EVALUASI PENDEKATAN PEMBANGUNAN TRACEABILITY LINK DALAM EVOLUSI PERANGKAT LUNAK

Noor Fitria A.[1], **Nuru Aini**[2]
[1, 2] Jurusan Teknik Informatika, Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember, Surabaya
Email: noorfitria@cs.its.ac.id[1]

ABSTRAK

Traceability merupakan hal penting pada proyek perangkat lunak, terutama pada proyek skala besar. Traceability berfungsi untuk mengetahui ketelusuran antar artefak dalam fase-fase yang berbeda (analisis kebutuhan, analisis desain, dan analisis implementasi) maupun antara artefak dan pihak pengembang yang terlibat. Sistem traceability otomatis diperlukan untuk membangun ketelusuran antar artefak. Penelitian ini bertujuan untuk mengeksplorasi sejumlah literatur pendekatan terbaru yang digunakan untuk membangun traceability link. Eksplorasi literatur mengacu pada taksonomi berbasis evolusi perangkat lunak terhadap sejumlah mekanisme karakterisasi perubahan dan faktor-faktor yang mempengaruhi mekanisme. Hasil penelitian dapat digunakan untuk mengidentifikasi bagaimana pendekatan tersebut dapat mendukung evolusi perangkat lunak serta memberikan garis besar dari kriteria yang dibutuhkan untuk membangun metode traceability yang lebih baik. Kesimpulan dari penelitian ini adalah variasi faktor suatu pendekatan tidak berbeda jauh dengan pendekatan lainnya kecuali jika terdapat perbedaan pada faktor temporal.

Kata Kunci: *Taksonomi evolusi perangkat lunak, Traceability link.*

1. PENDAHULUAN

Traceability (ketelusuran) adalah properti dari desain dan pengembangan perangkat lunak yang menghubungkan setiap artefak abstrak dengan artefak teknis maupun sebaliknya [1]. Pada proyek perangkat lunak dalam skala besar, *traceability* sangat penting untuk mengetahui ketelusuran antar artefak dalam fase-fase yang berbeda (analisis kebutuhan, analisis desain, dan analisis implementasi). Selain itu, *traceability* dapat mengetahui ketelusuran antara artefak dan pihak pengembang yang terlibat. Proses pembangunan ketelusuran yang dilakukan secara manual akan menghabiskan banyak waktu dan sangat berpotensi terjadi kesalahan sehingga informasi menjadi tidak lengkap. Oleh karena itu, sebuah sistem *traceability* secara otomatis diperlukan untuk membangun ketelusuran antar artefak.

Penelitian tentang pendekatan untuk membangan *traceability* telah dimulai sejak tahun 1984 oleh Dorfman dan Flynn dengan membangun kakas untuk *traceability* dengan nama ART (*Automated Requirements Traceability*) [2]. Hingga saat ini, penelitian *traceability* masih terbuka dan masih memiliki banyak tantangan, terutama dalam meminimalisasi kesalahan keterhubungan antar-artefak [3]. Penelitian yang dilakukan oleh Siti Rochimah, dkk pada tahun 2007 mengevaluasi berbagai pendekatan untuk *traceability* dengan menggunakan kerangka kerja taksonomi evolusi perangkat lunak yang diusulkan oleh Buckley [4].

Tujuan penelitian ini adalah untuk melanjutkan evaluasi pendekatan yang berkembang untuk membangun *traceability link* dengan menggunakan kerangka kerja taksonomi seperti yang telah dilakukan oleh Siti [5]. Evaluasi berfokus untuk mengetahui kemampuan pendekatan dalam upaya mendukung evolusi perangkat lunak. Hasil evaluasi dapat digunakan untuk mengidentifikasi bagaimana pendekatan pembangunan *traceability link* yang berkembang dapat mendukung evolusi perangkat lunak. Berdasarkan hasil evaluasi tersebut, peneliti berikutnya dapat mengetahui gambaran besar dan kriteria yang dibutuhkan untuk mengembangkan pendekatan *traceability* yang lebih baik.

Makalah ini disusun dalam lima subbab. Subbab pertama adalah pendahuluan. Subbab kedua adalah kajian pustaka mengenai metode pendekatan *traceability*. Subbab ketiga adalah penjelasan tentang teknik evaluasi. Subbab keempat adalah evaluasi metode pendekatan *traceability*. Subbab kelima adalah rangkuman dari semua hasil evaluasi. Subbab keenam menyampaikan tentang kesimpulan.

2. EKSPLORASI LITERATUR PENDEKATAN TERBARU MENGENAI TRACEABILITY LINK

Penelitian ini mencakup proses eksplorasi literatur sejumlah pendekatan untuk membangun *traceability link*. Literatur diperoleh dari jurnal Sciencedirect dan IEEE. Subbab ini menjelaskan secara garis besar mengenai beberapa pendekatan terbaru dalam hal tersebut.

2.1 Pendekatan Orthogonal Information Retrieval

Telah banyak pendekatan yang dikembangkan untuk membangun sebuah sistem *traceability link* dengan pendekatan *Information Retrieval*, seperti *Vector Space Model* (VSM) menggunakan model vektor term dari dokumen perangkat lunak, *Jensen and Shannon* (JS) menggunakan nilai probabilitas dokumen, *Latent Semantic Indexing* (LSI) menggunakan perpaduan antara vektor term dan nilai probabilitas, *Latent Dirichlet Allocation* (LDA) seperti LSI yang menggunakan kedua pendekatan tersebut untuk menghasilkan topik, *Relational Topic Modelling* (RTM) yang merupakan pengembangan dari LDA yang menghubungkan antara topik satu dengan topik lainnya. Sehingga dapat disimpulkan bahwa pendekatan *information retrieval* (IR) terdiri dari dua metode antara lain metode berbasis vektor maupun probabilitas. Pendekatan IR yang digunakan dalam membangun *traceability link* bekerja pada dokumen yang berbentuk tulisan, dalam perangkat lunak dokumen tersebut direpresentasikan dalam dokumen kebutuhan perangkat lunak.

Pendekatan *Orthogonal Information Retrieval* merupakan penelitian yang menggabungkan antara pendekatan IR satu dengan metode IR lainnya. Dalam penelitian Malcom dkk [6] dilakukan penggabungan metode-metode IR tersebut, misalnya JS dan RTM, VSM dan RTM. Hasil dari penelitian ini menunjukkan bahwa *Orthogonal Information Retrieval* lebih baik jika dibandingkan dengan metode IR yang berdiri sendiri.

Tahap pengerjaannya adalah preproses dokumen artefak yang terdiri dari *tokenizing, stopward removal, stemming.* Setelah itu, proses berikutnya adalah pembuatan vektor term untuk selanjutkan dikerjakan sesuai metode yang ingin diterapkan. Evaluasi dilakukan dengan ukuran presisi, *recall*, dan F-*measure.*

2.2 Pendekatan Bayesian

Omoronyia dkk mengusulkan pendekatan *traceability link* berbasis konsep Bayesian [7]. Pendekatan ini melihat permasalahan *traceability* sebagai permasalahan statistika. Pendekatan ini dapat berjalan dalam masalah yang tidak pasti, ketidakjelasannya dan terdapat banyak kesalahan. Dalam kasus ini, *traceability link* yang dibangun adalah ketelurusan antara *use case* dengan fungsionalnya berdasarkan aktivitas pengembangnya. Artefak-artefak tersebut dimodelkan keterhubungan-nya dengan node-node dalam bentuk *Bayesian Belief Network* (BBN). Penentuan *traceability link* ditentukan oleh nilai posterior probability.

Langkah yang harus dilakukan untuk menghubungkan artefak yang terkait dengan *use case* yang diamati antara lain: Langkah pertama membangun struktur jaringan berdasarkan dataset yang diobservasi. Langkah ini dikerjakan berdasarkan informasi mengenai interaksi antara pengembang dengan artefaknya. Interaksi tersebut mencakup aktivitas *view, create, update*, dan *delete.* Langkah kedua mengestimasi nilai PDT (tabel ditribusi probabilitas) lokal setiap nodenya. Estimasi ini juga berdasarkan informasi interaksi pengembang dengan artifaknya. Langkah berikutnya menghitung nilai probabilitas posterior dari artefak yang relevan dengan *usecase* yang diamati berdasarkan nilai PDT. Evaluasi metode ini dilakukan dengan ukuran presisi, *recall*, dan F-*measure.*

2.3 Pendekatan Linier

Omoronyia dkk juga mengusulkan pendekatan *traceability link* berbasis akumulasi linier terhadap nilai relevan dari sejumlah entitas. Pendekatan linier melihat permasalahan *traceability* sebagai permasalahan matematika. *Traceability link* ditentukan dengan cara menurunkan atribut secara akumulatif. Atribut yang dimaksud adalah tipe interaksi entitas dan ukuran entitas lainnya. Metode tersebut digunakan untuk pengurutan peringkat yang relevan terhadap *use case*, pihak pengembang, dan artefak *code.* Selain memantau terjadinya interaksi *view, update*, dan *create* dimana hal tersebut mempengaruhi *state* pada entitas (sebagaimana disebutkan pada metode Bayesian), metode Linier juga memperhitungkan ukuran konteks entitas-entitas yang terlibat pada tiap interaksi.

Pendekatan Linier menggunakan konsep mengenai pengembangan konteks kerja, pemantauan interaksi yang diasosiasi dengan entitas, serta konsep *sphere of influence* (SOI). Pada pendekatan ini, pemodelan Linier membangun nilai-nilai relevansi entitas secara akumulatif. Nilai relevansi merupakan nilai kelayakan suatu entitas untuk dimasukkan dalam *traceability link.* Nilai relevansi kumulatif tersebut kemudian diderivasi dalam bentuk *history mode. History mode* merupakan riwayat interaksi yang dilakukan terhadap artifak oleh pengguna, baik berupa *view, update*, dan *create.* Langkah terakhir adalah membentuk daftar yang relevan tentang pengembang, artefak *code*, ataupun *use case* yang terkait dengan *particular trace link.* Pendekatan ini bertujuan menghasilkan informasi *traceability* yang akurat, *real-time*, dan mencakup usaha kerja tiap individu pengembang; sebuah indikasi yang menunjukkan *use case* dan artefak mana yang mengonsumsi biaya terbesar di antara keseluruhan pengembang. Evaluasi biasanya dilakukan dengan ukuran presisi, *recall*, dan F-*measure.*

2.4 Pendekatan Pragmatis

Markus Aleksy dkk mengusulkan pendekatan pragmatis untuk membangun *traceability link* secara otomatis [8]. Pendekatan ini memiliki kelebihan yaitu: adanya proses *recovery*, validasi, dan *update traceability link* secara otomatis; dan integrasi kakas bantu dalam hal memudahkan untuk penggunaan

yang lebih luas dan lingkungan pengembangan yang lebih heterogen. Pendekatan ini digunakan pada konteks *Model-Driven Software Development* (MDSD). Pendekatan ini bekerja berdasarkan dua asumsi dasar: pertama, semua kebutuhan dalam sistem terpenuhi berikut dengan model-model yang merefleksikan kebutuhan tersebut; kedua, setiap model diimplementasikan dalam sebuah kode sehingga setiap modul bersifat unik.

Pada pendekatan ini, desain konseptual dibangun berdasarkan asumsi kebutuhan direpresentasikan dalam bentuk teks serta memiliki *identifier* yang unik. Asumsi lainnya yaitu setiap elemen model juga memiliki *identifier* yang unik sehingga tidak terjadi ambiguitas antar keduanya. Selain itu, kode dibangun secara otomatis sehingga setiap kode dibangun berdasarkan satu atau lebih model.

Penelitian ini mengimplementasi suatu prototipe untuk menunjukkan kemampuan bekerja pendekatan tersebut dalam lingkungan pengembangan yang lebih umum. Prototipe yang digunakan dikembangkan dalam *platform* Eclipse. Selain itu, pendekatan ini juga menggunakan kakas bantu kolaborasi, misalnya CodeBeamer untuk menangkap kebutuhan dalam sistem. Dengan demikian, pendekatan ini mudah pada praktiknya mudah untuk diaplikasikan. Prototipe tersebut juga menunjukkan bahwa pendekatan tersebut dapat bekerja secara otomatis dan tidak selalu memerlukan persyaratan tambahan serta tidak perlu menggunakan *trace* manual.

3. KERANGKA KERJA EVALUASI

Buckley [4] mengusulkan taksonomi berbasis evolusi perangkat lunak terhadap sejumlah mekanisme karakterisasi perubahan dan faktor-faktor. Taksonomi tersebut dikelompokkan sebagai berikut: Properti temporal, yakni sebuah dimensi dalam evolusi perangkat lunak yang mencakup aspek waktu dalam perubahan. Properti objek perubahan, yang mencakup aspek lokasi terjadinya perubahan, misalnya bagian mana dalam perangkat lunak yang dapat diubah. Properti sistem yang mencakup karakteristik dalam perangkat lunak ketika terjadi perubahan. Terakhir, properti dukungan perubahan yang mencakup mekanisme dukungan ketika perangkat lunak sedang diubah.

Properti temporal mencakup waktu perubahan, riwayat perubahan, dan frekuensi perubahan. Waktu perubahan mengategorikan waktu perubahan perangkat lunak antara lain *compile time, load time, dan run time*. *Compile-time* merupakan perubahan statis yang mana perubahan perangkat lunak bergantung pada *source code* sehingga perlu *re-compile* untuk melakukan perubahan. *Load-time* adalah perubahan perangkat lunak yang terjadi ketika berada dalam sistem yang *executable*. Dan *run-time* adalah perubahan perangkat lunak dinamis yang terjadi selama sistem dijalankan. Riwayat perubahan didukung oleh kemampuan memperbarui perangkat lunak. Riwayat perubahan yang *sequential* tidak mengijinkan perubahan dilakukan secara bersamaan sedangkan *parallel* mengijinkan perubahan perangkat lunak dilakukan secara bersaman oleh beberapa orang. Frekuensi perubahan mendeteksi apakah perangkat lunak dapat diubah pada interval secara kontinyu, periodik, ataupun tak beraturan.

Objek perubahan terdiri dari artefak yang mengindikasi artefak apa yang dapat diubah oleh perangkat lunak. Faktor g*ranularity* mendeskripsikan skala artefak yang dapat berubah dengan level *coarse* (sistem atau paket subsistem), *medium* (terjadi pada level objek/kelas), dan *fine granularity* (terjadi pada variabel, metode, dan kode). Faktor *impact* yang mengindikasi dampak terhadap perubahan, antara lain berdampak secara lokal (*local)* ataupun berdampak secara sistemik (*system-wide impacted).* Sedangkan faktor *change propagation* mencatat apakah perangkat lunak tersebut selama perubahan memiliki kapabilitas untuk mempropagasi menuju bagian lain dalam artefak. Propagasi tersebut diimplementasikan oleh fitur propagasi perubahan antara lain *Change Impact Analysis, Traceability Analysis*, dan *Effort Estimation*.

Properti sistem mencakup karakteristik dalam perangkat lunak ketika terjadi perubahan. Properti sistem terdiri dari faktor ketersediaan yang mengindikasi apakah perangkat lunak harus tersedia secara permanen atau parsial selama perubahan sedang dilakukan. Faktor keaktifan yang mengindikasi apakah perangkat lunak tersebut reaktif, misalnya perubahan diatur secara eksternal, ataupun proaktif, misalnya perubahan bergantung dari perangkat lunak itu sendiri. Faktor keterbukaan yang mengindikasi apakah perangkat lunak tersebut merupakan suatu sistem yang terbuka, misalnya dibangun untuk dimungkinkan adanya ekstensi-ekstensi, atau malah merupakan sistem yang tertutup, misalnya lagi tidak mengijinkan kerangka kerja untuk ditambahi ekstensi. Faktor keamanan membedakan antara keamanan statis dan dinamis. Keamanan statis dapat dipastikan hanya ketika perangkat lunak sedang di-*compile* dan hanya ada pada bidang-bidang yang telah ditentukan. Sedangkan keamanan dinamis tersedia selama sistem telah dipasang. Keamanan dinamis dapat mencegah kemungkinan ada gangguan ketika sistem sedang berjalan.

Dukungan perubahan terdiri dari tiga kriteria. Pertama, *degree of automation* yang membedakan dukungan perubahan antara yang otomatis, semi-otomatis, dan manual. Kedua, *degree of formality* yang mengindikasi apakah dukungan perubahan diimplementasi berdasarkan sejumlah *underlying mathematical formalism* atau tidak. Ketiga, tipe perubahan yang membedakan antara perubahan struktural dan semantik.

4. HASIL EVALUASI

Pada bab ini, akan dijelaskan hasil evaluasi terhadap metode yang ada berdasarkan parameter yang telah ditulis pada bab sebelumnya.

4.1 Properti Temporal (Temporal Properties: When)

Pada dasarnya, pendekatan Orthogonal IR melakukan preproses yang sama seperti halnya pendekatan IR lainnya. Hal ini dikarenakan pendekatan tersebut merupakan penggabungan hasil dari dua jenis pendekatan IR tertentu. Pendekatan tersebut membangun *traceability link* dengan cara memproses artefak perangkat lunak dalam bentuk format berkas teks. Dengan demikian, tiap format berkas lainnya akan ditransformasikan terlebih dahulu ke dalam format berkas teks. Hal ini dilakukan sebelum melakukan tahap preproses. Jika perubahan struktur *link* akan terjadi pada suatu artefak perangkat lunak, maka artefak yang akan mengalami perubahan tersebut harus ditransformasikan terlebih dahulu dalam format berkas teks, begitu juga seterusnya pada artefak lain yang akan mengalami perubahan juga. Tahap berikutnya adalah membangun versi baru *traceability link* untuk perangkat lunak tersebut berdasarkan perubahan yang terjadi. Ketika ada perubahan interaksi antar artefak dan pengembangnya selama proses pengembangan perangkat lunak. *Traceability* baru tidak langsung dibangun dari elemen-elemen yang *executable* dalam perangkat lunak. Oleh karena itu, tiap-tiap perubahan yang terjadi dalam perangkat lunak harus mengalami proses *recompiled* untuk mendapatkan versi yang baru. Sehingga, pendekatan IR dikatakan memiliki kriteria waktu perubahan pada saat *compile* berlangsung. Perubahan yang dilakukan tidak bergantung pada *link* yang telah ada sebelumnya. Karakteristik tersebut memungkinkan pendekatan ini untuk melakukan mekanisme *versioning*. Perubahan perangkat lunak dapat terjadi sesuai dengan kebutuhan evolusi. Jika terjadi perubahan pengembangan perangkat lunak yang menyebabkan perubahan interaksi antar artefak dan pengembang serta dibutuhkan *traceability link* yang baru maka pendekatakan ini digunakan lagi. Oleh karena itu, perubahan yang dilakukan oleh pendekatan ini terjadi pada periode yang tidak beraturan (*arbitrary).*

Pendekatan bayesian berjalan berdasarkan data interaksi antara artefak dengan pengembang. Data tersebut digunakan untuk membangun *traceability link* dengan perhitungan nilai PDT. Perubahan struktur *link* terjadi ketika ada perubahan interaksi antar artefak dan pengembangnya selama proses pengembangan perangkat lunak. *Traceability* yang baru dapat dibangun melalui perhitungan ulang data-data tersebut. Oleh karena itu, pendekatan ini memiliki waktu perubahan saat *compile* berlangsung. Pendekatan bayesian membangun *traceability link* secara struktural sehingga perubahan berlangsung secara *sequential.* Perubahan perangkat lunak dapat terjadi sesuai dengan kebutuhan evolusi. Oleh karena itu, frekuensi perubahan metode ini terjadi secara tidak beraturan (*arbitrary*).

Pendekatan linier berjalan berdasarkan data interaksi antara artefak dengan pengembang. Selain memantau terjadinya interaksi *view, update,* dan *create* dimana hal tersebut mempengaruhi *state* pada entitas, pendekatan linier juga memperhitungkan ukuran konteks entitas-entitas yang terlibat pada tiap interaksi. Perubahan struktur *link* terjadi ketika ada perubahan interaksi antar artefak dan pengembangnya selama proses pengembangan perangkat lunak. *Traceability link* yang baru dapat dibangun melalui perhitungan ulang data-data tersebut. Oleh karena itu, pendekatan ini memiliki waktu perubahan saat *compile* berlangsung. Pendekatan linear juga membangun *traceability link* secara struktural sehingga perubahan berlangsung secara *sequential.* Perubahan perangkat lunak dapat dilakukan sesuai dengan kebutuhan evolusi. Oleh karena itu, frekuensi perubahan metode ini terjadi tidak beraturan (*arbitrary*).

Pendekatan pragmatis dibangun berdasarkan konsep MDSD yang menggunakan asumsi-asumsi yang telah disebutkan sebelumnya, yaitu setiap kode dibangun berdasarkan model tertentu dan dilakukan secara otomatis pada saat aplikasi CodeBeamer dan Eclipse dijalankan. Hal ini mengindikasikan bahwa waktu perubahan pendekatan pragmatis dilakukan pada saat *load-time*. Berdasarkan asumsi itu pula, pendekatan ini juga memiliki riwayat perubahan paralel sehingga *traceability link* dibangun secara otomatis. Frekuensi perubahan pendekatan ini berjalan secara periodik, yaitu setiap ada perubahan *traceability link* yang baru akan dibangun. Tabel 1 di bawah ini menampilkan hasil evaluasi terhadap aspek properti temporal berdasarkan penjelasan di atas.

4.2 Objek Perubahan (Object of Change: Where)

Pendekatan Orthogonal IR yang dilakukan oleh Malcom Gethers membangun *traceability link* antara berkas dokumentasi dan *source code*, antara dokumen kebutuhan level tinggi dan level rendah, antara dokumen kebutuhan dan artefak UML, serta antara *source code* dan *test cases*. Oleh karena itu, cakupan artefak pada pendekatan Orthogonal IR mencakup artefak *high-level* dan *low-level*. Artefak yang dirubah adalah berupa modul sehingga faktor level *granularity* pendekatan IR berupa *fine-grained.* Pendekatan ini juga berdampak secara *local dan*

Tabel 1. Evaluasi terhadap Dimensi 'Properti Temporal'

Pendekatan Traceability Link	Properti Temporal							
	Waktu			Riwayat			Frekuensi	
	Compile	Load	Run	Versioning	Sequential	Paralel	Periodik	Tak beraturan
Orthogonal	√			√				√
Bayesian	√				√			√
Linear	√				√			√
Pragmatis		√				√	√	

system-wide impacted. Seperti yang dijelaskan sebelumnya bahwa Malcom Gethers melakukan penelitian pendekatan bayesian untuk membangun *traceability link* maka keluaran dari pendekatan ini adalah *traceability link.* Propagasi perubahan tidak sampai tahap *change impact analysis* dikarenakan pendekatan ini tidak memberikan gambaran bagian mana yang terkena dampak perubahan.

Pembangunan *traceability link* dengan pendekatan Bayes yang dilakukan oleh Inah Omoronyia menggunakan data interaksi antara pengembang dan artefak. Interaksi artefak yang dimaksud adalah proses *create, edit, update, delete*yang dilakukan terhadap modul. Sehingga objek perubahan dalam pendekatan bayesian terjadi pada artefak *low-level.*

Oleh karena artefak yang dirubah berupa modul, maka level *granularity* berupa *fine-grained* dan akibat dari metode ini evolusi perangkat lunak terjadi secara *local.* Seperti yang dijelaskan sebelumnya bahwa Inah Omoronyia melakukan penelitian pendekatan bayesian untuk membangun *traceability link* maka keluaran dari pendekatan ini adalah *traceability link.* Propagasi perubahan tidak sampai tahap *change impact analysis* dikarenakan pendekatan ini tidak memberikan gambaran bagian mana yang terkena dampak perubahan.

Seperti pada pendekatan bayes, pembuatan *traceability link* dengan pendekatan linear yang dilakukan oleh Inah Omoronyia menggunakan data interaksi antara pengembang dan artefak. Interaksi artefak yang dimaksud adalah proses *create, edit, update, delete* yang dilakukan terhadap modul. Sehingga objek perubahan dalam pendekatan bayesian terjadi pada artefak *low-level.* Oleh karena artefak yang dirubah berupa modul, maka level *granularity* berupa *fine-grained.* Dampak dari perubahan dengan pendekatan ini hanya terjadi secara *local.* Pendekatan linear mempropagasi perubahan sampai tahap *traceability link*. Propagasi perubahan tidak sampai tahap *change impact analysis* dikarenakan pendekatan ini tidak memberikan gambaran bagian mana yang terkena dampak perubahan.

Pendekatan pragmatis bekerja pada objek artefak *high level* dan *low level* karena *traceability link* dibangun berdasarkan dokumen kebutuhan, model, dan kode. Berdasarkan objek artefak itulah, skala perubahan (*granularity*) pendekatan pragmatis adalah *coarse*. Akibat dari perubahan ini bersifat sistemik yaitu merubah secara sistem, dengan kata lain pendekatan pragmatis berdampak pada perubahan *System-wide*. Pendekatan ini tidak berdampak secara lokal karena pendekatan ini dibangun berdasarkan asumsi bahwa setiap kode dibangun secara otomatis dengan menggunakan Code Beamer. Propagasi perubahan adalah *change impact analysis* serta *traceability analysis.* Hal ini dikarenakan pendekatan tersebut memberikan gambaran secara eksplisit bagian mana yang terkena dampak perubahan. Tabel 2 di bawah ini menampilkan hasil evaluasi terhadap aspek objek perubahan berdasarkan penjelasan di atas.

4.3 Properti Sistem (System Properties: What)

Hampir seluruh perangkat lunak berjalan secara parsial. Dengan demikian, perangkat lunak akan berhenti ketika akan dilakukan perubahan. Begitupun dengan semua pendekatan yang dibahas, ketersediaannya bersifat parsial karena setiap pendekatan tidak menggunakan waktu perubahan secara *run-time* dan harus di-*compile* ulang untuk membangun *traceability link* yang baru. Sifat parsial tidak berlaku pada pendekatan pragmatis. Hal ini dikarenakan kita tidak perlu melakukan *recompile* untuk membangun *traceability link* yang baru sehingga pendekatan ini berjalan secara permanen.

Sistem perangkat lunak dapat bersifat reaktif atau proaktif. Reaktif adalah ketika pihak eksternal diperlukan untuk melakukan perubahan. Sedangkan sifat proaktif adalah ketika sistem dapat berubah secara otomatis. Pendekatan orthogonal, bayesian, dan linear memerlukan pihak eksternal untuk membangun link, perlu pihak lain untuk melakukan *compile.* Oleh karena itu, sifat keaktifan semua pendekatan *traceability link* adalah reaktif. Sedangkan pendekatan pragmatis bersifat proaktif karena dapat bekerja secara otomatis untuk membangun *link.* Semua pendekatan *traceability link* mendukung keterbukaan sistem perangkat lunak. Apabila terjadi perubahan perangkat lunak, maka pendekatan-pendekatan tersebut dapat di-*compile* ulang untuk membangun *link-link* yang baru. Pendekatan orthogonal, bayesian, dan linear mendukung sistem keamanan yang bersifat statis. Hal ini dikarenakan sistem memerlukan pihak eksternal untuk membangun *traceability link.* Sedangkan pendekatan pragmatis mendukung sistem keamanan yang bersifat dinamis karena berjalan tanpa membutuhkan pihak eksternal.

Tabel 2. Evaluasi terhadap Dimensi 'Objek Perubahan'

Pendekatan Traceability Link	Objek Perubahan									
	Artefak		Granularity			Impact		Propagasi Perubahan		
	High-level	Low-level	Coarse	Medium	Fine	Local	System-wide	Change Impact Analysis	Trace-ability Analysis	Effort Estimation
Orthogonal	√	√			√	√	√		√	
Bayesian		√			√	√			√	
Linear		√			√	√			√	
Pragmatis	√	√	√				√	√	√	

Tabel 3. Evaluasi terhadap Dimensi 'Properti Sistem'

Pendekatan Traceability Link	Objek Perubahan							
	Ketersediaan		Keaktifan		Keterbukaan		Keamanan	
	Parsial	Permanen	Reaktif	Proaktif	Terbuka	Tertutup	Statis	Dinamis
Orthogonal	√		√		√		√	
Bayesian	√		√		√		√	
Linear	√		√		√		√	
Pragmatis		√		√		√		√

Tabel 3 menampilkan hasil evaluasi terhadap aspek properti sistem berdasarkan penjelasan di atas.

4.4 Dukungan Perubahan (Change Support: How)

Semua pendekatan yang dibahas membangun *traceability links* (dan melakukan proses *recovery links* berdasarkan perubahan yang terjadi) secara otomatis. Semua pendekatan yang dibahas tidak menggunakan cara manual dalam memperoleh *traceability links* dan dalam melakukan perubahan dalam artefak perangkat lunak.

Pendekatan orthogonal, bayesian, dan linear diimplementasikan menggunakan sejumlah model matematis. Algoritma *tracing* menggunakan model tersebut terutama pada bagian inti. Sementara itu, beberapa bagian lain masih menggunakan basis model informal. Dengan demikian, semua pendekatan tersebut merupakan pendekatan yang semi-formal. Pengguna juga dapat mengaplikasikan model informal ketika harus memutuskan apakah hasilnya diterima atau ditolak. Hal ini dilakukan ketika tidak ada model matematis yang dapat digunakan untuk menentukan keputusan. Hal ini didasarkan pada intuisi dan pengetahuan pengguna. Sedangkan pendekatan pragmatis bersifat ad-hoc dikarenakan tidak menggunakan model matematis untuk membangun *traceability link.* Pendekatan pragmatis, bayesian, dan linear memiliki dukungan terhadap perubahan secara struktural. Sedangkan pendekatan orthogonal memiliki perubahan secara struktural dan semantik. Pendekatan tersebut digunakan untuk *tracing* kebutuhan, baik pada performa pertama (pembangunan) maupun setelah *traceability* telah dicapai (*recovery* akibat perubahan). Sedemikian hingga suatu perubahan dapat dilakukan pada salah satu artefak, baik berupa dokumen kebutuhan, dokumen desain, ataupun *source code*. Seluruh artefak tersebut disimpan sebagai berkas. Dengan demikian, perubahan dilakukan pada salah satu dari berkas-berkas tersebut. Perubahan tersebut dapat berupa salah satu dari sekian jenis perubahan, yaitu penambahan, penghapusan, ataupun modifikasi. Perubahan tersebut dapat berupa perubahan yang struktural, *semantic-preserving*, ataupun semantik. Tabel 4 di bawah ini menampilkan hasil evaluasi terhadap aspek dukungan perubahan berdasarkan penjelasan di atas. Sedangkan pendekatan pramatis hanya bersifat struktural dikarenakan tidak menggunakan informasi lain selain struktur antara dokumen kebutuhan, model, dan kodenya.

5. DISKUSI

Penelitian yang dilakukan oleh Siti dkk menampilkan tujuh pendekatan antara lain *Information retrieval, rule-based, event-based, hypertext-based, feature model-based, value-based, scenario-based*. Pendekatan yang berkembang setelah tahun 2007 lebih kepada pengembangan dari pendekatan-pendekatan tersebut. Dalam pencarian literatur yang telah ditemukan, kami menemukan banyak variasi-variasi pendekatan *information retrieval.* Misalnya penggunaan *smoothing filtering* pada IR [9], penggunaan *noun-based indexing* pada IR [10], *Incremental* LSI pada IR [11].

Pendekatan orthogonal, bayessian, linear, dan pragmatis untuk membangun *traceability link* pada dasarnya tidak terlalu berbeda dari pendekatan yang telah disampaikan pada penelitian sebelumnya dalam hal dukungan terhadap evolusi perangkat lunak. Terdapat beberapa kesamaan faktor dan terdapat pula beberapa perbedaan, namun perbedaan yang paling mencolok terdapat pada pendekatan pragmatis. Kesamaan faktor terjadi karena pendekatan tersebut bekerja dengan bantuan pihak eksternal, dengan kata lain pembangunan *traceability link* dilakukan pada waktu *compile.* Sedangkan pendekatan pragmatis merupakan model yang dibangun dalam sistem yang

Tabel 4. Evaluasi terhadap Dimensi 'Dukungan Perubahan'

Pendekatan Traceability Link	Dukungan Perubahan							
	Derajat Otomasi			Derajat Formalitas			Tipe Perubahan	
	Otomatis	Semi-otomatis	Manual	Ad-hoc	Semi-formal	Formal	Struktural	Semantik
Orthogonal	√				√		√	√
Bayesian	√				√		√	
Linear	√				√		√	
Pragmatis	√			√			√	

kemudian dapat membangun *traceability link* secara otomatis. Selain itu pendekatan orthogonal, bayessian, dan linear menggunakan metode matematika untuk membangun menggunakan informasi struktur sistem (dokumen kebutuhan, model, dan kode). Di samping itu, pendekatan pragmatis juga memiliki kesamaan dengan pendekatan bayessian dan linear dalam hal tipe perubahan karena tidak menggunakan informasi semantik untuk proses membangun *traceability link.*

Selama proses studi literatur terhadap penelitian yang dilakukan setelah tahun 2007, penelitian yang banyak berkembang adalah penelitian berkaitan pengembangan metode itu sendiri. Belum ada pendekatan yang menggabungkan antara satu pendekatan dengan pendekatan lainnya seperti yang disarankan oleh Siti yaitu *rule-based* dan *hyperlink-based*. Berdasarkan hasil studi literatur (orthogonal, linear, bayessian, dan pragmatis) ini kami masih belum dapat melihat peluang untuk menggabungkan antar pendekatan. Kami lebih menyarankan untuk mengembangkan pendekatan itu sendiri, misalnya dalam hal otomatisasi validasi link yang sudah dibangun. Diharapkan pengembangan tersebut dapat memberikan perbaikan link yang dibangun sebelumnya.

6. VALIDASI

Penilaian pendekatan yang disampaikan dalam makalah ini dilakukan berdasarkan konten dari jurnal yang telah dijadikan referensi tanpa melakukan percobaan ulang. Sehingga kami memiliki keterbatasan dalam mengintepretasikan referensi tersebut. Selain itu, metode penilaian dilakukan secara subjektif tanpa menggunakan metode-metode penilaian khusus.

7. KESIMPULAN

Studi literatur ini menggunakan sejumlah pendekatan pembangunan *traceability link* yang dilakukan setelah tahun 2007. Evaluasi pendekatan-pendekatan tersebut didasarkan pada taksonomi evolusi perangkat lunak. Dibandingkan dengan penelitian serupa yang dilakukan oleh Siti tidak terdapat perubahan yang cukup signifikan dengan pendekatan yang disampaikan pada makalah ini. Dalam makalah ini, terdapat dua kelompok pendekatan yang cukup berbeda, yaitu antara pendekatan orthogonal, bayessian, dan linear yang berjalan pada saat waktu *compile;* dengan pendekatan pragmatis yang berjalan pada waktu *load-time*. Faktor inilah yang menyebabkan kedua kelompok ini memiliki nilai-nilai yang berbeda pada taksonomi evolusi perangkat lunak.

8. DAFTAR PUSTAKA

[1] Mens, Tom dan Demeyer, Serge, (2008). "Software Evolution." Berlin : Springer.

[2] Dorfman, Merlin dan Flynn, Richard F., (1984). "Arts-An Automated Requirements Traceability System." Journal of Systems and Software, Vol. 4, hal. 63-74.

[3] Kannenberg, Andy dan Saiedian, Hossein, (2009). "Why Software Requirements Traceability Remains a Challenge." The Journal of Defense Software Engineering, Juli, Vol. 22, hal. 14-19.

[4] Buckley, Jim, et al., (2003). "Towards a Taxonomy of Software Change." Journal of Software Maintenance and Evolution: Research and Practice, Vol. 17, hal. 309-332.

[5] Rochimah, Siti, Wan Kadir, W. M. N. dan Abdullah, Abdul H, (2007). "An Evaluation of Traceability Approaches to Support Software Evolution." International Conference on Software Engineering Advances, 2007.

[6] Gethers, Malcom, et al., (2011). "On Integrating Orthogonal Information Retrieval Methods to Improve Traceability Recovery." International Conference on Software Maintenance, 2011.

[7] Omoronyia, Inah, Sindre, Guttorm dan Stålhane, Tor., (2011). "Exploring a Bayesian and linear approach to requirements traceability." Information and Software Technology, Vol. 53, hal. 851-871.

[8] Aleksy, Marcus, et al. Manheim, (2009). "A Pragmatic Approach to Traceability in Model-Driven Development." Process Innovation for Enterprise Software, Germany.

[9] De Lucia, Andrea, et al. Fisciano, (2011). "Improving IR-based Traceability Recovery Using Smoothing Filters." Program

Comprehension (ICPC), hal. 21-30.
[10] Capobianco, Giovanni, et al. Pesche, (2010). "Improving IR-based Traceability Recovery via Noun-based Indexing of Software Artifacts." Journal of Software Maintenance and Evolution: Research and Practice, hal. 2-35.
[11] Jiang, Hsin-yi, et al., (2008). "Incremental Latent Semantic Indexing for Automatic Traceability Link Evolution Management." IEEE, hal. 59-68.