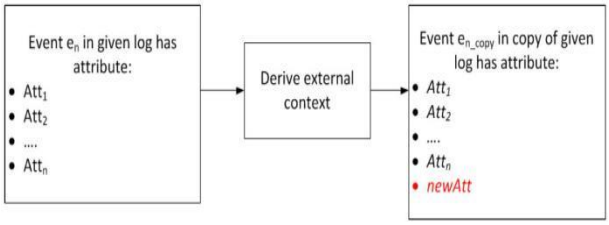
Post-process - count unique activities

Fungsi post-process pst_{cnt}_act : L → N mengembalikan jumlah *activity* dari semua *event log* yang ada di log (E, C, β, ») Є L, pstcnt_act((E, C, β, ») = {|#act(e) | e Є E} . Proses-proses ini dapat diilustrasikan melalui Gambar 7.

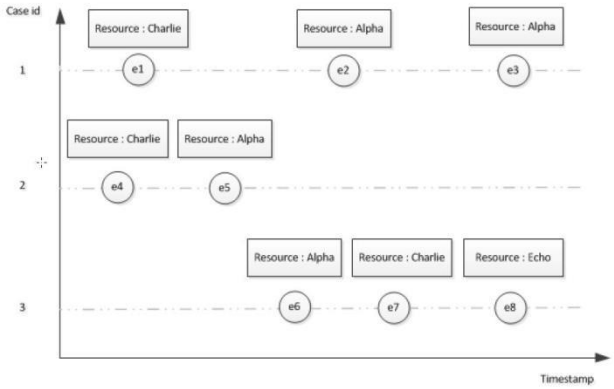
Mendapatkan konteks dari sumber eksternal

Beberapa konteks dapat diekstrak dari log tanpa informasi tambahan dari luar. Tetapi untuk mendapatkan konteks eksternal, beberapa informasi harus ditambahkan ke dalam log. Misal: cuaca ketika *event log* dieksekusi, atau saham pada periode tertentu. Gambar 8 mengilustrasikan bagaimana mendapatkan konteks eksternal. Seumpama *event log* pada *event log* en mempunyai atribut: Att1, Att2,..., Attn. Log asli akan diduplikasi, kemudian atribut baru akan ditambahkan semua *event log* pada log tersebut. Misalkan kita akan menambahkan atribut baru newAtt, maka pada log baru akan dihasilkan *event log* en dengan atribut Att1, Att2,..., Attn, dan newAtt. Gambar 8 mengilustrasikan bagaimana proses memperkaya *event log* dengan atribut baru. Untuk menambahkan atribut dari eksternal dapat

dilakukan dengan cara, yaitu numerik atau kategorial. Pada kasus kategorial misalkan terdapat log dengan kasus seperti yang diilustrasikan pada Gambar 9.



Gambar 8
Skema mendapatkan konteks dari sumber eksternal (Hermawan, 2013)



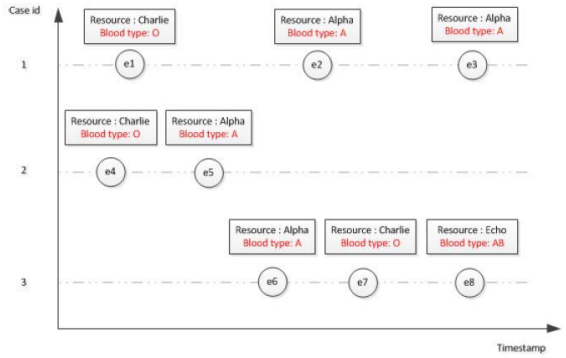
Gambar 9
Contoh log dengan atribut tipe kategorial sebelum ditambahkan nilai dari sumber eksternal (Hermawan, 2013)

Log kemudian ditambah dengan atribut baru seperti yang ditunjukkan pada Tabel 2.

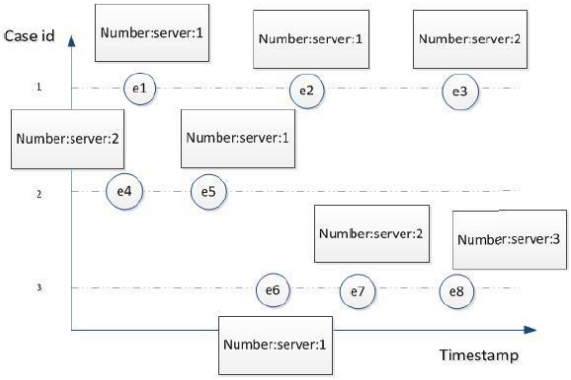
Tabel 1
Memetakan antara sumber daya manusia dengan golongan darah

Key(sumber daya manusia)	Value(Golongan darah)
Alpha	A
Charlie	O
Echo	AB

Log hasil duplikasi akan mempunyai atribut baru, seperti ditunjukkan pada Gambar 10. Dapat dilihat bahwa setiap *event log* yang mempunyai nilai atribut Alpha sebagai sumber daya manusia akan ditambah dengan atribut baru A, begitu pula dengan Charlie, akan ditambah dengan nilai O, dan Echo dengan nilai AB. Contoh lain dapat dilihat pada Gambar 11. Dalam log tersebut terdapat atribut nilai dengan tipe numerik, yaitu *Number:server*. Setiap *event log* yang mempunyai rentang nilai tertentu akan ditambah dengan atribut baru.



Gambar 10
Contoh log dengan atribut tipe kategorial setelah ditambahkan nilai dari sumber eksternal (Hermawan, 2013)

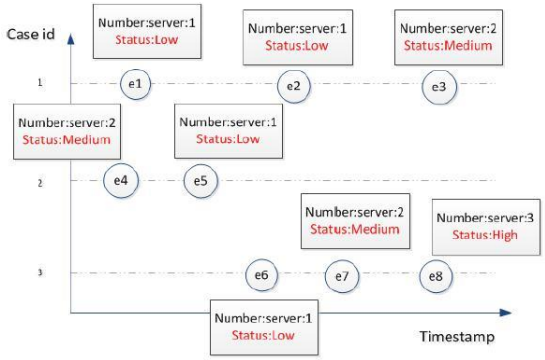


Gambar 11
Contoh log dengan tipe numerik sebelum ditambahkan nilai dari sumber eksternal (Hermawan, 2013*)*

Tabel 2
Memetakan antara *Number:server* dengan Status

Key(Number:server)	Value(Status)
0-1	Low
2-2	Medium
3-...	High

Gambar 12 menunjukkan hasil dari log dengan nilai atribut baru, yaitu Status. Nilai dari status tergantung dari nilai atribut *Number:server*, sesuai dengan aturan yang telah didefinisikan pada Tabel 3.

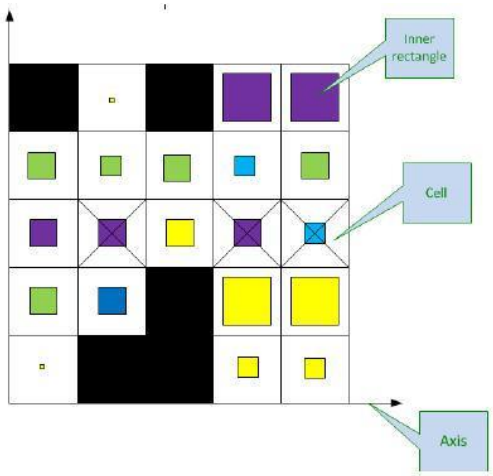


Gambar 12
Contoh log dengan tipe numerik setelah ditambahkan nilai dari sumber eksternal (Hermawan, 2013)

Context analyzer

Context analyzer terdiri dari 3 bagian: *inner rectangle*, *cell*, dan axis seperti diilustrasikan pada gambar 13.

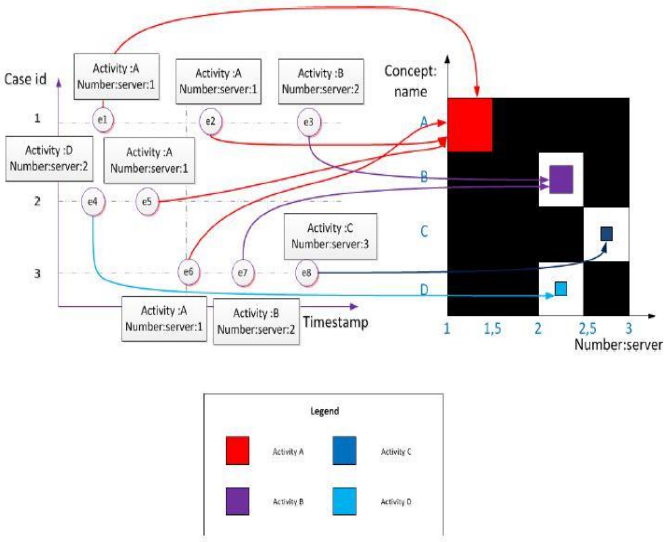
Cell adalah bagian terpenting dari *context analyzer*. Atribut dari log akan dipresentasikan di dalam *cell*. Pada *cell* terdapat kotak yang dinamakan *inner rectangle*. Axis digunakan untuk memetakan informasi berbasis konteks ke dalam *cell*. Pengamatan terhadap perilaku akan sulit diukur kecuali melalui alat ukur atau instrumen yang memiliki nilai ketepatan tinggi. Namun, untuk membuat instrumen dengan nilai ketepatan uji ukur tinggi tidaklah mudah. Beberapa kriteria yang dirumuskan dalam konstruk indikator dan variabel memiliki ketepatan ukur yang berbeda sehingga perlu dilakukan uji validitas dan reliabilitas untuk memperoleh hasil yang signifikan. Instrumen yang baik memiliki tingkat validitas dan reliabilitas yang cukup sehingga menghasilkan informasi akurat dan berlaku sebaliknya.



Gambar 13
Konsep tampilan dari *Context analyzer* (Hermawan, 2013)

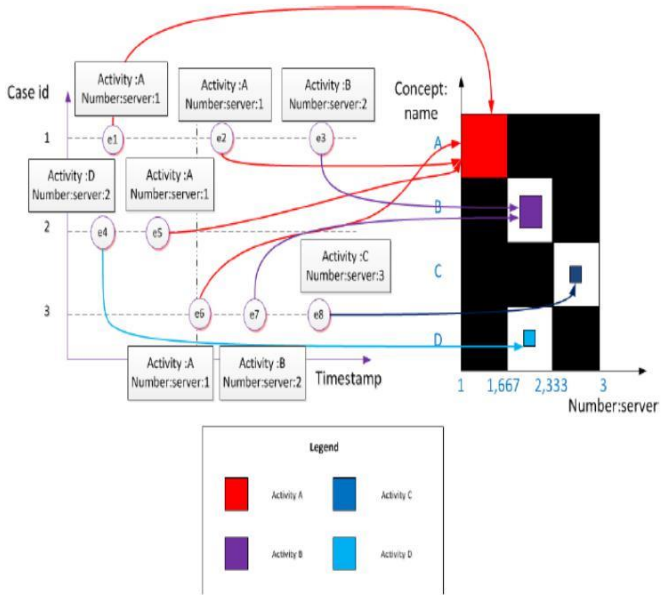
Gambar 14 mengilustrasikan bagaimana *event log* dapat divisualisasikan menggunakan *context analyzer*. Di bagian kiri gambar 14 dapat dilihat terdapat 8 *event log*. Setiap *event log* mempunyai atribut dan nilai. Atribut activity merupakan tipe kategorial, dan mempunyai nilai A, B, C, dan D. *Number:server* merupakan atribut tipe numerik, mempunyai rentang nilai antara 1 dan 3. Pada sumbu x menunjukkan atribut *Number:server*, sedangkan sumbu y menunjukkan atribut concept:name. Seumpama dalam satu log kita akan membagi menjadi 4 kolom, maka setiap kolom haruslah mempunyai nilai 0,5. Nilai ini dapat diperoleh menggunakan persamaan sebagai berikut:

$$Interval = \frac{nilai\ maksimum - nilai\ minimum}{Jumlah\ pembagi\ (kolom/baris)}$$



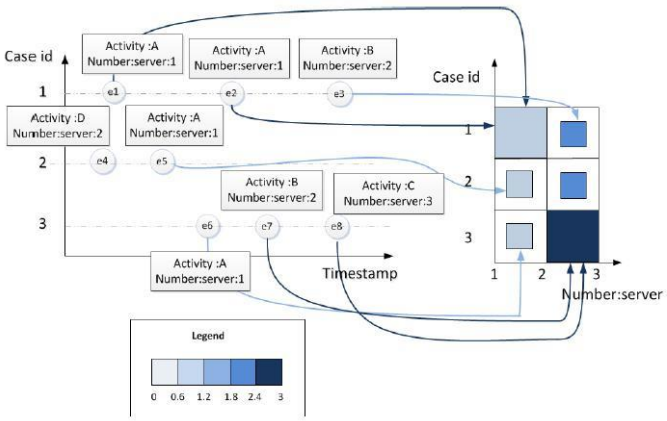
Gambar 14
Visualisasi dari "Event log" menggunakan *Context analyzer* (Hermawan, 2013)

Seumpama pengguna ingin merubah jumlah kolom menjadi 3 bagian, maka akan terlihat seperti pada Gambar 15. Lebar nilai kolom akan menyesuaikan.



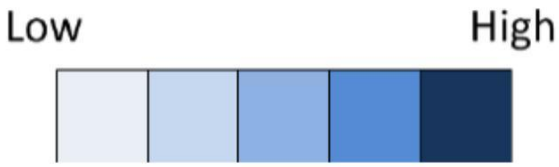
Gambar 15
Visualisasi dari "Event log" menggunakan *Context analyzer* (Hermawan, 2013)

Pada Gambar 16 dapat dilihat bahwa log dapat divisualisasikan menurut atributnya. Disini atribut yang digunakan sebagai acuan adalah *Number:server*, yaitu tipe numerik.



Gambar 16
Visualisasi dari "Event log" menggunakan *Context analyzer* dengan pewarnaan tipe numerik (Hermawan, 2013)

Pewarnaan atribut dapat dibagi menjadi 5 tingkatan kecerahan, seperti yang diilustrasikan pada Gambar 17.



Gambar 17
Lima tingkatan warna dalam atribut numerik (Hermawan, 2013)

Semakin kecil nilai dari atribut tersebut, maka warna biru yang ditampilkan akan semakin muda, sedangkan jika nilai atribut tersebut relatif besar, maka warna yang ditampilkan akan semakin gelap. Dalam *context analyzer*, ada beberapa pilihan sistem dalam menampilkan warna, yaitu:

Jumlah *event log*s. Merepresentasikan jumlah *event log* yang berada dalam satu *cell*.

Maksimum. Merepresentasikan nilai nilai minimum atribut dari *event log* yang berada dalam *cell*.

Minimum. Merepresentasikan nilai minimum atribut dari *event log* yang berada dalam *cell*.

Jumlah. Merepresentasikan jumlah nilai atribut dari *event log* yang berada dalam *cell*.

Rata-rata. Merepresentasikan rata-rata nilai atribut dari *event log* yang berada dalam *cell*.

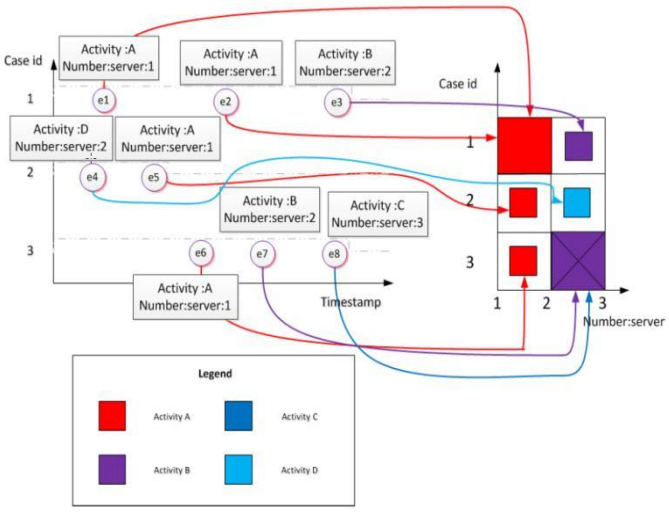
Standar deviasi. Merepresentasikan nilai standar deviasi dari nilai atribut dari *event log* yang berada dalam *cell*.

Dalam log ini terdapat 3 *event log*s dalam case 1: e1, e2, dan e3 yang termasuk ke dalam baris pertama, sedangkan pada baris kedua ada e4, e5, dan e6. Sisanya masuk ke dalam baris ketiga. e7 dan e8 termasuk ke dalam *cell* yang sama, tetapi mempunyai nilai *activity* yang berbeda yaitu activity B dan C. Pewarnaan dalam kasus ini adalah berdasarkan warna dari nilai atribut ativity. Terdapat 2 *activity* yang berbeda, sedangkan *cell* hanya bisa menampilkan satu warna saja. Oleh karena itu jika terdapat lebih dari satu nilai kategorial pada satu *cell*, maka *cell* tersebut akan diberi dengan tanda silang di dalamnya.

Implementasi di Dunia Nyata

Dalam percobaan ini kami menggunakan log yang mempunyai atribut sebagai berikut:

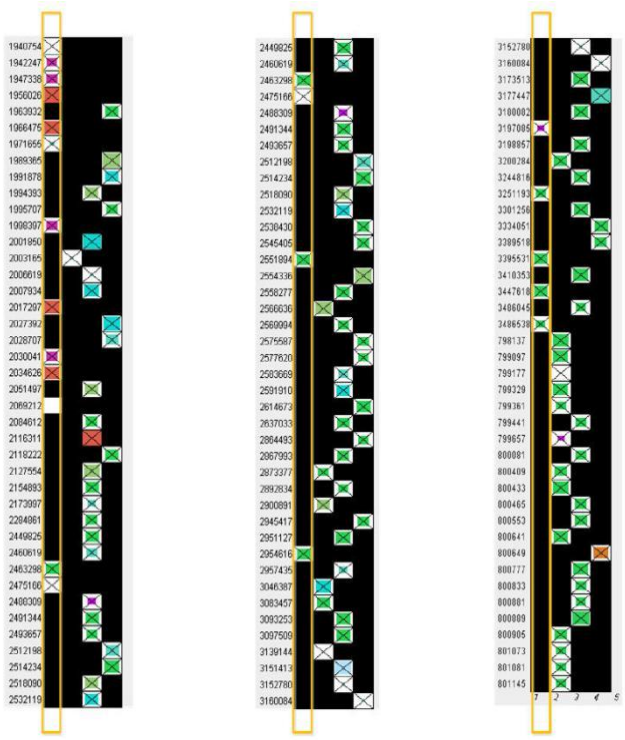
Concept:name
Lifecycle:transition
Org:resource
Time:timestamp
Time_stamp



Gambar 18
Visualisasi dari *Event log* menggunakan *Context analyzer* dengan pewarnaan tipe kategorial (Hermawan, 2013)

Interaksi antar sumber daya manusia dalam sebuah instance

Menggunakan *context analyzer.* Setelah log informasinya diperkaya menggunakan konteks internal, semua *event log* didalam log akan ditambah dengan atribut baru, dengan nama internal: *unique originator*, sebuah tipe numerik. Atribut ini akan memberikan informasi tentang banyaknya jumlah sumber daya manusia yang digunakan dalam satu *case*. Gambar 19 mengilustrasikan log yang telah diperkaya oleh konteks internal. Setiap baris merepresentasikan *case* sedangkan setiap kolomnya akan merepresentasikan banyaknya sumber daya manusia dalam satu *case*. Tipe kategorial akan diurutkan berdasarkan alphabet, sedangkan tipe numerikal akan diurutkan dari nilai terkecil hingga terbesar dari log tersebut. Dari gambar 19 bisa dilihat bahwa setiap baris mempunyai satu warna *cell*, yang artinya setiap *case* hanya mempunyai satu sumber daya manusia saja. Beberapa *case* mempunyai *inner rectangle* yang memenuhi satu *cell*, dan beberapa dari mereka ada yang sebagian dari *cell*. Dari hal ini bisa dilihat bahwa *inner rectangle* berhubungan dengan jumlah *event log* pada *cell* relatif terhadap jumlah *event log* di log pada semua *cell*. Dapat dilihat pula bahwa kebanyakan *case* terletak pada *inner rectangle* pada kolom pertama dan ketiga, yang berarti bahwa kebanyakan *case* dieksekusi oleh 2 sampai 3 SDM yang berbeda, hanya dari beberapa yang dieksekusi oleh 4 orang yang berbeda. Yang menarik lagi dalam gambar 19 adalah bahwa kebanyakan *case* mempunyai *inner rectangle* pada kolom pertama, artinya *cell* tersebut dieksekusi oleh hanya satu orang saja. Sedangkan jumlah *event log* yang dieksekusi lumayan tinggi. Ada *case* dimana hanya satu orang saja yang dieksekusi oleh orang yang sama, hal ini sangat tidak wajar pada proses bisnis. Misalkan ada satu orang yang membeli produk dari perusahaan dimana dia bekerja, dia dapat mengisi formulir pembelian, mengkonfirmasi pembelian produk dirinya sendiri, dan mengkonfirmasi bahwa uang yang digunakan untuk membeli produk itu telah diterima oleh perusahaan. Situasi ini dapat menjurus kepada pemalsuan transaksi.



Gambar 19
Interaksi antar sumber daya manusia dalam sebuah instance - Context analyzer (Hermawan, 2013)

Menggunakan *dotted chart.* Gambar 20 menunjukkan bagian dari visualisasi *dotted chart*. Setiap baris merepresentasikan *instance*, dan kolom mempresentasikan bulan. Dapat dilihat pada gambar tersebut bahwa di bagian baris ada beberapa titik yang mempunyai warna yang sama, ini berarti bahwa di *instance* tersebut hanya satu orang saja yang mengeksekusi semua *event log*. Beberapa baris mempunyai titik dengan warna yang berbeda, hal ini menandakan bahwa di instance tersebut ada beberapa orang yang mengeksekusi event log. Di bagian yang telah diberi tanda menunjukkan bahwa ada beberapa titik dengan berbagai warna, beberapa titik tersebut bergabung menjadi titik yang lebih besar. Untuk melakukan analisis dengan titik yang besar sangatlah sulit, karena kita harus mengetahui tiik itu di baris dan kolom yang mana.

Gambar 20
Interaksi antar sumber daya manusia dalam sebuah instance - Dotted chart
(Hermawan, 2013)

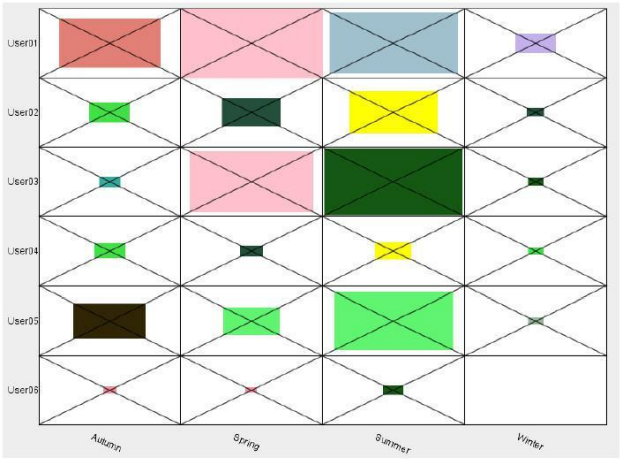
Pengaruh cuaca ke produktivitas sumber daya manusia

Menggunakan *context analyzer.* Untuk memetakan *event logs* ke dalam atribut yang baru, aturan pemetaan haruslah digunakan, seperti yang terlihat pada gambar 21. Dalam baris pertama aturan, *timestamp* sebelum 2009-03-20T01:00:00+02:00 dan hingga waktu tersebut maka akan diberi atribut Winter. Pada baris kedua, *timestamp* dari 2009-03-20T01:00:00+02:00 ingga 2009-06-21T01:00:00+02:00 diberi nilai atribut baru Spring, dst.

```
From,To,Value
,2009-03-20T01:00:00+02:00,Winter
2009-03-20T01:00:00+02:00,2009-06-21T01:00:00+02:00,Spring
2009-06-21T01:00:00+02:00,2009-09-22T01:00:00+02:00,Summer
2009-09-22T01:00:00+02:00,2009-12-21T01:00:00+02:00,Autumn
2009-12-21T01:00:00+02:00,2010-03-20T01:00:00+02:00,Winter
2010-03-20T01:00:00+02:00,2010-06-21T01:00:00+02:00,Spring
2010-06-21T01:00:00+02:00,2010-09-22T01:00:00+02:00,Summer
2010-09-22T01:00:00+02:00,2010-12-21T01:00:00+02:00,Autumn
2010-12-21T01:00:00+02:00,,Winter
```

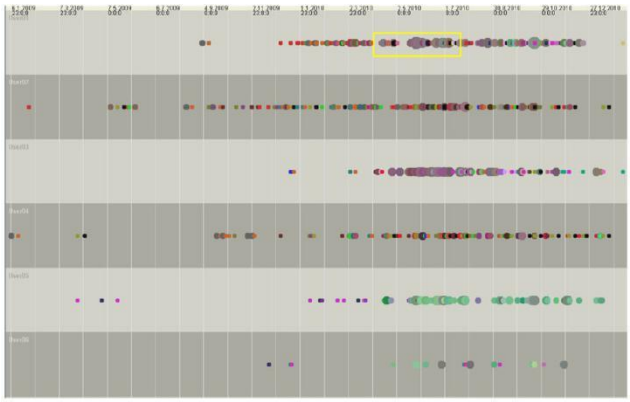
Gambar 21
Pengaruh cuaca ke produktivitas sumber daya manusia – Pemetaan
(Hermawan, 2013)

Gambar 22 menunjukkan keadaan visualisasi di *context analyzer*. Ada beberapa *cell* yang mempunyai warna. Warna tersebut merepresentasikan aktivitas yang paling sering dieksekusi dalam *cell* tersebut. Sedangkan baris mereprsentasikan *event logs* yang dilakukan oleh seseorang, dan setiap kolom merepresentasikan musim tertentu. Keduanya, SDM, dan musim akan diurutkan berdasar alphabet. Dalam gambar tersebut bisa juga dilihat bahwa ada di dalam *cell* terdapat *inner rectangle* dengan berbagai ukuran. Semakin besar ukurannya, maka *event log* yang di dalamnya semakin banyak, begitu pula sebaliknya semakin kecil ukurannya, maka *event log* yang di dalamnya semakin sedikit. Tanda silang di dalam *cell* menunjukkan ada lebih dari satu nilai aktivitas di dalam *cell* tersebut. Dari gambar 22 dapat dilihat bahwa User01, User03, dan User05 mengeksekusi sebagian besar *event log* yang ada pada log. Yang menarik lagi dapat dilihat pula bahwa sebagian besar *event log* dieksekusi pada masa musim semi ("spring") dan musim panas ("summer"). Di musim lainnya seperti pada musim gugur ("autumn") ataupun musim dingin ("winter") jumlah *event log* yang dieksekusi lebih sedikit, hal ini mungkin dikarenakan kebanyakan para klien organisasi ini tidak begitu aktif pada musim gugur dan dingin, karena pada musim ini udara relatif lebih dingin.



Gambar 22
Pengaruh cuaca ke produktivitas sumber daya manusia - Context analyzer
(Hermawan, 2013)

Menggunakan *dotted chart.* Gambar 23 menunjukkan visualisasi dari *dotted chart*. Setiap baris merepresentasikan sumber daya manusia, sedangkan setiap kolom merepresentasikan bulan. Dapat dilihat bahwa banyak titik pada User01, User 02, User03, User04, dan User05. Untuk memeriksa SDM mana yang paling aktif dalam suatu musim susah, karena pengguna harus menghitung kira-kira jumlah titik di dalam periode waktu tertentu. Beberapa titik dari musim berbeda bergabung menjadi satu, dan pengguna tidak mengerti lagi jumlah titik dalam satu musim. Oleh karena itu sangatlah susah untuk mengetahui pengaruh musim pada produktivitias sumber daya manusia di organisasi.



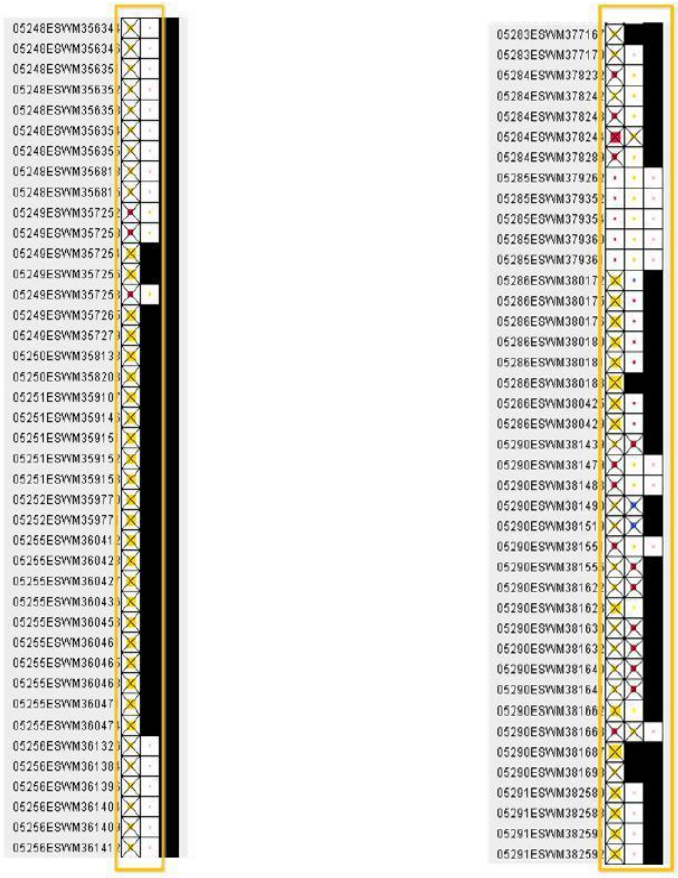
Gambar 23
Pengaruh cuaca ke produktivitas sumber daya manusia - Dotted chart
(Hermawan, 2013)

Dalam percobaan ini kami menggunakan log yang mempunyai atribut sebagai berikut:

Concept:name
Description
Batch
Concept:name
Error
Form
Id
Lifecycle:transition
Name
New_queue
O_resource
Org:resource
Priority
Queue
Rule_num
Server
Time:timestamp
Time_stamp
Type
workset

Interaksi antar sumber daya manusia dalam 9 jam

Menggunakan *context analyzer.* Pada gambar 24 dapat dilihat hasil visualisasi sebuah log menggunakan *context analyzer*. Setiap baris merepresentasikan *case* yang merupakan tipe kategorial, sedangkan kolom merepresentasikan atribut baru, yaitu *internal:unique resource*, tipe numerik. Tipe kategorial diurutkan berdasar alphabet, sedangkan tipe numerik diurutkan dari nilai yang paling kecil hingga yang paling besar yang berada dalam log. Dengan menggunakan setelan ini maka dapat diambil analisis sebagai berikut: Jika sebuah *case* mempunyai nilai pada kolom pertama maka *case* itu hanya mempunyai sebuah *event log*, dalam waktu rentang 9 jam setelah *event log* itu dieksekusi hanya ada satu sumber daya manusia yang mengeksekusi *event log* itu pada *case* yang sama. Jika sebuah *case* mempunyai nilai pada kolom pertama maka *case* itu hanya mempunyai sebuah *event log*, dalam waktu rentang 9 jam setelah *event log* itu dieksekusi hanya ada dua sumber daya manusia yang mengeksekusi *event log* itu pada *case* yang sama. Jika sebuah *case* mempunyai nilai pada kolom pertama maka *case* itu hanya mempunyai sebuah *event log*, dalam waktu rentang 9 jam setelah *event log* itu dieksekusi hanya ada tiga sumber daya manusia yang mengeksekusi *event log* itu pada *case* yang sama. Selain itu yang bisa didapat dari melihat gambar 24, di setiap baris kebanyakan mempunyai *inner rectangle* pada kolom pertama dan kedua, dan hanya beberapa saja yang mempunyai nilai pada kolom ketiga. Juga, dapat dilihat pula jumlah *event log* dengan melihat ukuran *innner rectangle* kebanyakan di kolom pertama dan kedua. Hal ini berarti bahwa dalam satu hari kerja atau 9 jam, seringnya *case* dilakukan oleh satu atau 2 orang sumber daya manusia saja.



Gambar 24
Interaksi antar sumber daya manusia dalam 9 jam - Context analyzer (Hermawan, 2013)

Menggunakan *dotted chart.* Kita akan meneliti interaksi antara sumber daya manusia dalam rentang waktu 9 jam, hal ini termasuk ke dalam konteks *instance*. Di *dotted chart*, sumbu x menunjukkan *timestamp*, dan sumbu y menunjukkan *instance*, sedangkan warna titik

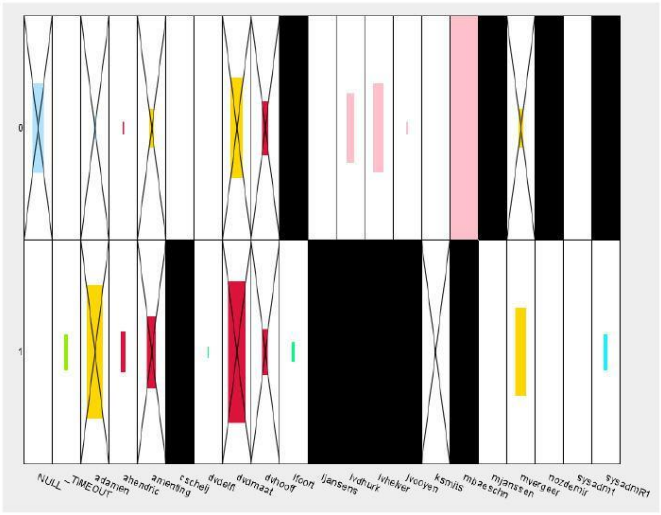
merepresentasikan sumber daya manusia yang mengeksekusi. Gambar 25 menunjukkan visualisasi dari *dotted chart*. Setiap baris merepresentasikan *instance*, setiap kolom merepresentasikan bukan, sedangkan warna berdasarkan *activity*. Dapat dilihat banyak titik dalam visualisasi tersebut. Sangat sulit untuk menarik kesimpulan dari hanya melihat sekilas gambar ini, tetapi jika kita berusaha melihat lebih detail menggunakan perbesaran (*zoom*) informasi penting lainnya banyak yang tidak tertangkap, seperti yang terlihat pada Gambar 26. Sangatlah susah untuk mengetahui pengaruh konteks *instance* menggunakan *dotted chart*.

Gambar 25
Interaksi antar sumber daya manusia dalam 9 jam - Dotted chart

Gambar 26
Interaksi antar sumber daya manusia dalam 9 jam - Dotted chart (perbesaran) (Hermawan, 2013)

Pengaruh prioritas di dalam alokasi sumber daya manusia

Menggunakan *context analyzer.* Gambar 27 menunjukkan visualisasi dari *context analyzer*. Terdapat banyak *cell*s dengan berbagai warna. Warna pada *cell* merepresentasikan *activity* yang paling sering muncul dalam satu *cell*, sedangkan baris merepresentasikan *event logs* yang dilakukan pada *priority* tertentu, kolom merepresentasikan *event logs* yang dilakukan oleh sumber daya manusia. *Priority* dan sumber daya manusia akan diurutkan berdasarkan abjad. Dalam *cell* dapat juga dilihat bahwa semakin besar *inner rectangle* maka akan semakin banyak jumlah *event log* di dalam *cell* tersebut. *Cross* atau tanda silang dalam *cell* menunjukkan bahwa ada lebih dari satu nilai activity di dalam *cell* tersebut. Gambar 27 dapat dilihat bahwa pada baris pertama *event logs* dengan priority 0 dan baris kedua menunjukkan *event logs* dengan *priority* 1. Sedangkan kolom merepresentasikan sumber daya manusia yang ada di dalam log. Beberapa sumber daya manusia hanya mengerjakan *case-case* pada *priority* yang sama, misal: csheij, ijansens dan ivdhurk mengerjkan *event logs* yang dieksekusi pada priority 0. Sedangkan ifoort, mjannssen, nozdemir, dan sysadmR1 pada priority 1. Temuan yang menarik lainnya adalah mbaeschn mengeksekusi sebagian besar *event logs* dalam log, hal ini dapat dilihat dari ukuran dari *inner rectangle* yang memenuhi *cell*, sedangkan sumber daya lainnya hanya mengisi sebagian dari *cell* saja. Dapat dilihat pula ada *cell* tanpa tanda silang, hal ini menunjukkan bahwa hanya ada satu tipe atribut pada activity, dalam *case* ini yaitu pada Domain:heus1.



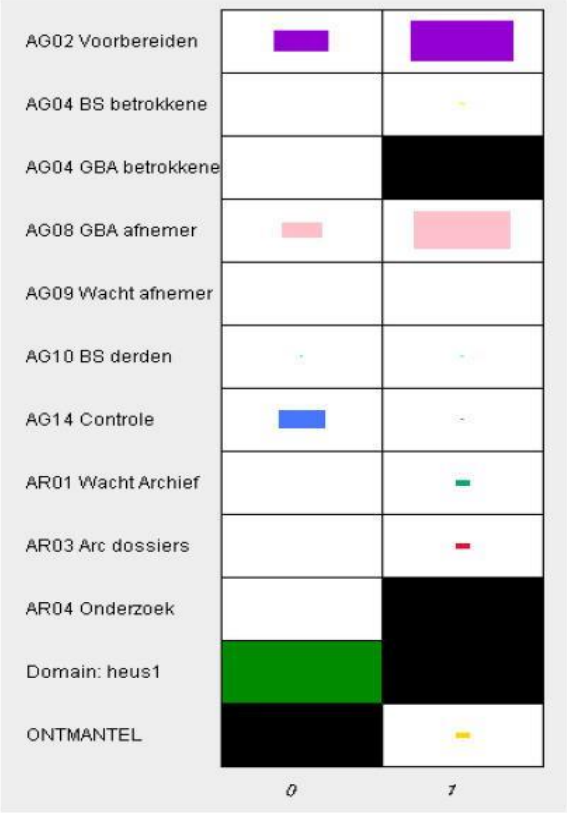
Gambar 27
Pengaruh prioritas di dalam alokasi sumber daya manusia - Context analyzer (Hermawan, 2013)

Menggunakan *dotted chart.* Karena keterbatasan pada *dotted chart* untuk meneliti pengaruh tipe *priority* ke alokasi sumber daya manusia tidak dapat dilakukan, karena *dotted chart* hanya bisa memvisualisasikan beberapa atribut saja: activity, *instance*, sumber daya manusia, atau *event log*.

Pengaruh priority ke activity

Menggunakan *context analyzer.* Gambar 28 menunjukkan visualisasi pada *context analyzer*. Ada beberapa *cell* dengan berbagai macam warna. Warna merepresentasikan *activity* yang paling banyak muncul dalam *cell* yang bersangkutan. Pada *case* ini baris merepresentasikan i pada log, sedangkan kolom menunjukkan *event log* yang dilakukan pada *priority* tertentu. *Activity* dan *priority* diurutkan berdasar abjad. Dapat dilihat pula bahwa ada *cell*s dengan *inner rectangle*, semakin besar *inner rectangle* dalam *cell*, maka makin banyak pula *event logs* di dalamnya. Gambar xx menunjukkan Domain:heus1 adalah *event log* yang

paling banyak dieksekusi di 0, sedangkan AG08 GBA afnemer dan AG02 Voorbereiden merupakan *event log* yang paling banyak dieksekusi pada 1. Yang menarik dari gambar ini adalah tidak semua *event logs* dieksekusi pada kedua priority. ONTMANTEL hanya dieksekusi pada priority 1, sedangkan Domain:heus1, AR04 Onderzoek, AG04 GBA betrokkene hanya dieksekusi pada priority 0. Temuan yang menarik lagi adalah jumlah dari *event logs* dari AG02 Voorbereiden, dan AG08 GBA afnemer pada *priority* 1 lebih tinggi daripada di priority 0. Hal ini juga terjadi pada AR01 Wacht Archief, dan AR03 Arc dossiers, tetapi pada kedua *case* ini perbedaannya hanya sedikit. Sedangkan pada *case priority* 1, *event logs* AG14 Control lebih banyak dieksekusi.



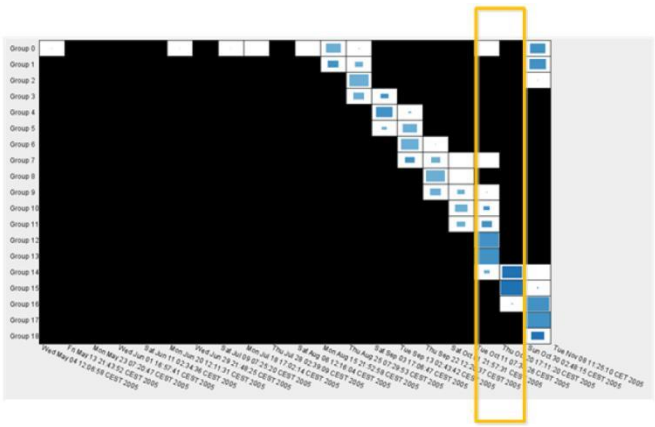
Gambar 28
Pengaruh priority ke activity - Context analyzer (Hermawan, 2013)

Menggunakan *dotted chart.* Mirip dengan *case* sebelumnya, karena keterbatasan pada *dotted chart* untuk meneliti pengaruh tipe *priority* ke alokasi sumber daya manusia tidak dapat dilakukan, karena *dotted chart* hanya bisa memvisualisasikan beberapa atribut saja: activity, *instance*, sumber daya manusia, atau *event log*.

Hubungan antara workload dan jumlah sumber daya manusia

Menggunakan *context analyzer.* Gambar 29 menunjukkan visualisasi pada *context analyzer*.

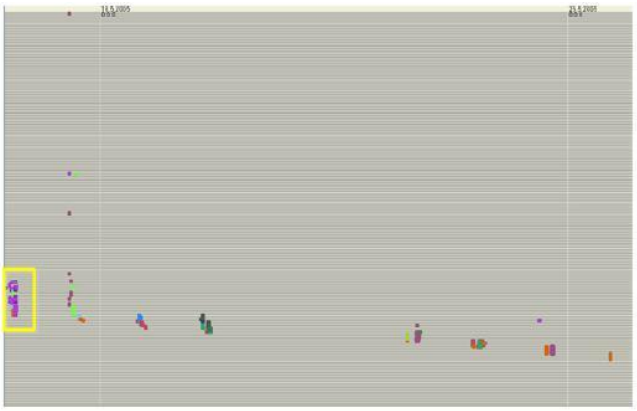
Terdapat beberapa *cell* dengan warna yang berbeda-beda. Setiap baris merepresentasikan *instance*, sedangkan kolom merepresentasikan rentang waktu dari *timestamp*. Kumpulan *instance* diurutkan berdasarkan abjad, sedangkan *timestamp* diurutkan dari nilai yang paling kecil hingga yang paling besar. Semakin besar *inner rectangle* dalam *cell*, maka makin banyak pula *event logs* yang terdapat pada *cell* tersebut. Dapat dilihat pula bahwa ada beberapa *cell* yang tidak ada *inner rectangle*, hal ini berarti tidak ada *event log* di dalamnya. Setiap warna dari *inner rectangle* mempunyai arti tersendiri. Semakin gelap warna biru, maka nilai rata-rata dari *internal:unique* resource makin tinggi pula. Secara garis besar Gambar 29 menunjukkan bahwa sebagian besar *cell* berwarna biru terdapat pada sisi kanan visualisasi. Hal ini berarti banyak *event log* yang dieksekusi pada periode waktu yang lebih dekat dengan waktu sekarang ini. Dengan banyak sekali jumlah *inner rectangle* pada 3 kolom terakhir, menunjukkan bahwa waktu tersibuk terdapat pada tiga kolom terakhir tersebut. Sebut saja kolom A, B, dan C merupakan 3 kolom terakhir, diurutkan dari sebelah kiri visualisasi. Jumlah *event log* pada kolom A lebih tinggi daripada kolom B. Tetapi kolom B mempunyai warna yang lebih gelap daripada kolom A, hal ini mengindikasikan bahwa sumber daya manusia di A bekerja lebih efisien daripada di B. Atau bisa juga mengindikasikan bahwa B mempunyai banyak *case* yang pada waktu yang relatif singkat. Hal ini bisa juga dapat digunakan sebagai pertimbangan apakah perlu untuk menambah jumlah sumber daya manusia pada perusahaan, sehingga tidak terjadi "bottleneck" yang mengakibatkan lambatnya eksekusi suatu proses.



Gambar 29
Hubungan antara workload dan jumlah sumber daya manusia - Context analyzer (Hermawan, 2013)

Menggunakan *dotted chart.* Gambar 30 menunjukkan visualisasi dari *dotted chart*. Setiap baris merepresentasikan sebuah *instance*, sedangkan kolom merepresentasikan bulan, pewarnaan merepresentasikan sumber daya manusia. Dapat dilihat banyak sekali titik-titik warna. Untuk meneliti hubungan

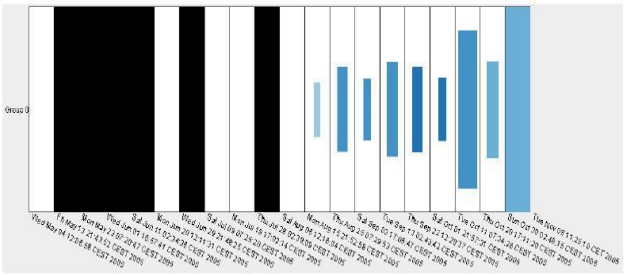
antara workload dengan jumlah sumber daya manusia pengguna harus dapat menghitung jumlah titik dengan warna yang berbeda pada rentang waktu tertentu. Misalkan saja kita dalam kotak kuning pengguna harus dapat menghitung jumlah titik pada *case* yang masih aktif saja. Dan membandingkan pada rentang periode yang berbeda. Untuk menarik kesimpulan menggunakan metode ini sangatlah susah, karena ketidakakuratan data yang dapat diambil.



Gambar 30
Hubungan antara workload dan jumlah sumber daya manusia - Dotted chart (Hermawan, 2013)

Hubungan antara case yang aktif dan jumlah sumber daya manusia

Menggunakan *context analyzer.* Gambar 31 menunjukkan visualisasi dari *context analyzer*. Terdapat sejumlah *cell* dengan berbagai warna. Baris merepresentasikan kumpulan dari *instance* dan kolom merepresentasikan rentang waktu dari *timestamp*. *Timestamp* diurutkan dari nilai yang paling kecil hingga yang paling besar. Semakin besar *inner rectangle* dalam suatu *cell* maka semakin banyak *event log* yang berada pada *cell* tersebut. Warna pada *inner rectangle* menunjukkan nilai dari rata-rata *internal:runningcase*, semakin gelap warna biru, maka semakin besar nilai rata-ratanya. Pada kolom keempat dan kelima dari kanan dapat dilihat bahwa ukuran dari *inner rectangle* relatif lebih kecil daripada lainnya, akan tetapi warnanya lebih gelap dibandingkan dengan sebagian besar *inner rectangle* lainnya. Hal ini menunjukkan bahwa pada rentang waktu tersebut *event log* yang dieksekusi pada *case* yang aktif pada waktu tersebut sangatlah padat.



Gambar 31
Hubungan antara case yang aktif dan jumlah sumber daya manusia - *Context analyzer* (Hermawan, 2013)

Menggunakan *dotted chart.* Gambar 32 menunjukkan visualisasi dari *dotted chart*. Setiap baris merepresentsikan *instance*, sedangkan kolom merepresentasikan bulan, warna dari titik merepresentasikan sumber daya manusia. Untuk meneliti *case* mana yang aktif dalam waktu tertentu tidak mudah, karena pengguna harus menghitung secara manual satu persatu. Dapat ditarik kesimpulan menggunakan metode ini untuk meneliti hubungan antara *case* yang aktif dengan jumlah sumber daya manusia susah.

Gambar 32
Hubungan antara case yang aktif dan jumlah sumber daya manusia - Dotted chart (Hermawan, 2013)

Performa berdasar ukuran log

Untuk membuktikan bahwa pendekatan yang telah kita buat mempunyai performa yang baik pada *case* nyata, kami bandingkan *context analyzer* dengan plugin *dotted chart*. Evaluasi dilakukan dengan menggunakan berbagai macam *event log* dari dunia nyata. Deskripsi dari setiap log dapat dilihat pada tabel 4.

Tabel 4
Event log yang digunakan sebagai tes performa

Event log	Jumlah event
Afchriften	1343

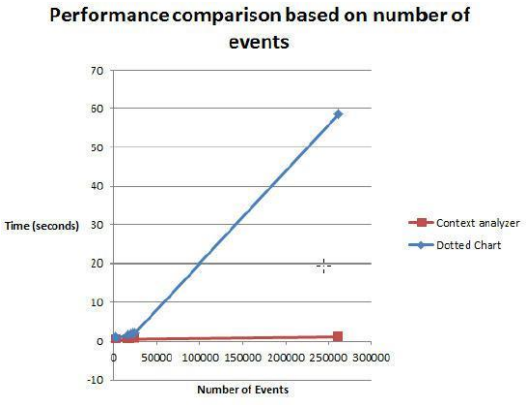
Bouw	2803
BPI challenge	262200
Wabo1	22130
Wabo2	16039
Wabo3	2442
Wabo4	18549

Perbandingan performa antara context analyzer dengan dotted chart

Performa berdasar ukuran log untuk setiap *event log* kami rekam waktu yang dibutuhkan oleh *context analyzer* dan *dotted chart* untuk menvisualisasikan log. Keduanya diberikan parameter yang sama. Axis x kedua plugin akan menampilkan *timestamp*, sedangkan y axis akan menampilkan instan. Jumlah kolom pada *context analyzer* diset pada 7 kolom. Dari Gambar 33 dapat dilihat bahwa secara umum, *context analyzer* mempunyai performa yang lebih baik daripada *dotted chart*. Hal ini dikarenakan *dotted chart* harus memetakan setiap *event log* pada visualiasasi, sedangkan pada *context analyzer* mempunyai proses yang berbeda, tidak semua *event log* dipetakan pada visualisasi, tetapi yang dilakukan adalah menjumlah nilai yang ada pada *event log* setelah itu ditampilkan pada visualisasi.

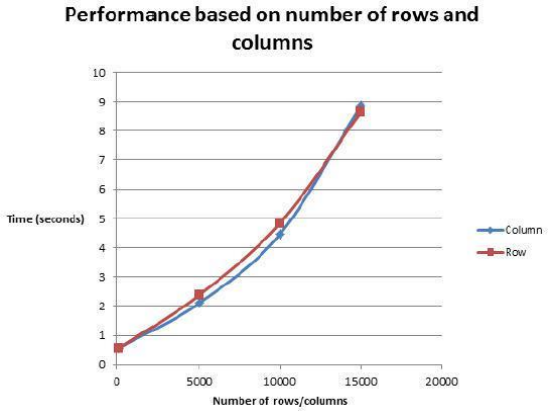
Perbandingan performa berdasar baris dan kolom

Pada percobaan kedua, kami mencoba meneliti performa *context analyzer* berdasarkan jumlah baris dan kolom. Log dari Bpi challenge digunakan sebagai log test. Dalam eksperimen ini, jumlah kolom dibuat dengan nilai 100, dan jumlah baris diubah sesuai dengan parameter saat itu. Hal ini dilakukan pula untuk eksperimen dengan baris, jumlah baris dibuat dengan nilai 100 dan jumlah kolom diubah sesuai dengan parameter saat itu.



Gambar 33
Perbandingan performa *context analyzer* dengan dotted chart (Hermawan, 2013)

Dari gambar 33 dapat dilihat bahwa kedua eksperimen menghasilkan hasil yang serupa. Setelah jumlah kolom atau baris 20.000, maka *context analyzer* akan kekurangan memori.



Gambar 34
Perbandingan performa berdasarkan jumlah baris dan kolom
(Hermawan, 2013)

Penutup

Dalam paper ini telah dievaluasi masalah yang terjadi ketika mendapatkan informasi melibatkan konteks pada *event log*. Masalah pertama yang dipecahkan adalah proses identifikasi tipe konteks mana yang berguna yang didapat dari *event log* dan bagaimana mendapatkannya. Kita juga telah menunjukkan bahwa konteks yang didapat seperti konteks *instance* dan konteks sosial dapat didapatkan dari proses *filtering, partitioning, selecting*, dan *aggregating* dalam proses log. Masalah yang dipecahkan berikutnya adalah bagimana mengatasi hilangnya data. Pendekatan secara heuristik dilakukan untuk mengatasi hal ini. Kami juga telah memberikan algoritma yang dapat digunakan untuk mengatasi hilangnya atau tidak tepatnya nilai dari *timestamps* Masalah terakhir yang dipecahkan adalah bagaimana memvisualisasi informasi agar dapat dimengerti. Kami menvisualisasikan dalam bentuk grid yang beraneka warna. Setiap grid mereprsentasikan data sesuai informasi yang ingin didapatkan.

Daftar Pustaka

Aalst, W.M.P. van, dan Dustdar, S. (2012). Process mining put into context. IEEE Internet Computing, 16(1):82-86.

Westergaard, M. and Wynn, M. T. (2012). Process mining manifesto. In BPM 2011 Workshops, Part I, volume 99, pages 169

Dey, A.K. (2001). Understanding and using context. *Personal and Ubiquitous Computing(*5:4-7)

Eren, C. (2012). Providing running case predictions based on contextual information. Master thesis. Eindhoven University of Technology.

Hao, M.C., Dayal, U., and Casati(2004), F.. Visual mining business service using pixel bar charts. In Erbacher,

B.F., Chen, P.C., Roberts, J.C., Grohn, M.T., and Borner, K., *Visualization and Data Analysis*, volume 5295 of SPIE Conference, pages 117{123, June 2004.

Hermawan, A.A. (2013) Context Analysis of Business Processes Based on *Event logs*. Master Thesis Eindhoven University of Technology.

Ploesser,K., , Peleg, M., Soer, P., Rosemann, M., and Recker, J.C., Learning from context to improve business processes. BPTrends, 6(1):1{7, January 2009.

Rosemann, M, Recker, J.C., and Flender, C. (2008). Contextualisation of business processes. *International Journal of Business Process Integration and Management*, 3(1):47{60, July 2008.

Rosemann, M, Recker, J.C., and Flender, C., and Ansell P.D. (2006) Understanding context-awareness in business process design. In Su Spencer and Adam Jenkins, editors, 17th A*ustralasian Conference on Information Systems*,Adelaide, Australia. Australasian Association for Information Systems.

Schilit , B. dan Theimer, M. (1994) Disseminating active map information to mobile hosts. IEEE Network, 8:22

Schmidt , A. (2010). Implicit human computer interaction through context. Technical report, Personal Technologies.