

Klasifikasi Kebutuhan Pada Diagram SysML

Enny Dwi Oktaviyani

Jurusan Teknik Informatika, Fakultas Teknik, Universitas Palangkaraya

Email : enny_obrien@yahoo.com

Abstract

Requirements engineering activities are more widespread and difficult lead to the increasing complexity of software systems. User needs often change sometimes make changes that affect other needs and at a later stage. One user requirement modeling that can be used is the SysML diagrams. SysML diagrams can be used for the activity of complex software systems. In SysML diagrams are requirements diagrams to classify requirements into functional requirements, non-functional requirements and external interfaces requirements. SysML diagrams help to determine the order of the most important requirements to be done at the time of implementation. SysML diagrams help breaking into the complex requirements of simple requirements and establish a hierarchy that links one needs with other requirements.

Key Words : sysML diagrams, functional requirements, non-functional requirements, external interfaces requirements, requirements classification.

1. Pendahuluan

Aktifitas rekayasa kebutuhan yang semakin luas dan sulit menyebabkan meningkatnya kompleksitas sistem perangkat lunak. Ada beberapa masalah yang sering dijumpai dalam membuat *requirement*, yaitu : membuat asumsi yang buruk, menulis implementasi (HOW) daripada requirement (WHAT), menjelaskan operasional daripada kebutuhan, menggunakan istilah yang salah, menggunakan bahasa yang kurang tepat, kebutuhan yang tidak lengkap, dan menspesifikasikan kebutuhan secara berlebihan. User requirement menggambarkan kebutuhan fungsional dan kebutuhan non fungsional yang dapat dipahami oleh user yang tidak memiliki latar belakang teknis yang cukup.

Untuk membuat suatu sistem perangkat lunak yang kompleks, maka diperlukan suatu model pengembangan user requirement, terutama kegiatan mendokumentasikan dan menganalisis user requirement. Perkembangan di bidang teknik perangkat lunak intensif

sistem memiliki keuntungan yang besar dalam produktivitas dan kemakmuran bahwa masyarakat telah melihat dalam beberapa tahun terakhir (Dedrick et al., 2003). Kompleksitas ini terus meningkat karena banyaknya faktor elemen dan kehandalan, dengan demikian sistem perangkat lunak ini harus dipecah menjadi beberapa komponen yang kecil dalam rangka untuk mengelola kompleksitas dan pelaksanaan verifikasi. Sistem requirement berasal dari user requirement dan biasanya dimodelkan dengan metode formal atau semi-formal dan bahasa memperkenalkan event sink, sourcer, dan trigger untuk WSDL.

Ada beberapa pendekatan untuk pemodelan user requirement. Pada dasarnya, pendekatan ini dapat diklasifikasikan sebagai berbasis grafis, murni teks, atau kombinasi keduanya. Pendekatan yang digunakan dalam penelitian ini adalah murni teks klasifikasi. Pendekatan yang paling umum untuk menulis user requirement adalah menggunakan bahasa alami. Keuntungannya bahasa alami adalah

komunikasi yang lebih mudah antara para stakeholder. Namun, terdapat beberapa masalah umum ketika bahasa alami digunakan seperti ketidaktepatan, kesalahpahaman, ambiguitas dan inkonsistensi (Kamsties, 2005). Bahasa alami kadang tidak cukup untuk menjelaskan, atau membuat dokumen menjadi sulit dibaca, jenis-jenis kebutuhan terkadang menjadi sulit dibedakan, seringkali kebutuhan digabungkan menjadi satu kumpulan kebutuhan saja.

Seperti halnya dengan penggunaan model dalam ilmu pengetahuan, model user kebutuhan dapat dicirikan oleh semantik, ontologi, dan epistemologi (Frigg dan Hartmann, 2006). Pemodelan user requirement yang digunakan dalam penelitian ini adalah diagram SysML untuk membantu mengorganisir kebutuhan menjadi lebih baik dan terstruktur melalui semantik (Michel et al, 2010). Diagram SysML menunjukkan secara eksplisit beberapa hubungan antara kebutuhan yang berbeda. Keuntungan menggunakan diagram SysML ini adalah menetapkan standarisasi kebutuhan yang didefinisikan secara semantic.

Keuntungan lain dari SysML adalah memungkinkan kebutuhan yang kompleks dipecah-pecah menjadi lebih sederhana, dan membentuk suatu hirarki kebutuhan yang saling berhubungan. Diagram SysML bersifat hirarki sehingga jika terjadi perubahan-perubahan kebutuhan oleh user dalam jumlah yang besar, maka tidak akan menimbulkan perubahan yang signifikan pada kebutuhan lainnya. Keuntungannya adalah bahwa kompleksitas sistem dimulai dari awal pengembangan, dengan menguraikan requirement yang kompleks. Dalam hal ini, kebutuhan umum dapat dibagi oleh kebutuhan lainnya.

Beberapa penelitian sebelumnya mengenai klasifikasi kebutuhan seperti yang dilakukan Jane Cleland-Huang, Raffaella Settini, Xuchang Zou, Peter Solc (2006); dan Ceyda Güngör, Hayri Baraç (2009), penelitian ini mengklasifikasikan kebutuhan berdasarkan kebutuhan non fungsional. Permasalahan yang sering terjadi pada Diagram SysML ini khususnya pada tahap pembuatan diagram kebutuhan adalah kesulitan untuk menentukan jenis-jenis kebutuhan. Michel dos Santos Soares, Jos Vrancken, Alexander Verbraeck (2010) telah melakukan penelitian pemodelan user requirement dengan menggunakan diagram SysML untuk menentukan jenis dan mengklasifikasikan kebutuhan berdasarkan kebutuhan fungsional, kebutuhan non fungsional dan kebutuhan eksternal interface. Klasifikasi kebutuhan termasuk dalam teks klasifikasi.

Sejumlah pendekatan telah digunakan untuk klasifikasi teks misalnya metode klasifikasi kNN (k-Nearest Neighbor) seperti yang dilakukan oleh Y. Yang dan X. Liu. KNN memiliki beberapa kelebihan yaitu ketangguhan terhadap *training data* yang memiliki banyak *noise* dan efektif apabila *training data*-nya besar. Sedangkan, kelemahan KNN adalah KNN perlu menentukan nilai dari parameter k (jumlah dari tetangga terdekat), *training* berdasarkan jarak dari tiap *query instance* pada keseluruhan *training sample* tidak jelas mengenai jenis jarak apa yang harus digunakan dan atribut mana yang harus digunakan untuk mendapatkan hasil terbaik.

Klasifikasi teks juga telah dilakukan dengan menggunakan naive bayesian oleh Jantima Polpinij dan Aditya Ghose (2008), tetapi tingkat akurasi dengan menggunakan SVM lebih tinggi daripada menggunakan naive bayesian. Metode klasifikasi yang

digunakan pada penelitian ini adalah menggunakan Support Vector Machine (SVM). Namun metode SVM sendiri sedang dikembangkan untuk klasifikasi multiclass.

Lutz (1993) menunjukkan bahwa 60% dari kesalahan dalam sistem adalah hasil dari kesalahan kebutuhan. Studi yang dilakukan oleh Standish Group (TSG, 2003) dan peneliti lainnya (Genuchten, 1991; Hofmann et al, 2001) menemukan bahwa faktor utama untuk masalah dengan proyek perangkat lunak (over biaya, keterlambatan, ketidakpuasan pengguna) terkait dengan masalah kebutuhan, seperti kurangnya input pengguna, spesifikasi kebutuhan tidak lengkap, perubahan requirement yang tidak terkendali, dan tujuan tidak jelas. Dari sudut pandang desain perangkat lunak, mengklasifikasikan kebutuhan pada fase awal pengembangan perangkat lunak membantu dalam mengidentifikasi subsistem, komponen, dan hubungannya. Mengklasifikasikan kebutuhan sangat membantu pada saat merancang arsitektur perangkat lunak

2. Pembahasan

2.1. Kebutuhan Perangkat Lunak

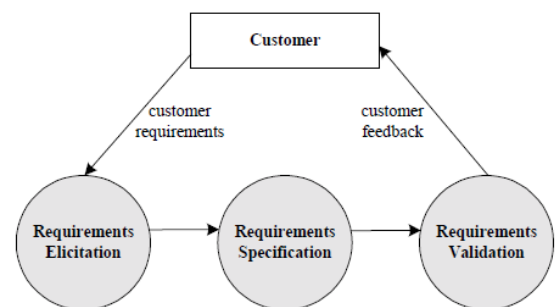
IEEE Standard Glossary of Software Engineering Terminology (1990) mendefinisikan kebutuhan sebagai berikut :

1. Sebuah kondisi atau kemampuan yang dibutuhkan oleh *user* untuk memecahkan masalah atau tujuan arsip.
2. Sebuah kondisi atau kemampuan yang harus dipenuhi atau proses oleh komponen sistem atau sistem untuk memenuhi kontrak, standar, spesifikasi, atau dokumen resmi lainnya.
3. Sebuah representasi terdokumentasi dari kondisi atau kemampuan seperti pada poin 1 atau 2.

Berikut ini adalah keseragaman dari definisi kebutuhan menurut Sommerville dan Sawyer (1997) :

“Kebutuhan adalah spesifikasi dari apa yang harus diimplementasikan. Kebutuhan mendeskripsikan bagaimana system harus dilakukan atau sistem kepemilikan atau atribut. Kebutuhan mungkin mempunyai kendala pada proses pengembangan system”

Proses rekayasa kebutuhan dapat digambarkan pada gambar 1.



Gambar 1. Proses rekayasa kebutuhan

2.2. Requirements Elicitation

Requirement Elicitation atau pengumpulan kebutuhan adalah proses mengumpulkan dan memahami kebutuhan dari *user*. Kadang masalah yang muncul berakar dari perbedaan disiplin ilmu yang dimiliki. *Customer* adalah orang yang *expert* pada domain yang softwarnya ingin dikembangkan (domain specialist), dilain pihak pengembang atau *requirements analyst* sama sekali buta terhadap *knowledge domain* tersebut, meskipun tentu memahami dengan benar bagaimana sebuah software harus dikembangkan. Perbedaan disiplin ilmu tersebut yang diharapkan bisa diatasi dengan adanya interaksi terus menerus dan berulang (iterasi) antara pengembang dan customer. Proses interaksi tersebut kemudian dimodelkan menjadi beberapa teknik dan metodologi diantaranya adalah *interviewing, brainstorming, prototyping, use case, dsb*

2.3. Requirements Specification

Setelah masalah berhasil dipahami, pengembang mendeskripsikannya dalam bentuk dokumen spesifikasi. Spesifikasi ini berisi tentang fitur dan fungsi yang diinginkan oleh customer, dan sama sekali tidak membahas bagaimana metode pengembangannya. IEEE mengeluarkan standard untuk dokumen spesifikasi requirements yang terkenal dengan nama IEEE Recommended Practice for Software Requirements Specifications [IEEE-830]. Dokumen spesifikasi requirements bisa berisi *functional requirements, performance requirements, external interface requirements, design constraints, maupun quality requirements*

2.4. Requirements Validation and Verification

Setelah requirements specification berhasil dibuat, maka dilakukan proses :

- Validation (validasi), yaitu proses untuk memastikan bahwa kebutuhan yang benar sudah ditulis
- Verification (verifikasi), yaitu proses untuk memastikan bahwa kebutuhan sudah ditulis dengan benar

Proses validasi dan verifikasi ini melibatkan customer (user) sebagai pihak yang menilai dan memberi feedback berhubungan dengan requirements.

2.5. Jenis-Jenis Kebutuhan

2.5.1. Kebutuhan Pengguna

Kebutuhan pengguna adalah pernyataan tentang layanan yang disediakan sistem dan tentang batasan batasan operasionalnya. Pernyataan ini dapat dilengkapi dengan gambar/diagram yang dapat dimengerti dengan mudah. Kebutuhan pengguna menggambarkan kebutuhan fungsional dan kebutuhan non fungsional yang dapat dipahami oleh

pengguna (user) yang tidak memiliki latar belakang teknis yang cukup.

Kebutuhan pengguna menjelaskan perilaku luar dari sistem, tidak secara teknis, karena itu perlu menggunakan bahasa alami, atau bahasa yang sederhana. Masalah dalam menyiapkan kebutuhan pengguna adalah: Bahasa alami kadang tidak cukup untuk menjelaskan, atau membuat dokumen jadi sulit dibaca, Jenis-jenis kebutuhan kadang jadi sulit dibedakan, Sering digabungkan menjadi satu kumpulan kebutuhan saja.

2.5.2. Kebutuhan Sistem

Sekumpulan layanan/kemampuan sistem dan batasan-batasannya yang ditulis secara detail. Dokumen kebutuhan sistem sering disebut functional specification (spesifikasi fungsional), harus menjelaskan dengan tepat dan detail. Ini bisa berlaku sebagai kontrak antara klien dan pembangun. Kebutuhan sistem merupakan deskripsi sistem yang lebih detail dari kebutuhan pengguna (jadi masih berisi kebutuhan fungsional dan kebutuhan non fungsional). kebutuhan ini berlaku sebagai kontrak pembangunan sistem dan terdiri dari model sistem seperti model object atau model data-flow.

Kebutuhan sistem menyatakan apa yang harus dikerjakan sistem, dan bukan bagaimana sistem diimplementasikan. Untuk itu digunakan bahasa yang lebih spesifik dan bersifat teknis, misalnya PDL (Program Description Language). PDL digunakan untuk menggambarkan kebutuhan secara operasional dan berguna untuk implementasi program.

2.5.3 Spesifikasi Rancangan Perangkat Lunak

Spesifikasi rancangan perangkat lunak adalah gambaran abstrak dari rancangan perangkat lunak yang menjadi dasar bagi perancangan dan implementasi

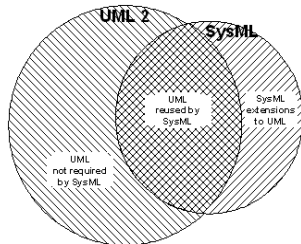
secara detil. Ketiga jenis kebutuhan tersebut diperlukan dalam pembangunan perangkat lunak karena memberi pengertian ke pihak yang berbeda kepentingan.

2.6. Diagram SysML

Diagram SysML membantu dalam mengorganisir kebutuhan menjadi lebih baik, dan menunjukkan hubungan antara kebutuhan yang berbeda secara eksplisit. Keuntungan menggunakan diagram SysML adalah sebagai standarisasi cara menetapkan kebutuhan yang didefinisikan secara semantik. Diagram SysML dapat dikombinasikan dengan UML untuk perancangan perangkat lunak, kebutuhan yang disediakan oleh SysML, biasanya ditulis dalam bahasa alami, dan diagram usecase, digunakan sebagai spesifikasi awal kebutuhan sistem (Soares dan Vrancken, 2008a).

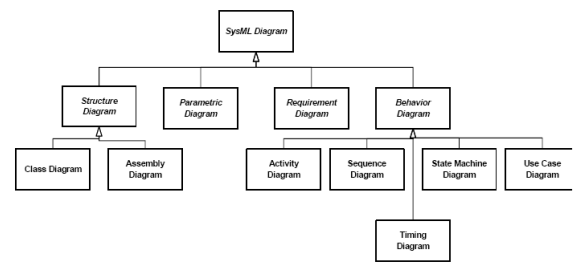
Gambar 2 menggambarkan hubungan antara SysML dan UML. Kebutuhan yang dikerjakan semua dalam satu sistem menjadi sulit karena kendala biaya yang tinggi, tekanan pasar, tekanan dari klien, staf dan waktu yang kurang memadai. Hal ini merupakan kesulitan untuk membuat prioritas kegiatan mana yang paling penting selama Proses Rekayasa Kebutuhan.

SysML menggunakan dan merupakan extends dari paket UML, ekstensi Mekanisme seperti stereotip, meta class dan model library. Menggunakan kombinasi dari model profiling dan metateknik yang menggunakan bahasa yang tepat untuk menentukan kendala dan semantik



Gambar 2. Hubungan sysML dan UML

Diagram SysML memungkinkan kebutuhan yang kompleks dipecah menjadi kebutuhan yang lebih sederhana, sebagai hirarki kebutuhan yang berkaitan satu dan yang lainnya. Keuntungannya adalah bahwa system yang kompleks dapat dikerjakan diawal pengembangan, dengan menguraikan kebutuhan yang kompleks. Konsep hirarki juga memungkinkan penggunaan kembali kebutuhan. Gambar 3. menggambarkan hirarki dari diagram SysML



Gambar 3. Hirarki diagram sysML

2.7. Klasifikasi Kebutuhan

Pada saat mendesain arsitektur, bagian dari kebutuhan fungsional harus diketahui. Selain itu, arsitektur dari kebutuhan fungsional harus sesuai dan harus dibuat secara eksplisit. Kebutuhan diklasifikasikan menjadi kebutuhan fungsional dan kebutuhan non fungsional (Robertson, 2006). Kebutuhan non-fungsional juga disebut kebutuhan kualitas, dan merupakan dasar untuk menentukan keberhasilan system. Kebutuhan pada diagram SysML diklasifikasikan sebagai berikut :

2.8. Kebutuhan Fungsional :

Menjelaskan tentang layanan yang perlu disediakan oleh sistem, bagaimana sistem menerima dan mengolah masukan, dan bagaimana system mengatasi situasi-situasi tertentu. Selain itu, menentukan apa yang tidak dikerjakan oleh system.

Kebutuhan fungsional menggambarkan kebutuhan sistem secara detail seperti input, output dan pengecualian yang berlaku. Masalah yang mungkin terjadi dalam menyusun kebutuhan fungsional adalah:

- Dapat diinterpretasikan atau diartikan berbeda oleh user atau developer
- Hasil interpretasi sering tidak menjawab kebutuhan klien.
- Pada sistem yang besar, kelengkapan kebutuhan dan konsisten sulit dicapai karena kerumitan sistem
- Perlu analisis yang dalam dan menyeluruh untuk mengurangi kesalahan

Menurut IEEE Std 830-1998, Kebutuhan fungsional mencakup :

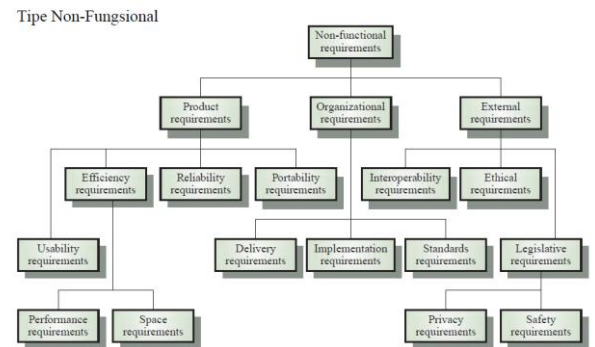
- Validitas pemeriksaan di input
- Urutan operasi
- Tanggapan situasi normal, termasuk
 - Overflow
 - Fasilitas Komunikasi
 - Error handling dan recovery
- Pengaruh parameter
- Hubungan output ke input, termasuk
 - Urutan Input / Output
 - Formula untuk konversi input ke output .

2.9. Kebutuhan Non Fungsional :

Berisi batasan-batasan pada layanan atau fungsi yang disediakan oleh sistem. Termasuk batasan waktu, batasan proses pembangunan, standar-standar tertentu. Karena berkaitan dengan kebutuhan sistem secara keseluruhan, jika kebutuhan non fungsional gagal terpenuhi, maka berakibat pada sistem secara keseluruhan.

Gambar 4. menunjukkan tipe kebutuhan non fungsional. Contoh kebutuhan non fungsional adalah kecepatan akses, keamanan data,

besarnya kapasitas penyimpanan yang diperlukan, privasi masing-masing profil /account, bahasa pemrograman yang digunakan, sistem operasi yang digunakan.



Gambar 4. Tipe kebutuhan non fungsional

Kebutuhan non fungsional mencakup :

- Usability
- Reliability
- Interoperability
- Scalability
- Security
- Maintainability
- Efficiency
- Portability

2.10. Kebutuhan External Interface :

Kebutuhan External Interface harus penjelasan rinci semua input dan output dari sistem perangkat lunak. Kebutuhan external Interface mencakup baik konten dan format sebagai berikut:

- Nama item
- Tujuan Deskripsi;
- Sumber masukan atau tujuan output;
- jangkauan valid, akurasi, dan / atau toleransi;
- Unit ukuran;
- Waktu;
- Hubungan untuk input lain / output;
- format Layar / organisasi;
- format Window / organisasi;

- j) Data format;
- k) format Perintah;
- l) pesan Akhir.

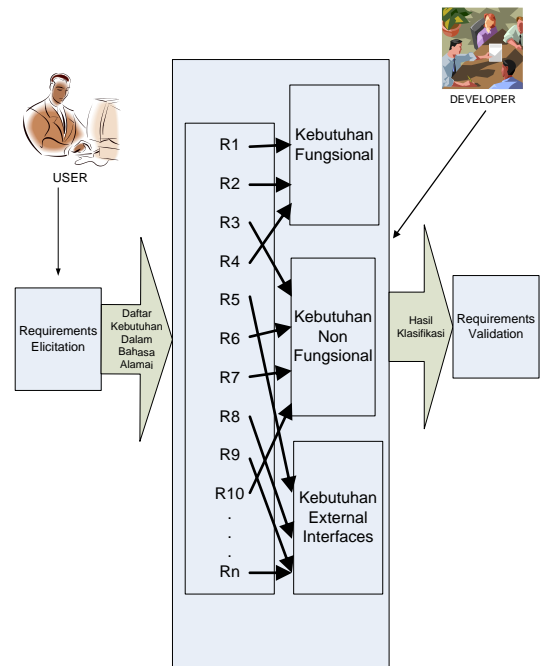
Mengklasifikasikan kebutuhan sangat penting untuk menguraikan sistem menjadi subsistem dan membantu dalam identifikasi arsitektur sistem.

Memprioritaskan kebutuhan ini memberikan indikasi bahwa kebutuhan harus ditangani. Klasifikasi kebutuhan untuk menentukan :

- Apakah suatu kebutuhan didapat dari satu atau lebih kebutuhan yang berlevel lebih tinggi atau merupakan emergent property atau ditetapkan oleh pihak yang berkepentingan atau sumber lain
- Apakah kebutuhan ada pada produk atau proses.
- Prioritas. Secara umum, semakin tinggi prioritas suatu kebutuhan semakin mendesak pula untuk dipenuhi dalam produk akhir.
- Cakupan dari kebutuhan. Hal ini sangat berpengaruh pada arsitektur software dan desain komponen stabilitas.

Kebutuhan kadang berubah dalam suatu siklus hidup software bahkan mungkin dalam proses pengembangannya. Gambar 5. menggambarkan klasifikasi kebutuhan pada diagram kebutuhan SysML, pada tahap elicitasi, daftar kebutuhan yang masih dalam bahasa alami diklasifikasikan menjadi kebutuhan fungsional, kebutuhan non fungsional dan kebutuhan eksterna interface.

Kemudian hasil klasifikasi akan di validasi pada tahap validasi.



Gambar 5. Klasifikasi kebutuhan diagram sysML

3. Kesimpulan

1. Diagram SysML membantu dalam mengorganisir kebutuhan menjadi lebih baik, dan juga menunjukkan hubungan antara kebutuhan yang berbeda secara eksplisit.
2. Klasifikasi Kebutuhan sangat berguna untuk tahap analisis kebutuhan.

4. Saran

Saran Untuk penelitian lebih lanjut adalah mengklasifikasikan kebutuhan yang hampir mirip menjadi lebih dari satu kategori atau jenis kebutuhan.

Daftar Pustaka

B.J.Mattew, S.Jawed, *The Classification of Requirements Engineering Methods*

- D.S.S.Michel, V.Jos, V.Alexander (2010), *User requirements modeling and analysis of software-intensive systems* : The Journal of Systems and Software 84 (2011) 328–339
- E.W. Karl (2003), *Software Requirements*, 2th Edition
- G.S. Ceyda, B.Hayri (2010), *Fuzzy quality function deployment based methodology for acquiring enterprise software selection requirements* : Expert System with Application 37(2010)3415-3426
- H.C.Jane, S.Raffaella, Z.Xuchang, S.Peter, *The Detection and Classification of Non-Functional Requirements with Application to Early Aspects*. Center for Requirements Engineering School of Computer Science, Telecommunications, and Information Systems DePaul University
- Sommerville, Ian. "Software Engineering" .6th . Addison Wesley. 2001
<http://www.sysml.org/specification>. Juni 2010
- Polpinij Jantina, Ghose Aditya (2008), *An Automatic Elaborate Requirement Specification By Using Hierarchical Text Classification* : 2008 International Conference on Computer Science and Software Engineering
- S.J.Jung, C.C. Yu (2009), *Pseudo software: A mediating instrument for modeling software requirements* : The Journal of Systems and Software 83 (2010) 599–608
- Xue Chie, Q.Qing-Ying, F.Pei-En, Y.Zhen-Nong (2010), *An Automatic Classification Method for Patents* : 2010 Seventh International Conference on Fuzzy Systems and Knowledge Discovery (FSKD 2010)