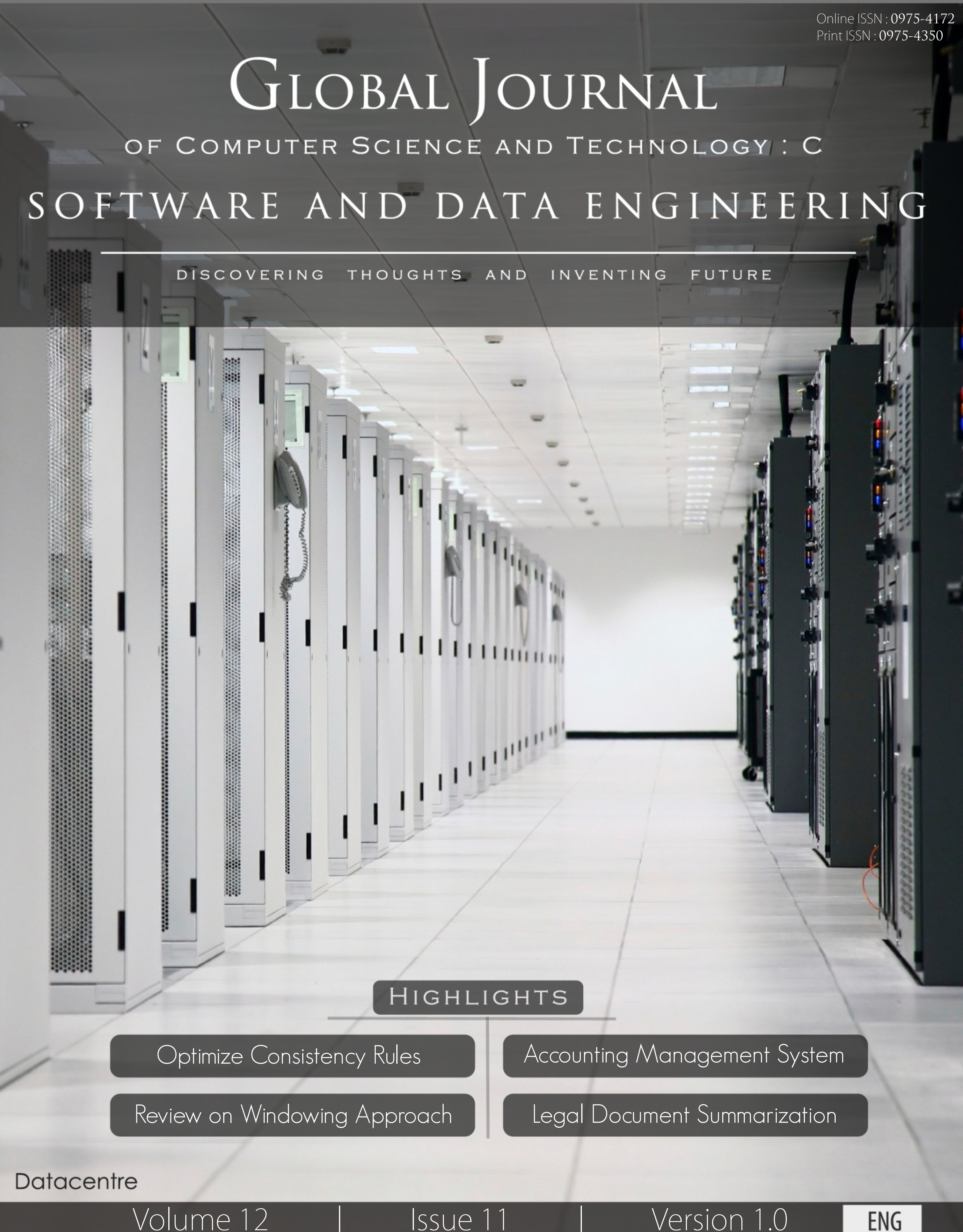
Online ISSN : 0975-4172
Print ISSN : 0975-4350

Global Journal

of Computer Science and Technology : C

Software and Data Engineering

DISCOVERING THOUGHTS AND INVENTING FUTURE



Highlights

- Optimize Consistency Rules
- Accounting Management System
- Review on Windowing Approach
- Legal Document Summarization

Datacentre

Volume 12 | Issue 11 | Version 1.0 | ENG

Global Journal of Computer Science and Technology: C
Software & Data Engineering

Global Journal of Computer Science and Technology: C
Software & Data Engineering

Volume 12 Issue 11 (Ver. 1.0)

Open Association of Research Society

Incorporation No.: 0423089
License No.: 42125/022010/1186
Registration No.: 430374
Import-Export Code: 1109007027
Employer Identification Number (EIN):
USA Tax ID: 98-0673427

Global Journals Inc.

*(A Delaware USA Incorporation with "Good Standing"; **Reg. Number: 0423089**)*

Sponsors: Global Association of Research

Open Scientific Standards

Publisher's Headquarters office

Global Journals Inc., Headquarters Corporate Office, Cambridge Office Center, II Canal Park, Floor No. 5th, ***Cambridge (Massachusetts)***, Pin: MA 02141
United States
USA Toll Free: +001-888-839-7392
USA Toll Free Fax: +001-888-839-7392

Offset Typesetting

Global Association of Research, Marsh Road, Rainham, Essex, London RM13 8EU
United Kingdom.

Packaging & Continental Dispatching

Global Journals, India

Find a correspondence nodal officer near you

To find nodal officer of your country, please email us at *local@globaljournals.org*

eContacts

Press Inquiries: *press@globaljournals.org*
Investor Inquiries: *investers@globaljournals.org*
Technical Support: *technology@globaljournals.org*
Media & Releases: *media@globaljournals.org*

Pricing (Including by Air Parcel Charges):

For Authors:
22 USD (B/W) & 50 USD (Color)
Yearly Subscription (Personal & Institutional):
200 USD (B/W) & 250 USD (Color)

Editorial Board Members (HON.)

Contents of the Volume

Global Journal of Computer Science and Technology
Software & Data Engineering
Volume 12 Issue 11 Version 1.0 Year 2012
Type: Double Blind Peer Reviewed International Research Journal
Publisher: Global Journals Inc. (USA)
Online ISSN: 0975-4172 & Print ISSN: 0975-4350

UML Modeling for Tea / Coffee Machine

By Mr. S. Venu Gopal, Mr. H. Venkateswara Reddy & Mr. G. Sreenivasulu
Vardhaman College of Engineering

Abstract - Unified Modeling language (UML) is one of the important modeling languages used for the visual representation of the research problem. In the present paper, UML model is designed for the Tea / Coffee Machine which is used for the purpose of the public in the hotels or restaurants'. The class and use case diagrams are designed & performance is evaluated as a sample program through a case study.

Coffeemakers or coffee machines are cooking appliances used to brew coffee without having to boil water in a separate container. While there are many different types of coffeemakers using a number of different brewing principles, in the most common devices, coffee grounds are placed in a paper or metal filter inside a funnel, which is set over a glass or ceramic coffee pot, a cooking pot in the kettle family. Cold water is poured into a separate chamber, which is then heated up to the boiling point, and directed into the funnel.

Keywords : UML, Modeling, things, class diagram, use case diagram, annotational things, package, note notation, Relationships and stereo types.

GJCST-C Classification : D.2.2

UML Modeling for Tea / Coffee Machine

Mr. S. Venu Gopal [α], Mr. H. Venkateswara Reddy [σ] & Mr. G. Sreenivasulu [ρ]

Abstract - Unified Modeling language (UML) is one of the important modeling languages used for the visual representation of the research problem. In the present paper, UML model is designed for the Tea / Coffee Machine which is used for the purpose of the public in the hotels or restaurants'. The class and use case diagrams are designed & performance is evaluated as a sample program through a case study.

Coffeemakers or *coffee machines* are cooking appliances used to brew coffee without having to boil water in a separate container. While there are many different types of coffeemakers using a number of different brewing principles, in the most common devices, coffee grounds are placed in a paper or metal filter inside a funnel, which is set over a glass or ceramic *coffee pot*, a cooking pot in the kettle family. Cold water is poured into a separate chamber, which is then heated up to the boiling point, and directed into the funnel.

IndexTerms : UML, Modeling, things, class diagram, use case diagram, annotational things, package, note notation, Relationships and stereo types.

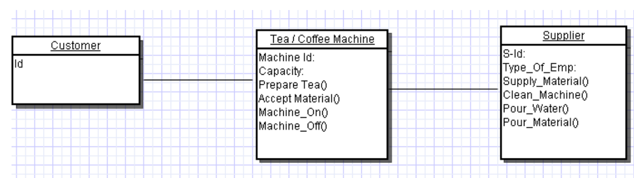
I. Introduction to UML and Topic

This section provides a general overview of UML concentrating on the syntax that is relevant to this paper. Figure 1 displays the different types of UML syntax used in this paper. In addition, we introduce new UML syntax (Vocabulary) in the form of stereotypes. For further information on UML the reader is referred to[1] . UML has three main building blocks: Things, Relationships, and Diagrams. "Things" are the main components of the model. "Things" are connected by Relationships. Diagrams display the Things and Relationships in different active or passive contexts. For example, a diagram can document a dynamic process in which a student may register for a class or it can document a static data structure of an organization. There are four kinds of things: Structural, Behavior, Grouping, and Annotational. One of the seven structural ‚things' of interest is a class. A class can contain a name, attributes, and operations. Classes will be used with objects. Behavior "things" are the verbs of UML. They are the dynamic parts of the UML. Behavior "things" will not be discussed in this paper. A grouping "thing" as the name states, permits the combining of different parts under a similar category. We will use the grouping "thing" named 'package". The final "thing" is annotational (it can also be called as a note). Notes comment a model. Notes can be used to comment the enterprise constraints of a key chain.

Author α : M. Tech (CSE), MISTE. Sr. Assistant Professor, Department of Computer Science and Engineering.
E-mail : venusekkeri@gmail.com
Author σ : M.Tech(CSE), (Ph.D), Associate Professor, Department of Computer Science and Engineering.
E-mail : venkat_nidhish@yahoo.co.in
Author ρ : ME, MISTE, Associate Professor, Department of Computer Science and Engineering, ACE Engineering College , Hyderabad.
E-mail : gvsreenu@gmail.com

II. Modeling of Tea/ Coffee Machine Main Idea

Object Diagram for Tea / Coffee Machine working model.



III. Basic Notations to Model Tea/ Coffee Machine

The following are the basic notations to model the Tea / Coffee machine working model.

Basic Notations to Model tea / Coffee Machine

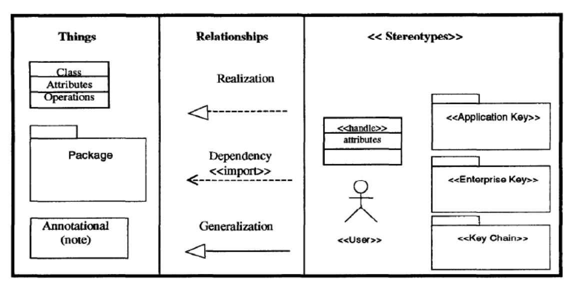


Figure 2 :

Class Diagram for Tea / Coffee Machine working model.

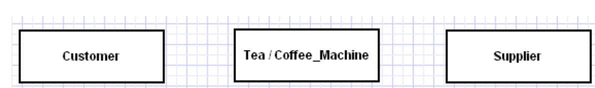


Figure 3 :

Use case or Behavioral Model for the Tea / Coffee Machine.

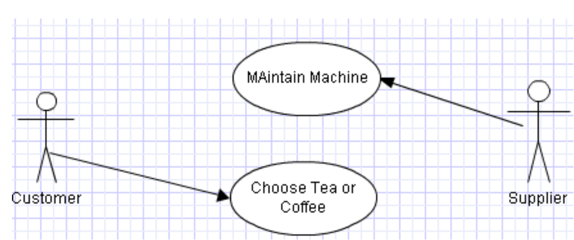


Figure 4 :

Global Journal of Computer Science and Technology (C) Volume XII Issue XI Version I Year 2012

Global Journal of Computer Science and Technology (C) Volume XII Issue XI Version I Year 2012

IV. Conclusion

From the above, it is concluded that the UML Class model is a powerful model used to depict the software development problems and the hardware problems. In the Tea / Coffee Machine Designing, it's a time consuming with compare to normal process. The present work is further extended by considering the different kinds of activities performed by customer and supplier. The present work is considered only for the basic model of tea / Coffee machine at therefore, the UML modeling for Tea / Coffee machine can be further extended for the automatic machine.

References Références Referencias

1. User Interface Modelling with UML Paulo Pinheiro da Silva and Norman W. PatonOMG ,
2. "Unified Modeling Language Specification", Available Online Via www.omg.org , 2001
3. B. Selic, and J. Rumbaugh, "UML for Modeling Complex Real Time Systems", Available Online Via ww.rational.com/Products/ Whitepapers/100230.Jsp.
4. Mr. S. Venu Gopal is a Sr. Assistant Professor, Department of Computer Science and Engineering,Vardhaman College of Engineering, Shamshabad – 501 218, Hyderabad. He got his M. Tech (CSE) from JNTU Hyderabad. He has more than 7 years of teaching experience.
5. Mr. H. Venkateswara Reddy is a Associate Professor, Department of Computer Science and Engineering, Vardhaman College of Engineering, Shamshabad – 501 218, Hyderabad. He is perusing Ph.D from JNTU Hyderabad. He has more than 15 Years of teaching experience.
6. , is a Associate Professor, Department of Computer Science and Engineering, Engineering College Hyderabad, He is perusing Ph.D from JNTU Hyderabad. He has more than 10 Years of teaching experience.

Global Journal of Computer Science and Technology
Software & Data Engineering
Volume 12 Issue 11 Version 1.0 Year 2012
Type: Double Blind Peer Reviewed International Research Journal
Publisher: Global Journals Inc. (USA)
Online ISSN: 0975-4172 & Print ISSN: 0975-4350

Developing an Embedded Model for Test Suite Prioritization Process to Optimize Consistency Rules for Inconsistencies Detection and Model Changes

By Muzammil H Mohammed & Sultan Aljahdali
Taif University, Taif, Saudi Arabia

Abstract - Software form typically contains a lot of contradiction and uniformity checkers help engineers find them. Even if engineers are willing to tolerate inconsistencies, they are better off knowing about their existence to avoid follow-on errors and unnecessary rework. However, current approaches do not detect or track inconsistencies fast enough. This paper presents an automated approach for detecting and tracking inconsistencies in real time (while the model changes). Engineers only need to define consistency rules-in any language-and our approach automatically identifies how model changes affect these consistency rules. It does this by observing the behavior of consistency rules to understand how they affect the model. The approach is quick, correct, scalable, fully automated, and easy to use as it does not require any special skills from the engineers using it. We use this model to define generic prioritization criteria that are applicable to GUI, Web applications and Embedded Model. We evolve the model and use it to develop a unified theory. Within the context of this model, we develop and empirically evaluate several prioritization criteria and apply them to four stand-alone GUI and three Web-based applications, their existing test suites and mainly embedded systems. In this model we only run our data collection and test suite prioritization process on seven programs and their existing test suites. An experiment that would be more readily generalized would include multiple programs of different sizes and from different domains. We may conduct additional empirical studies with larger EDS to address this threat each test case has a uniform cost of running (processor time) monitoring (human time); these assumptions may not hold in practice. Second, we assume that each fault contributes uniformly to the overall cost, which again may not hold in practice.

GJCST-C Classification : *D.2.5*

DEVELOPING AN EMBEDDED MODEL FOR TEST SUITE PRIORITIZATION PROCESS TO OPTIMIZE CONSISTENCY RULES FOR INCONSISTENCIES DETECTION AND MODEL CHANGES

Strictly as per the compliance and regulations of:

Developing an Embedded Model for Test Suite Prioritization Process to Optimize Consistency Rules for Inconsistencies Detection and Model Changes

Muzammil H Mohammed [α] & Sultan Aljahdali [σ]

Abstract - Software form typically contains a lot of contradiction and uniformity checkers help engineers find them. Even if engineers are willing to tolerate inconsistencies, they are better off knowing about their existence to avoid follow-on errors and unnecessary rework. However, current approaches do not detect or track inconsistencies fast enough. This paper presents an automated approach for detecting and tracking inconsistencies in real time (while the model changes). Engineers only need to define consistency rules - in any language - and our approach automatically identifies how model changes affect these consistency rules. It does this by observing the behavior of consistency rules to understand how they affect the model. The approach is quick, correct, scalable, fully automated, and easy to use as it does not require any special skills from the engineers using it. We use this model to define generic prioritization criteria that are applicable to GUI, Web applications and Embedded Model. We evolve the model and use it to develop a unified theory. Within the context of this model, we develop and empirically evaluate several prioritization criteria and apply them to four stand-alone GUI and three Web-based applications, their existing test suites and mainly embedded systems. In this model we only run our data collection and test suite prioritization process on seven programs and their existing test suites. An experiment that would be more readily generalized would include multiple programs of different sizes and from different domains. We may conduct additional empirical studies with larger EDS to address this threat each test case has a uniform cost of running (processor time) monitoring (human time); these assumptions may not hold in practice. Second, we assume that each fault contributes uniformly to the overall cost, which again may not hold in practice.

I. Introduction

There are lots of problems involving the consistency of the software during the development cycle. A lot of cost and investment is put forth to reduce the inconsistency in the software which brings out a consistent software. The main objective of our research is in this area of identifying the inconsistencies in software automatically using various tools and techniques. Also we have hereby focused on the automated model change identification which may also help in identifying the inconsistencies automatically.

Author α: *Department of Information Technology College of Computers and Information Technology Taif University, Taif, Saudi Arabia.*
E-mail : m.muzammil@tu.edu.sa
Author σ : *Department of Computer Science College of Computers and Information Technology Taif University, Taif, Saudi Arabia.*
E-mail : Aljahdali@tu.edu.sa

Determining the inconsistencies in software automatically will definitely help in reducing the complexity of software maintenance and as well as enhances the performance of the software.

The main focus of the proposed system of automating the consistency checking is on the UML since UML is the basic for any software development. When we track all the dynamic consistency changes and the rule inconsistencies in the UML we can almost very well say that the software inconsistencies are tracked down, since the software depends on the UML.

In our proposed model of inconsistencies tracking we have laid down the emphasis on the UML rule consistency, UML model changes, Dynamic constraints, meta model constraints, etc.

To identify inconsistencies in an automatable fashion we have devised and applied a view integration framework accompanied by a set of activities and techniques. Our view integration approach exploits the redundancy between views which can be seen as constraints. Our view integration framework enforces such constraints and, thereby, the consistency across views. In addition to constraints and consistency rules, our view integration framework also defines *what* information can be exchanged and *how* information can be exchanged. This is critical for scalability and automates ability.

We made use of many tools those analyses the UML and the model to help us in figuring out all the inconsistencies and changes. The major tool is UML analyzer.

(UML/Analyzer is a synthesis and analysis tool to support model-based software development. It implements a generic view integration framework which supports automated model transformation and consistency checking within UML object and class diagrams as well as the C2SADEL architectural description language).

II. Consistency Checking and Rule Analysis

a) Consistency checking

Consistency checking is a mechanism for checking whether rules are semantically consistent.

Ambiguities can be found either in a single rule or in a set of rules. For example:

- A single rule may contain selfcontradictory conditions and therefore will never apply.
- Two rules may apply to the same object, and set a given attribute to two different values. These rules are conflicting.

Consistency checking goes beyond the simple syntax of rules to consider semantics as well. That is, how the rule behaves during execution. Using Rule Studio, you can choose which checks are carried out. Consistency checks can be categorized into two types:

Checks that analyze an individual rule. These checks are activated when you build the rule and when you run the Consistency checking analysis:

- Rules that are never selected
- Rules that never apply
- Rules with range violation

Checks that analyze rules in relation to other rules. These checks are activated only when you run the Consistency checking analysis.

- Rules with equivalent conditions
- Equivalent rules
- Redundant rules
- Conflicting and self-conflicting rules

Consistency checking reports problems on rules

If there is a rule flow in your rule project, it reports problems on rules that are included in a rule task, and that may be selected at runtime.

It only compares rules that may be in the same task. In the case of a rule task with dynamic selection filtering, the consistency checking mechanism takes into account the rules that are potentially selected by this task. A rule can be potentially selected when it cannot be established that it definitely cannot be selected.

If there is no rule flow in your rule project, all the rules in the project may be selected.

Consistency checking gives an indication of the consistency of your rules but cannot identify all potential problems. An empty Consistency checking report is therefore not a guarantee that there are no problems in the analyzed rules.

b) Rules that are never selected

Rules are reported as "never selected" when they are not part of a rule task and cannot be selected at runtime. For more information, see Rule selection and Rule overriding.

c) Rules that never apply

This occurs when the conditions of the rule can never be met.

Typically, the syntax of such rules is correct but the rules contain common logic errors. For example:

The wrong operator is used to combine condition statements, for example and instead of or: the category of the customer is Gold and the category of the customer is Platinum.

Values are inverted, for example, in the following rule: the age of the customer is between 70 and 50.

Values in the conditions are not within the permitted range.

d) Rules with range violation

In order to reduce the risk of errors, some members can only be assigned values within a specified range. For example, the yearly interest rate on a loan may be limited to values between 0 and 10.

If a rule contains an action that tries to assign a value that is not within the permitted range, Rule Studio displays a range violation error in the report and in the Rule Editor.

e) Rules with equivalent conditions

This occurs when two rules contain condition parts that have the same meaning and their actions are different although conflict.

Rules with equivalent conditions do not necessarily represent an error situation, but they may be good candidates to be merged.

f) Equivalent rules

Equivalent rules are reported when both their conditions and actions are the same.

In the following example, Rule1 and Rule2 are equivalent:

Rule 1
definitions
  set minDiscount to 5
  set ageDiscount to 10
if
  the age of the borrower is more than 65
then
  set the discount to minDiscount + ageDiscount

Rule 2
if
  the age of the borrower is at least 66
then
  set the discount to 15

Although the syntax of these two rules is different, rule analysis evaluates the numeric expressions and reports that the rules are equivalent. You can therefore delete one of them.

Note

Equivalent rules often arise between a decision table that you create and an existing rule.

g) Redundant rules

When two rules have the same actions, one of them becomes redundant when its conditions are included in the conditions of the other.

In the following example, the Else part of Rule2 makes Rule1 redundant:

Rule 1
if
the category of the customer is Gold
then
set the discount to 10

Rule 2
if
the category of the customer is Platinum
then
set the discount to 15
else
set the discount to 10

Although Rule1 is correct, it is redundant and can therefore be deleted.

Note

Redundant rules often arise between a decision table that you create and an existing rule.

h) Conflicting and self-conflicting rules

i. *Conflicting rules*

Rules may conflict when the actions of two different rules set a different value for the same business term (member). Conflicts occur in these two rules in circumstances in which the conditions are equivalent or cover the same values.

Rule 1
if
the loan report is approved and the amount of the loan is at least 300 000
then
set the category of the borrower to Gold

Rule 2
if
the age of the latest bankruptcy of the borrower is less than 1 and the category of the borrower is not Platinum
then
set the category of the borrower to No Category

Rule1 and Rule2 will conflict when the loan report is approved, the amount of the loan is 300000 (or more), the borrower has not had a bankruptcy in the last year, and the category is anything but Platinum. In these specific circumstances, the rules will set the category of the borrower to different values.

Conflicting rules can be corrected by changing the conditions, deleting one of the rules, or setting different priorities on the rules.

ii. *Self-conflicting rules*

A rule is **self-conflicting** when two executions of a rule assign different values to the same member. For example, a self-conflicting rule:

may apply twice on a given working memory (and ruleset parameters)

will set different values to a common attribute
For example:
if
the customer category is Gold
then
set the discount of the cart to the bonus points of the customer

If there are two customer objects with different bonus points in the working memory, the rule is executed twice and a conflict occurs because the two executions of the rule set different values to the discount of the cart.

i) Decision table conflicts

To check decision tables, you need to enable the option Include decision tables and decision trees in the inter-rule checks.

This option allows you to check rules between different decision tables or decision trees, but not within a decision table or decision tree.

Consistency checking then handles decision tables as follows:
It checks individual decision tables/trees for:

never applicable rules

rules with range violation

It checks between two elements. For example, it checks lines between two decision tables/trees, or between a decision table/tree and a BAL rule.

If you do not select this option, rule analysis does not perform any overlapping, redundancy, or conflict checks on decision tables or trees. If you select this option, overlapping, redundancy, or conflict errors are reported on decision tables or trees, except when these errors occur within the same decision table or tree.

III. Tool for Consistency Analysis and Checking

a) UML/Analyzer

Model-Based Software Development is about modeling real problems, solving the model problems, and interpreting the model solutions in the real world. This cycle places a major emphasis on transformation and inconsistency detection between various representations of software systems (e.g., models, diagrams, source code, etc.). UML/Analyzer is a synthesis and analysis tool to support model-based software development. It implements a generic view integration framework which supports automated model transformation and consistency checking within UML object and class diagrams as well as the C2SADEL architectural description language.

The UML/Analyzer tool, integrated with IBM Rational Rose&8482;, fully implements this approach. It was used to evaluate 29 models with tens-of-thousands of model elements, evaluated on 24 types of consistency rules over 140,000 times. We found that the approach provided design feedback correctly and required, in average, less than 9ms evaluation time per model change with a worst case of less than 2 seconds at the expense of a linearly increasing memory need. This is a significant improvement over the state-of-the-art.

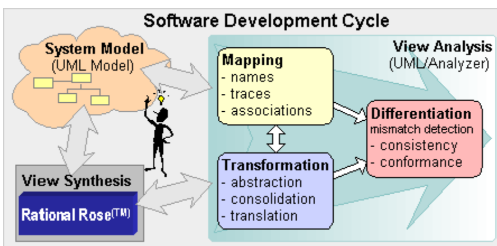


Figure 1: Software Development life cycle

b) UML/Analyzer Architecture

To identify inconsistencies in an automatable fashion we have devised and applied a view integration framework accompanied by a set of activities and techniques. Our view integration approach exploits the redundancy between views which can be seen as constraints. Our view integration framework enforces such constraints and, thereby, the consistency across views. In addition to constraints and consistency rules, our view integration framework also defines *what* information can be exchanged and *how* information can be exchanged. This is critical for scalability and automate ability.

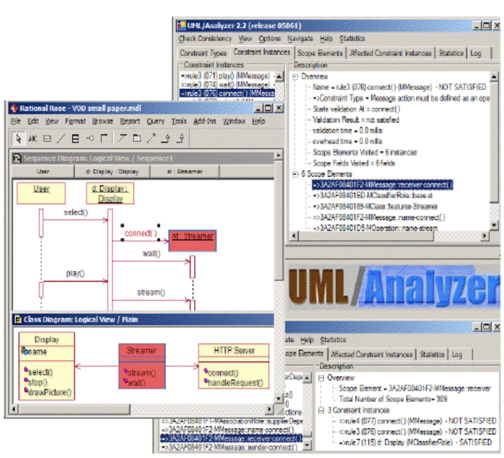


Figure 2 : UML Analyzer

c) UML/Analyser Tool Depicting the inconsistencies in IBM Rational Rose ™

Our approach has the following activities:

1) ***Mapping:*** identifies and crossreferences related modeling elements that describe overlapping and thus redundant pieces of information. Mapping is often done manually via naming dictionaries or traceability matrices (e.g., trace matrices). Mapping assists consistency checking by defining *what* to compare.

2) ***Transformation:*** converts modeling elements or diagrams into intermediate models in such a manner that they (or pieces of them) can be understood easier in the context of other diagram(s). Transformation assists consistency checking by defining *how* to compare.

3) ***Differentiation:*** compares model elements and diagrams with intermediate models that were generated through transformation where differences indicate inconsistencies.

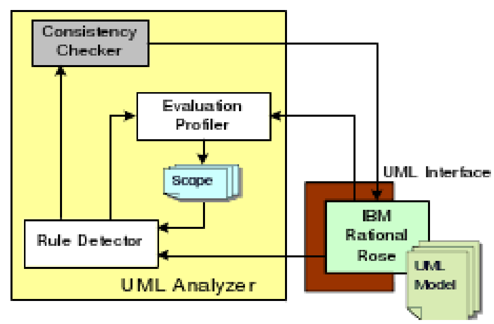


Figure 3 : UML Analyzer with interface

d) Illustration of the problem

The illustration in Fig. 1 depicts three diagrams created with the UML [17] modeling tool IBM Rational Software Modeler. The given model represents an early design-time snapshot of a video-on-demand (VOD) system [4]. The class diagram (top) represents the structure of the VOD system: a Display used for visualizing movies and receiving user input, a Streamer for downloading and decoding movie streams, and a Server for providing the movie data. In UML, a class's behavior can be described in the form of a statechart diagram. We did so for the Streamer class (middle). The behavior of the Streamer is quite trivial. It first establishes a connection to the server and then toggles Simplified UML model of the VOD system between the waiting and streaming mode depending on whether it receives the wait and stream commands.

The sequence diagram describes the process of selecting a movie and playing it. Since a sequence diagram contains interactions among instances of classes (objects), the illustration depicts a particular user invoking the select method on an object, called disp, of type Display. This object then creates a new object, called st, of type Streamer, invokes connect and then wait.

When the user invokes play, object disp invokes stream on object st. These UML consistency rules describe conditions that a UML model must satisfy for it to be considered a valid UML model. Fig. 2 lists 24 such rules covering consistency, well-formedness, and best practice criteria among UML class, sequence, and statechart diagrams. The first four consistency rules are elaborated on for better understanding. Note that these consistency rules apply to UML only. For the other modeling notations, different consistency rules were needed, which are not described here.

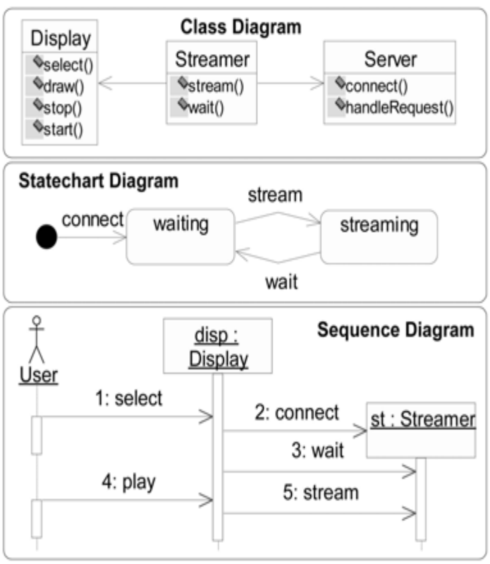


Figure 4 : Class Diagram

A consistency rule may be thought of as a condition that evaluates a portion of a model to a truth value (true or false). For example, consistency rule 1 states that the name of a message must match an operation in the receiver's class.

If this rule is evaluated on the third message in the sequence diagram (the wait message), then the condition first computes operations ¼ message: receiver: base: operations, where message.receiver is the object st (this object is on the receiving end of the message; see arrowhead), receiver.base is the class Streamer (object st is an instance of class Streamer), and base. operations is {stream(),wait()} (the list of operations of the class Streamer). The condition then returns true because the set of operation names (operations> name) contains the message name wait.

IV. Implementation

a) Inconsistencies

We use the term inconsistency to denote any situation in which a set of descriptions does not obey some relationship that should hold between them. The relationship between descriptions can be expressed as a consistency rule against which the descriptions can be checked. In current practice, some rules may be captured in descriptions of the development process; others may be embedded in development tools. However, the majority of such rules are not captured anywhere.

Here are three examples of consistency rules expressed in English:

1. In a dataflow diagram, if a process is decomposed in a separate diagram, the input flows to the parent process must be the same as the input flows to the child data flow diagram.
2. For a particular library system, the concept of an operations document states that user and borrower are synonyms. Hence, the list of user actions described in the help manuals must correspond to the list of borrower actions in the requirements specification.
3. Coding should not begin until the Systems Requirement Specification has been signed off by the project review board. Hence, the program code repository should be empty until the status of the SRS is changed to "approved."

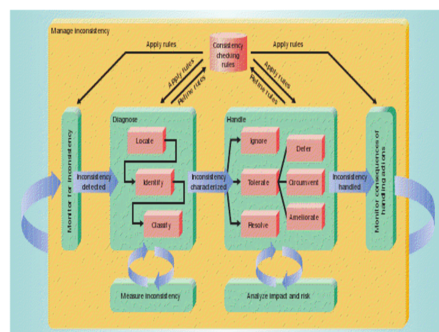


Figure 5 : Manage Inconsistency

In our framework, when you iterate through the consistency management process, you expand and refine the set of consistency rules. You will never obtain a complete set of rules covering all possible consistency relationships in a large project. However, the rule base acts as a repository for recording those rules that are known or discovered so that they can be tracked appropriately.

Consistency rules can emerge from several sources:

- Notation dentitions. Many notations have welldefined syntactic integrity rules. For example, in a strongly typed programming language, the notation requires that the use of each variable be consistent with its declaration.
- Development methods. A method provides a set of notations, with guidance on how to use them together. For example, a method for designing distributed systems might require that for any pair of communicating subsystems, the data items to

be communicated must be defined consistently in each subsystem interface.

- Development process models. A process model typically defines development steps, entry and exit conditions for those steps, and constraints on the products of each step. Local contingencies. Sometimes a consistency relationship occurs between descriptions, even though the notation, method, or process model does not predetermine this relationship. Examples include words used as synonyms, and relationships between timing values in parallel processes.
- Application domains. Many consistency rules arise from domain-specific constraints.

b) Monitoring and diagnosing inconsistency

With an explicit set of consistency rules, monitoring can be automatic and unobtrusive. If certain rules have a high computational overhead for checking, the monitoring need not be continuous—the descriptions can be checked at specific points during development, using a lazy consistency strategy.

Our approach defines a scope for each rule, so that each edit action need be checked only against those rules that include in their scope the locus of the edit action.

When you find an inconsistency, the diagnosis process begins. Diagnosis includes parts of a description have broken a consistency rule;

- identifying the cause of an inconsistency, normally by tracing back from the manifestation to the cause; and
- classifying an inconsistency.

Classification is an especially important stage in the process of selecting a suitable handling strategy.

Inconsistencies can be classified along a number of different dimensions, including the type of rule broken, the type of action that caused the inconsistency, and the impact of the inconsistency.

c) Handling inconsistency

The choice of an inconsistency-handling strategy depends on the context and the impact it has on other aspects of the development process. Resolving the inconsistency may be as simple as adding or deleting information from a software description. But it often relies on resolving fundamental conflicts or making important design decisions. In such cases, immediate resolution is not the best option. You can ignore, defer, circumvent, or ameliorate the inconsistency.

Sometimes the effort to fix an inconsistency is significantly greater than the risk that the inconsistency will have any adverse consequences. In such cases, you may choose to ignore the inconsistency. Good practice dictates that such decisions should be revisited as a project progresses or as a system evolves.

Deferring the decision until later may provide you with more time to elicit further information to facilitate resolution or to render the inconsistency unimportant. In such cases, flagging the affected parts of the descriptions is important.

Sometimes software developers won't regard a reported inconsistency as an inconsistency. This may be because the rule is incorrect or because the inconsistency represents an exception to the rule. In these cases, the inconsistency can be circumvented by modifying the rule or by disabling it for a specific context.

Sometimes, it may be more cost-effective to ameliorate an inconsistency by taking some steps toward a resolution without actually resolving it.

This approach may include adding information to the description that alleviates some adverse effects of an inconsistency and resolves other inconsistencies as a side effect.

d) Measuring inconsistency

For several reasons, measurement is central to effective inconsistency management. Developers often need to know the number and severity of inconsistencies in their descriptions, and how various changes that they make affect these measures. Developers may also use given a choice, which is preferred.

Sometimes developers need to prioritize inconsistencies in different ways to identify inconsistencies that need urgent attention. They may also need to assess their progress by measuring their conformance to some predefined development standard or process model.

The actions taken to handle inconsistency often depend on an assessment of the impact these actions have on the development project. Measuring the impact of inconsistency-handling actions is therefore a key to effective action in the presence of inconsistency. You also need to assess the risks involved in either leaving an inconsistency or handling it in a particular way.

The 24 rules were chosen to cover the needs of our industrial partners. They cover a significant set of rules and we demonstrated that they were handled extremely efficiently. But it is theoretically possible to write consistency rules in a no scalable fashion.

Consistency rules for UML class, sequence, and state chart diagrams. Details sketched for first three rules only. Rules 7 and 8 are classical best practice rules (and not necessarily errors). Rules 9-25 are typical UML well-formedness rules defined in UML 1.3. Different rules apply to other modeling languages (e.g., Dopler).

e) Dynamic Constraints

The research community at large has focused on a limited form of consistency checking by assuming that only the model but not the constraints change (the latter are predefined and existing approaches typically require a complete, exhaustive reevaluation of the entire model if a constraint changes!). *The focus of this work is on how to support dynamically changeable.*

constraints – that is constraints that may be added, removed, or modified at will *without losing the ability for instant, incremental consistency checking and without requiring any additional, manual annotations*. Such dynamic.

Rule	Description and Implementation
1	**Name of message must match an operation in receiver's class** operations=message.receiver.base.operations & base.parents.operations return operations->name->contains(message.name)

2	**Calling direction of message must match an association**
	in=message.receiver.base.incomingAssociations & base.parents.incomingAssociations;
	out=message.sender.base.outgoingAssociations & base.parents.outgoingAssociations;
	return in.intersectsWith(out)

3	**Sequence of object messages must correspond to events** startingPoints = find state transitions equal first message name startingPoints->exists(message sequence equal transition sequence reachable from startingPoint)
4	**Cardinality of association must match sequence interaction**
5	**Statechart action must be defined as an operation in owner's class**
6	**Parent class attribute should not refer to child class**
7	**Parent class should not have a method with a parameter referring to a child class**
8	**Association ends must have a unique name within the association**
9	**At most one association end may be an aggregation or composition**
10	**The connected classifiers of the association end should be included in the namespace of the association**
11	**The class of an association end cannot be an interface if there is an association navigable away from that end**
12	**A classifier may not belong by composition to more than one composite classifier**
13	**Method parameters must have unique names**
14	**Type of Method Parameters must be included in the Namespace of method owner**
15	**A class may not use the same attribute names as outgoing association end names**
16	**No two behavioral features may have the same signature in a classifier**
17	**No two attributes may have the same name within a class**
18	**A classifier may not declare an attributes that has been declared in parents**
19	**Outgoing association ends names must be unique within classifier**
20	**The elements owned by a namespace must have unique names**
21	**An interface can only contain public operations (no attributes)**
22	**No circular inheritance**
23	**A generalizable element may only be a child of another such element of the same kind**
24	**The parent must be included in the Namespace of the GeneralizableElement**

Table 1 : Rules and Description

Constraints arise naturally in many domain specific contexts In addition to meta model constraints, this work also covers application specific model constraints that are written from the perspective of a concrete model at hand (rather than the more generic meta model). We will demonstrate that model constraints can be directly embedded in the model and still be instantly and incrementally evaluated together with meta model constraints based on the same mechanism. For dynamic constraints, any constraint language should be usable. We demonstrate that our approach is usable with traditional kinds of constraint languages (e.g., OCL [5]) and even standard programming languages (Java or C#). Furthermore, our approach is independent of the modeling language used. We implemented our approach for UML 1.3, UML 2.1, Matlab/Stateflow and a modeling language for software product lines.

f) Meta Model and Model Constraints (and Their Instances)

Fig. 6 illustrates the relationships between the meta model/model constraints and their instances.

$$Constraint = < condition, context\ element >$$

Meta Model Constraint: context element is element of Meta model Constraint: context element is element of model Meta model constraints are written from the perspective of a Meta model element.

Many such constraints may exist in a meta model. Their conditions are written using the vocabulary of the meta model and their context elements are elements of the meta model. For example, the context element of constraint C1 in Fig. 3 is a UML Message (a meta model element). This implies that this constraint must be evaluated for every instance of a Message in a given model. In Fig.3 there are three such messages. Model constraints, on the other hand, are written from the perspective of a model element (an instance of a meta model element). Hence, its context element is a model element.

Fig. 6 shows that for every meta model constraint a number of constraint instances are instantiated (top right) – one for each instance of the meta model element the context element refers to. On the other hand, a model constraint is instantiated exactly once – for the model element it defines.

Constraint Instance = <constraint, model element >

While the context elements differ for model and meta model constraints, their instances are alike: the instances of meta model constraints and the instances of model constraints have model elements as their context element. The only difference is that a meta model constraint results in many instances whereas a model constraint results in exactly one instance. Since the instances of both kinds of constraints are alike, our approach treats them in the same manner. Consequently, the core of our approach, the model profiler with its scope elements and reevaluation mechanism discussed above, functions identical for both meta model constraints and model constraints as is illustrated in Fig. 6. The only difference is in how constraints must be instantiated.

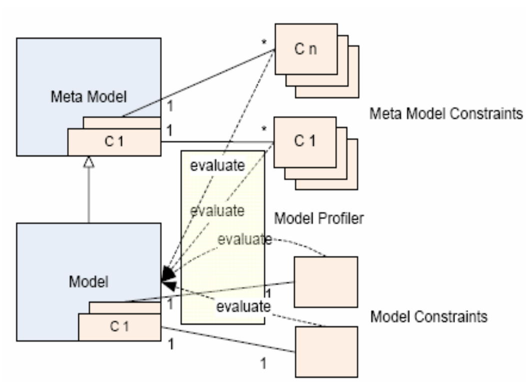


Figure 6 : Relation between meta model and model constraint definitions and constraints

This is discussed further below in more detail.

As discussed above, we support the definition of both meta model and model constraints in Java, C#, and OCL. These languages are vastly different but our approach is oblivious of these differences because it cares only about a constraint's evaluation behavior and not its definition. The key to our approach is thus in the model profiling which happens during the evaluation of a constraint. During the evaluation, a constraint accesses model elements (and their fields).

```
processModelChange(changedElement)
if changedElement was created
 for every definition d where type(d.contextElement)=type(changedElement)
   constraint = new <d, changedElement>
   evaluate constraint
else if changedElement was deleted
 for every constraint where constraint.contextElement=changedElement
   destroy <constraint, changedElement>
for every constraint where constraint.scope contains changedElement
 evaluate <constraint, changedElement>
```

Figure 7 : Process model change

For example, if C1 defined in Fig. 7 is evaluated on message *turnOn()* in Fig.7 (a constraint instance denoted in short as *<C1, turnOn>*), the constraint starts its evaluation at the context element – the message. It first accesses the receiver object *light* and asks for the base class of this object, *WorkroomLight*. Next, all methods of this class are accessed ({*isOn*, *turnOn*, *turnOff*, *setLevel*}) and their names are requested. This behavior is observed and recorded by the model profiler. We define the model elements accessed during the evaluation of a constraint as a *scope* of that constraint. Our approach then builds up a simple database that correlates the constraint instances with the scope elements they accessed (<Model Element, Constraint Instance> pairs) with the simple implication that a constraint instance must be reevaluated if and only if an element in its scope changes:

ScopeElements(Constraint Instance)=Model Elements accessed during Evaluation ReEvaluated Constraints (ChangedElement) = all CI where Scope Elements(CI) includes ChangedElement.

Next, we discuss the algorithm for handling model changes analogous to the discussion above. Thereafter, we discuss the algorithm for handling constraint changes which is orthogonal but similar in structure.

g) Model Change

If the model changes then all affected constraint instances must be re-evaluated. Above we discussed that our approach identifies all affected constraint instances through their scopes, which are determined through the model profiler. In addition to the model profiler, we also require a change notification mechanism to know when the model changes. Specifically, we are interested in the creation, deletion, and modification of model elements which are handled differently. Fig. 7 presents an adapted version of the algorithm for processing model changes published in [10]. If a new model element is created then we create a constraint instance for every constraint that has a type of context element equal to the type of the created model element. The constraint is immediately evaluated to determine its truth value. If a model element is deleted then all constraint instances with the same context element are destroyed. If a model element is changed then we find all constraint instances that contain the model element in their scope and reevaluate them. A model change performed by the user typically involves more than one element to be changed at the same time (e.g. adding a class also changes the *ownedElements* property of the owning package). We start the re-evaluation of constraints only after all changes belonging to a group are processed, i.e. similar to the transactions concept known in databases. Since the model constraints and meta model constraints are alike, our algorithm for handling model changes remains the same.

processModelChange(changedElement)

if changedElement was created for every definition d where type(d.contextElement)=type(changedElement)

constraint = new <d, changedElement>

evaluate constraint

else if changedElement was deleted

for every constraint where

constraint.contextElement=changedElement

destroy <constraint, changedElement>

for every constraint where constraint.scope

contains changedElement

evaluate <constraint, changedElement>

h) Constraint Change

With this paper, we introduce the ability to dynamically create, delete, and modify constraints (both meta model and model constraints). The algorithm for handling a constraint change is presented in Fig. 8. If a new constraint is created then we must

Instantiate its corresponding constraints:

1) for meta model constraints, one constraint is instantiated for every model element whose type is equal to the type of the constraint's context element. For example, if the meta model constraint C1 is created a new (Fig. 3) then it is instantiated three times – once for each message in Fig.3 (<*C1, getDevices*>, <*C1, press*>, <*C1, turnOn*>) because C1 applies to UML messages as defined in its context element.
2) for model constraints, exactly one constraint is instantiated for the model element of the constraint's context element. For example, if the model constraint C4 is defined anew (Fig. 3) then it is instantiated once for the *WorkroomThermostat* as defined in Fig.2 (<*C4, workroomThermostat*>) because this constraint specifically refers to this model element in its context. Once instantiated, the constraints are evaluated immediately to determine their truth values and scopes. If a constraint is deleted then all its instances are destroyed. If a constraint is modified all its constraints are re-evaluated assuming the context element stays the same. If the context element is changed or the constraint is changed from a meta model to a model constraint or vice versa, then the change is treated as the deletion and re-creation of a constraint (rather than its modification).

processConstraintChange(changedDefinition)

if changedDefinition was created for every modelElement of type/instance changedDefinition.contextElement
constraint = new <changedDefinition, modelElement>
evaluate constraint
else if changedDefinition was deleted
for every constraint of changedDefinition, destroy constraint
else if condition of changedDefinition was modified
for every constraint of changedDefinition, evaluate constraint
else
for every constraint of changedDefinition, destroy constraint
for every modelElement of type/instance changedDefinition.contextElement
constraint = new <changedDefinition, modelElement>
evaluate constraint

```
processConstraintChange(changedDefinition)
if changedDefinition was created
 for every modelElement of type/instance changedDefinition.contextElement
   constraint = new <changedDefinition, modelElement>
   evaluate constraint
else if changedDefinition was deleted
 for every constraint of changedDefinition, destroy constraint
else if condition of changedDefinition was modified
 for every constraint of changedDefinition, evaluate constraint
else
 for every constraint of changedDefinition, destroy constraint
 for every modelElement of type/instance changedDefinition.contextElement
   constraint = new <changedDefinition, modelElement>
   evaluate constraint
```

Figure 8 : Algorithm for processing a Constraint change instantly

V. Test Results

a) Computational Scalability

We applied our instant consistency checking tool (the Model/Analyzer) to the 34 sample models and measured the scope sizes S size and the ACRI by considering all possible model changes. This was done through automated validation by systematically changing all fields of all model elements. In the following, we present empirical evidence that S size and ACRI are small values that do not increase with the size of the model.

We expected some variability in Ssize because the sample models were very diverse in contents, domain, and size. Indeed, we measured a wide range of values between the smallest and largest Ssize (average/max), but found that the averages stayed constant with the size of the model. Fig. 9 depicts the values for Ssize relative to the model sizes for the 34 sample models. The figure depicts each model as a vertical range (average to 98 percent maximum), where the solid dots are the average values for any given model. Notice the constant, horizontal line of average scope sizes.

The initial, one-time cost of computing the truth values and scopes of a model is thus linear with the size of the model and the number of rule types OðRT^{+} M$_{size}$ Þ because Ssize is a small constant and constants are ignored for computational complexity.

To validate the recurring computational cost of computing changed truth values and scopes, we next discuss how many CRIs must be evaluated with a single change (ACRI). Since the scope sizes were constant, it was expected that the ACRI would be constant also (i.e., the likelihood for CRIs to be affected by a change is directly proportional to the scope size). Again, we found a wide range of values for ACRI across the many diverse models but confirmed that the averages stayed constant with the size of the model. Fig. 10 depicts the average ACRI through solid dots and their98 percent maximums.

ACRI was computed by evaluating all CRIs and then measuring in how many scopes each model element appeared. The figure shows that in some cases, many CRIs had to be evaluated (hundreds and more). But the average values reveal that most changes required few evaluations (between 3 and 11 depending on the model).

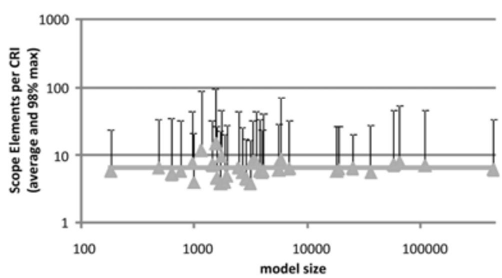


Fig. 9: CRI scope sizes remain constant with model sizes

It depicts the average cost of evaluating a model change based on the type of change. We see that a change t o the association field of an AssociationEnd was the most expensive kind of change, with over 4 ms reevaluation cost, on average. A message name change (as was used several times in this paper) was comparatively cheap, with 0.12 ms to reevaluate, on average. First and foremost, we note that all types of model changes are quite reasonable to reevaluate. This implies that irrespective of how often certain types of changes happen, our approach performs.

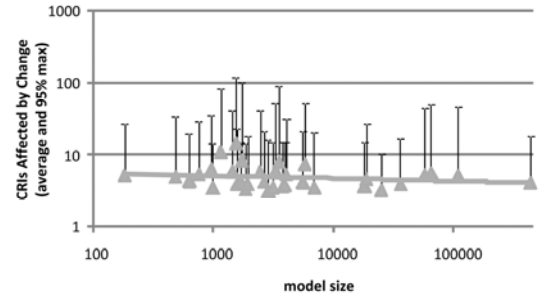


Fig. 10 : Few consistency rule instances are affected by a model change

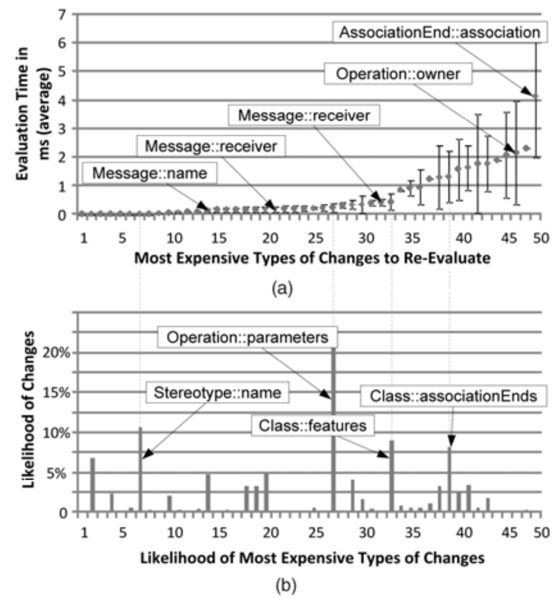


Fig. 11 : The most expensive types of model changes to evaluate and the likelihoods of these changes occurring

Well on all of them. However, not all changes are equally likely and we thus investigated the likelihood of these most expensive types of model changes. For 8 out of the 34 models, we had access to multiple model versions - covering 4,075 changes across them. Fig. 11b depicts that the model changes were unevenly distributed across the types, but as was expected, there is no single (or few) dominant kinds of model changes. Indeed, the most expensive types of model changes never occurred.

Previously, we mentioned that most changes required very little reevaluation time and that there were very rare outliers (0.00011 percent of changes with evaluation time >100 ms). The reason for this is obvious in Fig. 12, where we see that it is exponentially unlikely for CRIs to have larger scope sizes (Fig. 12a) or for changes to affect many CRIs (Fig. 12b). We show this datum to exemplify how similar the 34 models are in that regard, even though these models are vastly different in size, complexity, and domain. Fig. 12a depicts for all 34 models separately what percentage of CRIs (y-axis) had a scope of <¼ 5; 10; 15; . . . scope elements (x-axis).

The table shows that over 95 percent of all CRIs accessed less than 15 fields of model elements (scope elements). Fig. 12b depicts for all 34 models separately what percentage of changes (yaxis) affected <¼ 2; 4; 6; . . . CRIs. The table shows that 95 percent of all changes affected fewer than 10 CRIs (ACRI).

The data thus far considered a constant number of consistency rules (24 consistency rules). However, the number of consistency rules is variable and may change from model to model or domain to domain. Clearly, our approach (or any approach to incremental consistency checking) is not amendable to arbitrary consistency rules. If a rule must investigate all model elements, then such a rule's scope is bound to increase with the size of the model. However, we demonstrated on the 24 consistency rules that

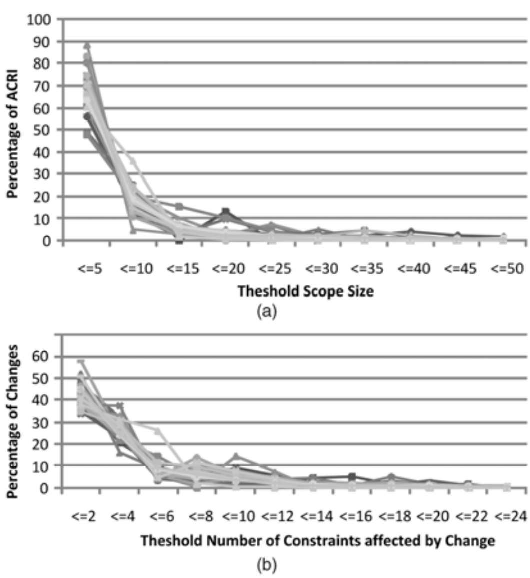


Fig.12. (a) : The number of model elements accessed by constraints and (b) the number of constraints affected by changes as percentages relative to thresholds

Rules typically are not global; they are, in fact, surprisingly local in their investigations. This is demonstrated in Fig. 13, which depicts the cost of evaluating changes for each consistency rule separately. Still, each consistency rule takes time to evaluate and Fig. 13 is thus an indication of the increase in evaluation cost in response to adding new consistency rules.

We see that the 24 consistency rules took, on average, 0.004-0.21 ms to evaluate with model changes. Each new consistency rule thus increases the evaluation time of a change by this time (assuming that new consistency rules are similar to the 24 kinds of rules we evaluated). The evaluation time thus increases linearly with the number of consistency rules (RT#).

It is important to note that the evaluation was based on consistency rules implemented in C#. Rules implemented in Java were slightly slower to evaluate but rules implemented in OCL [38] were comparatively expensive due to the high cost of interpreting them.

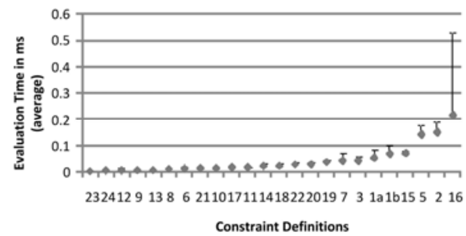


Fig. 13 : The cost of adding a consistency rule

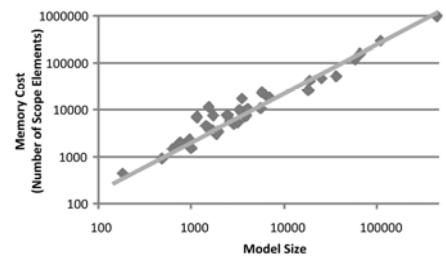


Fig. 14 : Memory cost increases linearly with model size

b) Positive result regarding the memory cost and usability

i. *Memory Cost*

On the downside, our approach does require additional memory for storing the scopes. Fig. 14 depicts the linear relationship between the model size and this memory cost. It can be seen that the memory cost rises linearly. This should not be surprising given that the scope sizes are constant with respect to the model size but the number of CRIs increases linearly. As with the evaluation time, this cost also increases with the number of consistency rules (RT#). The memory cost is thus RT# + Ssize . For scalability, this implies a quite reasonable trade-off between the extensive performance gains over a linear (and thus scalable) memory cost. To put this rather abstract finding into a practical perspective, the scope is maintained as a simple hash table referencing the impacted CRIs in form of arrays. With the largest model having over 400,000 scope elements, each of which affects fewer than 10 CRIs, the memory cost is thus equivalent to 400,000 arrays of fewer than 10 CRIs each- quite manageable with today's computing resources. The memory cost stays the same if the scope is stored persistently, in which case the recomputation of the scope upon model load is no longer required.

ii. *Usability*

One key advantage of our approach is that engineers are not limited by the modeling language or consistency rule language. We demonstrated this by implementing our approach on UML 1.3, UML 2.1, Matlab/Stateflow, and Dopler Product Line, and using a wide range of languages to describe consistency rules (from Java, C# to the interpreted OCL). But, most significantly, engineers do not have to understand our approach or provide any form of manual annotations (in addition to writing the consistency rule) to use it. These freedoms are all important for usability.

This paper does not address how to best visualize inconsistencies graphically. Much of this problem has to do with human-computer interaction and future work will study this. This paper also does not address downstream economic benefits: For example, how does quicker (instant) detection of inconsistencies really benefit software engineering at large. How many p roblems are avoided, how much less does it cost to fix an error early on as compared to later on? These complex issues have yet to be investigated.

However, as an anecdotal reference, it is worth pointing out that nearly all programming environments today support instant compilation (and thus syntax and semantic checking), which clearly benefits programmers. We see no reason why these benefits would not apply to modeling.

VI. Conclusion

The main issues addressed in this paper includes – identifying the inconsistencies correctly and quickly in an automated fashion by reducing the complexity, cost and the effort Next, to evaluate the consistency rules which are not necessarily to be written in special language and special annotations our approach used a form of profiling to observe the behavior of the consistency rules during evaluation. We demonstrated on 34 large-scale models that the average model change cost 1.4 ms, 98 percent of the model changes cost less than 7 ms, and that the worst case was below 2 seconds. It is very significant to understand that our approach maintains a separate scope of model elements for every application (instance) of a consistency rule. This scope is computed automatically during evaluation and used to determine when to reevaluate the rule. In the case of an inconsistency, this scope tells the engineer all of the model elements that were involved. Moreover, if an engineer should choose to ignore an inconsistency (i.e., not resolve it right away), an engineer may use the scopes to quickly locate all inconsistencies that directly relate to any part of the model of interest. This is important for living with inconsistencies but it is also important for not getting overwhelmed with too much feedback at once.

This paper significantly identifies the dynamic model changes and a wide variety of consistency rules and the proposals were made for automatic detection and tracking of those inconsistencies and model changes that are static as well as dynamic considering also the cost and the efficiency factors of the automated system that is to be inbuilt as an embedded system to perform the task of automatic detection and embarking techniques to solve the inconsistencies and the model changes in any software development process by using the UML diagram as the base and UML analyzer for evaluation of the constraints and the results are then processed for further actions.

VII. Future Work

We cannot guarantee that all consistency rules can be evaluated instantly. The 24 rules of our study were chosen to cover the needs of our industrial partners. They cover a significant set of rules and we demonstrated that they were handled extremely efficiently. But it is theoretically possible to write consistency rules in a nonscalable fashion, although it must be stressed that of the hundreds of rules known to us, none fall into this category. It is future work to discuss how to best present inconsistency feedback visually to the engineer. Also, the efficiency of our approach depends, in part, on how consistency rules are written.

References Références Referencias

1. U.A. Acar, A. Ahmed, and M. Blume, "Imperative Self-Adjusting Computation," Proc. 35th ACM SIGPLAN-SIGACT Symp. Principles of Programming Languages, pp. 309-322, 2008.
2. R. Balzer, "Tolerating Inconsistency," Proc. 13th Int'l Conf. Software Eng., pp. 158-165, 1991.
3. B. Belkhouche and C. Lemus, "Multiple View Analysis and Design," Proc. Int'l Workshop Multiple Perspectives in Software Development, 1996.
4. X. Blanc, I. Mounier, A. Mougenot, and T. Mens, "Detecting Model Inconsistency through Operation-Based Model Construc- tion," Proc. 30th Int'l Conf. Software Eng., pp. 511-520, 2008.
5. B.W. Boehm, C. Abts, A.W. Brown, S. Chulani, B.K. Clark, E. Horowitz, R. Madacy, D. Reifer, and B. Steece, Software Cost Estimation with COCOMO II. Prentice Hall, 2000.
6. L.C. Briand, Y. Labiche, and L. O'Sullivan, "Impact Analysis and Change Management of UML Models," Proc. Int'l Conf. Software Maintenance, p. 256, 2003.
7. L.A. Campbell, B.H.C. Cheng, W.E. McUmber, and K. Stirewalt, "Automatically Detecting and Visualising Errors in UML Diagrams," Requirements Eng. J., vol. 7, pp. 264-287, 2002.
8. B.H.C. Cheng, E.Y. Wang, and R.H. Bourdeau, "A Graphical Environment for Formally Developing Object-Oriented Software," Proc. Sixth Int'l Conf. Tools with Artificial Intelligence, pp. 26-32,1994.
9. D. Dhungana, R. Rabiser, P. Gru¨ nbacher, K. Lehner, and C. Federspiel, "DOPLER: An Adaptable Tool Suite for Product Line Engineering," Proc. 11th Int'l Software Product Line Conf., pp. 151-152, 2007.

10. S. Easterbrook and B. Nuseibeh, "Using ViewPoints for Incon- sistency Management," IEE Software Eng. J., vol. 11, pp. 31-43, 1995.
11. A. Egyed, "Automated Abstraction of Class Diagrams," ACM Trans. Software Eng. And Methodology, vol. 11, pp. 449-491, 2002.
12. A. Egyed, "Instant Consistency Checking for the UML," Proc. 28th Int'l Conf. Software Eng., pp. 381-390, 2006.
13. A. Egyed, "Fixing Inconsistencies in UML Design Models," Proc. 29th Int'l Conf. Software Eng., pp. 292-301, 2007.
14. A. Egyed and B. Balzer, "Integrating COTS Software into Systems through Instrumentation and Reasoning," Int'l J. Automated Software Eng., vol. 13, pp. 41-64, 2006.
15. A. Egyed, E. Letier, and A. Finkelstein, "Generating and Evaluating Choices for Fixing Inconsistencies in UML Design Models," Proc. 23rd Int'l Conf. Automated Software Eng., 2008.
16. W. Emmerich, "GTSL—an Object-Oriented Language for Specification of Syntax Directed Tools," Proc. Eighth Int'l Workshop Software Specification and Design, pp. 26-35, 1996.
17. S. Fickas, M. Feather, and J. Kramer, Proc. ICSE-97 Workshop Living with Inconsistency, 1997.
18. A. Finkelstein, D. Gabbay, A. Hunter, J. Kramer, and B. Nuseibeh, "Inconsistency Handling in Multi-Perspective Specifications," IEEE Trans. Software Eng., vol. 20, pp. 569-578, 1994.
19. C. Forgy, "Rete: A Fast Algorithm for the Many Pattern/Many Object Pattern Match Problem," Artificial Intelligence, vol. 19, pp. 17-37, 1982.
20. I. Groher, A. Reder, and A. Egyed, "Instant Consistency Checking of Dynamic Constraints," Proc. 12th Int'l Conf. Fundamental Approaches to Software Eng., 2010.
21. J. Grundy, J. Hosking, and R. Mugridge, "Inconsistency Manage- ment for Multiple-View Software Development Environments," IEEE Trans. Software Eng., vol. 24, no. 11, pp. 960-981, Nov. 1998.
22. A.N. Habermann and D. Notkin, "Gandalf: Software Development Environments," IEEE Trans. Software Eng., vol. 12, no. 12, pp. 1117-1127, Dec. 1986.
23. S.M. Kaplan and G.E. Kaiser, "Incremental Attribute Evaluation in Distributed Language-Based Environments," Proc. Fifth Ann. Symp. Principles of Distributed Computing, pp. 121-130, 1986.
24. M. Lee, A.J. Offutt, and R.T. Alexander, "Algorithmic Analysis of the Impacts of Changes to Object-Oriented Software," Proc. 34th Int'l Conf. Technology of Object- Oriented Languages and Systems, pp. 61-70, 2000.
25. M. Lindvall and K. Sandahl, "Practical Implications of Trace- ability," J. Software—Practice and Experience, vol. 26, pp. 1161-1180, 1996.
26. A.K. Mackworth, "Consistency in Networks of Relations," J. Artificial Intelligence, vol. 8, pp. 99-118, 1977.
27. C. Nentwich, L. Capra, W. Emmerich, and A. Finkelstein, "xlinkit: A Consistency Checking and Smart Link Generation Service," ACM Trans. Internet Technology, vol. 2, pp. 151-185, 2002.
28. C. Nentwich, W. Emmerich, and A. Finkelstein, "Consistency Management with Repair Actions," Proc. 25th Int'l Conf. Software Eng., pp. 455-464, 2003

This page is intentionally left blank

Global Journal of Computer Science and Technology
Software & Data Engineering
Volume 12 Issue 11 Version 1.0 Year 2012
Type: Double Blind Peer Reviewed International Research Journal
Publisher: Global Journals Inc. (USA)
Online ISSN: 0975-4172 & Print ISSN: 0975-4350

Diligence of Domain Engineering in Accounting Management System

By Mukesh Kumar, Dr. Parveen Kumar & Seema

NIMS University, Shobha Nagar, Jaipur (Rajasthan) India

Abstract - This paper presents on domain feature modeling, domain architecture design and domain implementation in an enterprise. This paper demonstrates the accounting management feature modeling based on the extended (Feature-Oriented Domain Analysis) FODA method and system architecture of accounting management domain, integrates Aspect Object Oriented Programming technology with domain implementation, and designs a whippersnapper AOP framework based on the object proxy pattern to separates crosscutting concerns in the domain implementation phrase. Research result shows this method can effectively seal insulate and abstract variability in requirements of accounting management domain, instruct the designing and implementation of accounting management components, get the requirement of software reuse, resource sharing and collaboration in accounting management domain.

Keywords : Feature-Oriented Domain Analysis, Aspect Object Oriented Programming, whippersnapper.

GJCST-C Classification : D.2.1

DILIGENCE OF DOMAIN ENGINEERING IN ACCOUNTING MANAGEMENT SYSTEM

Strictly as per the compliance and regulations of:

Open Association of Research Society, USA

Research | Diversity | Ethics

Diligence of Domain Engineering in Accounting Management System

Mukesh Kumar [α], Dr. Parveen Kumar [σ] & Seema [ρ]

Abstract - This paper presents on domain feature modeling, domain architecture design and domain implementation in an enterprise. This paper demonstrates the accounting management feature modeling based on the extended (Feature-Oriented Domain Analysis) FODA method and system architecture of accounting management domain, integrates Aspect Object Oriented Programming technology with domain implementation, and designs a whippersnapper AOP framework based on the object proxy pattern to separates crosscutting concerns in the domain implementation phrase. Research result shows this method can effectively seal insulate and abstract variability in requirements of accounting management domain, instruct the designing and implementation of accounting management components, get the requirement of software reuse, resource sharing and collaboration in accounting management domain.

Keywords : Feature-Oriented Domain Analysis, Aspect Object Oriented Programming, whippersnapper.

I. Introduction

Domain engineering is a reusable approach that focus on a selected application domain as like inventory control, finance management, word processing etc. The motto of domain engineering is find, catalog, construct and broadcast set of software artifacts that could apply for future software in specialized application domain. In domain engineering, we perform domain analysis and capture domain knowledge in the form of reusable software assets. By reusing the domain assets, an organization will be able to deliver a new product in the domain in a shorter time and at a lower cost. In industry, domain engineering forms a basis for software product line practices. Domain engineering is most often divided into three phases: domain analysis, domain design, and domain implementation. At present, from the point of domain engineering, little research has been carried on the accounting management domain. Based on the real project, this paper introduces domain engineering method into the development of accounting management system. In the domain analysis phrase, we use the FODA method to analyze the accounting management domain, expand its feature-oriented modeling method, establish the feature model of accounting management domain; in the domain design phrase, we design multi-tier system architecture of accounting management domain; In the domain implementation phrase, We combine AOP technology with OOP technology, separate crosscutting multi modules concerns in software, reduce the dependence between components effectively. Practice has proved the systems developed by this method have a better performance of maintainability, extendibility and reusability.

Author α : Computer Science & Engineering. NIMS University, Shobha Nagar, Jaipur (Rajasthan) India. E-mail : mukesharya79@yahoo.com
Author σ : Computer Science Department, Meerut Institute of Engineering and Technology, Meerut (Uattar Pardesh) India. E-mail : pk223475@gmail.com
Author ρ : Lecturer in Department of Computer Science & Engineering, KCGPW Ambala City, (Haryana) India. E-mail : ahlawat6789@yahoo.com

II. Analysis of Account Management Domain

a) Feature Oriented Domain Analysis

A method specifically designed for DA is the Feature Oriented Domain Analysis (FODA) method developed at the SEI. This process is for domain analysis which supports the discovery, analysis, and documentation of commonality and differences within a domain. The feature oriented concept emphasis on findings the capabilities that are normally expected in applications in a given domain. The FODA domain model captures the similarities and differences among domain assets in terms of a set of related features. A feature is a distinctive aspect, quality, or characteristic of the domain asset. The features identified by the FODA method can be used to parameterize the system product line and Implementations of the domain assets. The features differentiating domain entities arise from differences in capabilities, operating environments, domain technology, implementation techniques, etc., i.e., a range of possible implementations within the domain. A specific implementation consists of a consistent set of feature values describing its capabilities. The feature diagram depicts the decomposition of features into sub-features in a hierarchical way. For each sub-feature below a certain feature it can be specified if it is compulsory, second-stringer or optional. The graphical notations introduced in are used here. We first briefly describe the representations used in illustrated in Figure 1. The compulsory feature is represented by being attached to an edge ending with a filled circle. So the feature F consists of both K1 and K2 in this case, and the feature instances here are {F, K1, K2}. The optional feature is

Global Journal of Computer Science and Technology (C) Volume XII Issue XI Version I Year 2012

represented by being attached to an edge ending with an unfilled circle. So the feature F may or may not contain K1. The optional feature instances here are {F, K2} and {F, K1, K2}. The second-stringer feature is represented by connecting edges with an arch. So the feature F consists of exactly one of its child features. The second-stringer feature instances here are {F, K1} and {F, K2}. Note that if K1 is optional while K2 is compulsory, then the second-stringer feature instances here are {F}, {F, K1} and {F, K2}, because the child feature instances derived from the K1 side contain an empty feature. The OR feature is represented by connecting edges with a filled arch. The OR feature instances here are {F, K1}, {F, K2} and {F, K1, K2}. If there is an optional child feature, then the OR representation is actually equivalent to the situation that all the child features are optional, i.e., the OR feature instances will be {F}, {F, K1}, {F, K2} and {F, K1, K2}.

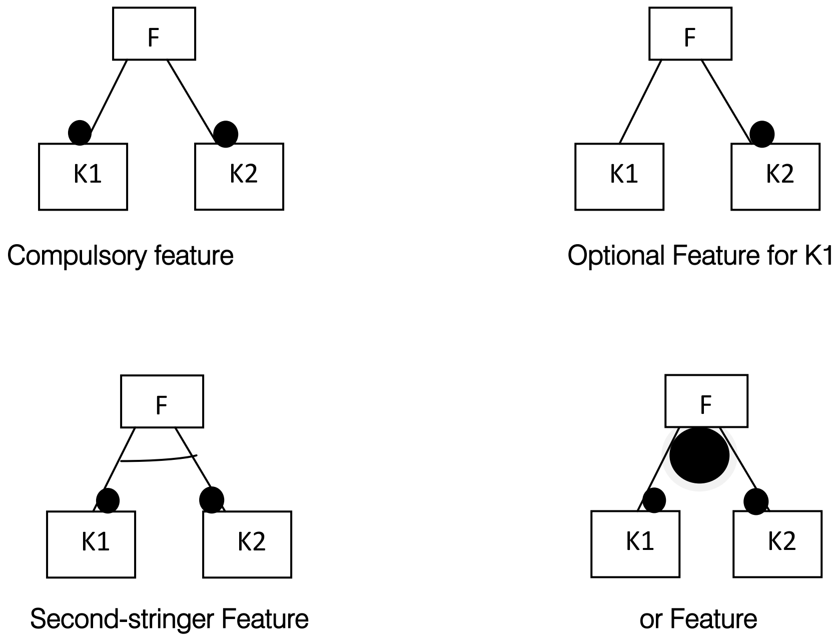


Figure 1:

b) Feature Modeling of Account Management Domain

Through domain analysis, we find common and variant features of different account management systems, from different requirements: business requirement, user requirement, and functional requirement. Business requirement depicts business ability that the software system should have. User requirement depicts the interaction process between user and system, and this process may reflect the generally accepted business process in this domain. Functional requirement depicts functions that software system must have in order to realize the specific business requirements. Through domain analysis, we divide the service of account management domain into the following types: Account Drafting, Account Auditing, Account Implementation, Account Adjustment, Account Analysis, Account assessment. Among them, account assessment is optional features.

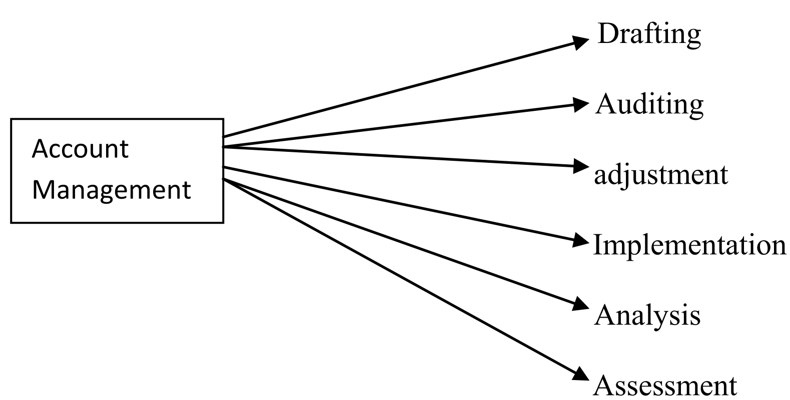


Figure 2 : Major services of account management domain

The second analysis is to identify functional features which the service has, analyze the specific functions which systems must have in order to complete special service. Taking account implementation control service as example, its functional layer includes compulsory features and optional features. And as shown in Figure 2, Compulsory features include execution account drafting, execution account auditing, execution account management and query analysis.

Optional features include data import. Compulsory features, namely common features, exist in each member system of the special domain, but optional features are one type of representation style of variant features, and only exist in parts of member system of the special domain. Optional features represents the variability which is relative to whole features, its introduction enables the feature model to respond the different system's diversity of domain, and makes the feature model to have better tailorability and expansibility.

The third Behavior characteristics layer analysis. The task of behavior characteristics layer analysis is to identify behavior characteristics what the function should be there, analyze behavior features of the early stages of functional implementation, such as preconditions of functional implementation, preparatory works; analyze the principal behavior characteristics of function part, find its outstanding features and its possible variability; analyze behavior features of the later period of function implementation, such as the postposition condition of functional implementation and the domination shift after the functional implementation.

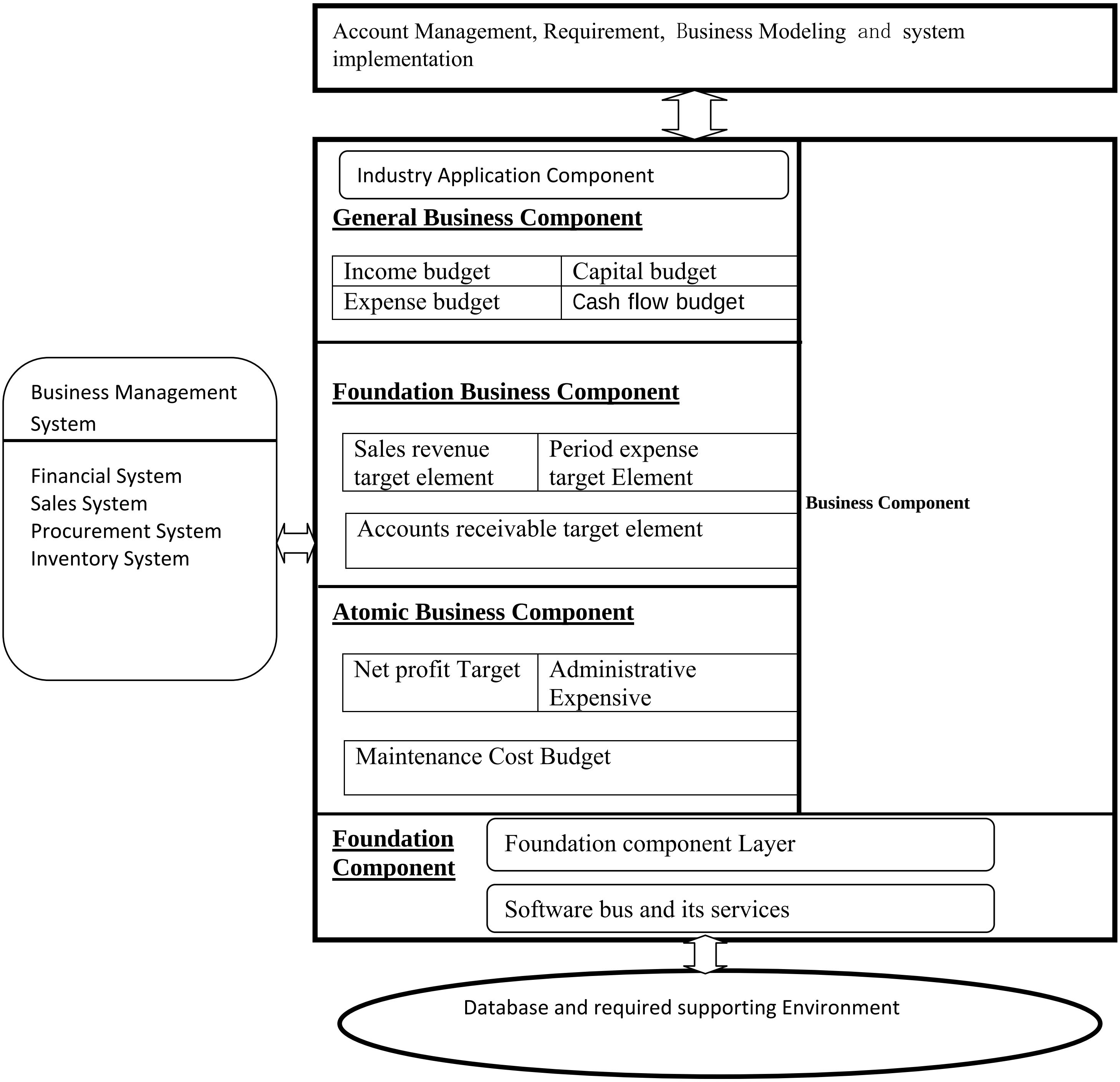
III. Account Management Domain with Architectural Design

Domain designing is the core architecture for a family of applications according to domain analysis model, namely a Domain-Specific Software Architecture (DSSA), and based on the DSSA, We can identify, develop and organize the reusable components. According to the requirements defined in the domain analysis stage, considering the actual development environment (such as operating system, database, communication mechanism, middleware, and so on, this paper designs Account Management domain architecture, This architecture uses the hierarchical architecture style. The hierarchical architecture style can avoid system component’s coupling, protect and divide system function, improve maintainability, reusability and extendibility of software.

This domain architecture has five components: foundation component layer, atomic business component layer, foundation business component layer, general business component layer, industry application component layer.

(1) Industry application component:- This component is designed to satisfy special industry business requirements. It can be encapsulated by one or more atomic business components, or by one or more foundation business components, and even also can be combined by atomic business components, foundation business components and general business components.

(2) General business component: - This component is a subsystem level application component which is formed by encapsulates foundation business components or atomic business components, such as revenue budget components, investment budget components, capital budget components, cash flow budget components.



Picture 2 : Architecture of Account Management System based on Component

(3) Foundation business component: - On the basis of atomic business component, these components are able to complete certain business functions through aggregation of some atomic components. This type of component faces to application directly, such as sales revenue target components, period expense target components, business interface components.

(4) Atomic business component: - According to the decomposition business object, this is made by encapsulation of various types of foundation components. This level usually includes the following component types: representation components (forms according to object's method) data components (forms according to object's attribute).

(5) Foundation component: - This component is the lowest level in this architecture, and it is the core support to implement the business object function. It takes Database, Document, Mathematical formula, Documentary evidence and so on as the object, carries

on the code level encapsulation according to component standard, forms general representation components, data components, operational components or generic component template. The components of previous layer may call it directly.

IV. Implementation of Budget Management Domain

In the part of domain design, we have putted required and harder structural DSSA and assigned the stable parts to the budget management domain system architecture and the variable parts to components. In the process of component implementation, we normally use OOP Object-Oriented Programming) for the simplifying the things and encapsulating the class. Aspect-oriented Programming (AOP) is a new programming technology which compensates the weakness of Object-Oriented Programming (OOP) for applying common behavior that spans multiple related object models. AOP introduces Aspect, it packages the behavior which impacts multiple classes into a reusable model, it allows programmers to model crosscutting concerns and eliminates the code tangling and scattering caused by OOP, the code is more readable and easier to maintain. The key to achieve AOP is to intercept normal method call. In order to complete some extra requirements, we will need to add extra features transparent "weaving" to these methods. Generally speaking, the weaving method includes two major types: Static weaving method and Dynamic weaving method. Static weaving method usually need to extend compiler's function, directly weave codes into the appropriate weaving point by modifying byte codes(Java) or IL code(.Net). Or, we need to add new syntax structure for original language to support AOP. As for dynamic weaving method, there are many specific implementation methods. In the Java platform, we can use Proxy pattern, or custom Class Loader to implement AOP features. Generally, at the .Net platform, the following methods can be used to achieve the dynamic weaving method:

1. Use Context Attribute and Context Bound Object to intercept the object methods.
2. Use Emit technology in the run-time to build new class which codes are woven into.
3. Use Proxy pattern

V. Conclusion

In this paper it is depicts the application of domain engineering in account management system development. Domain analysis method of FODA this paper has extended its feature oriented modeling method and design multi-layer framework according to the domain analysis result. At the domain implementation segment we applied a lightweight AOP framework with the name of SJAOP. This technology with the help of OOP separates crosscutting multi modules concerns in software, reduces the dependence between components effectively, and implements the system with a better performance of maintainability, extendibility and reusability.

References Références Referencias

1. James A. Hess, William E. Novak, A. Spencer Peterson,"Feature-Oriented Domain Analysis (FODA) Feasibility Study", Carnegie Mellon Software Engineering Institute (SEI), USA,2000.
2. Steven Kelly and Juha-Pekka Tolvanen. Domain-Specific Modeling : Enabling Full Code Generation. Wiley-Interscience, 2008.
3. Wang Qian-xiang,Wu Qiong,Li Ke-qin,Yang Fu-qing,"An Object-Oriented Method for Domain Engineering",Journal of Software, vol.13, no.10, pp.1977-1984, 2002.
4. Li Ke-qin, Chen Zhao-liang,Mei Hong,Yang Fu-qing,"An outline of Domain Engineering", Computer Science, vol.26,no.5,pp.21-25, 1996
5. Wang Fan,Tan Guo-zhen,Wang Hao,He Qin-lai,"Feature Modeling Method Oriented to Traffic Domain Component", Computer Engineering, vol.35,no.1,pp.280-282, 2009.
6. Zhengmin Liu, "Research on Enterprise Budget Management System Based On Domain Engineering". International Journal of Digital Content Technology and its Applications, Vol. 5, No. 9, pp. 88 ~ 94, 2011
7. Kai Chen, Janos Sztipanovits, and Sandeep Neema. Towards a semantic anchoring infrastructure for domain-specific modeling languages. In International Conference on Embedded Software, pages 35–43, 2005.

This page is intentionally left blank

Global Journal of Computer Science and Technology
Software & Data Engineering
Volume 12 Issue 11 Version 1.0 Year 2012
Type: Double Blind Peer Reviewed International Research Journal
Publisher: Global Journals Inc. (USA)
Online ISSN: 0975-4172 & Print ISSN: 0975-4350

Data Stream Mining: A Review on Windowing Approach

By Pramod S. & O.P.Vyas
Ravishankar Shukla University, Raipur

Abstract - In the data stream model the data arrive at high speed so that the algorithms used for mining the data streams must process them in very strict constraints of space and time. This raises new issues that need to be considered when developing association rule mining algorithms for data streams. So it is important to study the existing stream mining algorithms to open up the challenges and the research scope for the new researchers. In this paper we are discussing different type windowing techniques and the important algorithms available in this mining process.

Keywords: Data Stream Mining, Association Rule Mining, Data Mining, Online Data Mining.

GJCST-C Classification: D.2.2

DATA STREAM MINING A REVIEW ON WINDOWING APPROACH

Strictly as per the compliance and regulations of:

Data Stream Mining: A Review on Windowing Approach

Pramod S. [α] & O.P.Vyas [σ]

Abstract - In the data stream model the data arrive at high speed so that the algorithms used for mining the data streams must process them in very strict constraints of space and time. This raises new issues that need to be considered when developing association rule mining algorithms for data streams. So it is important to study the existing stream mining algorithms to open up the challenges and the research scope for the new researchers. In this paper we are discussing different type windowing techniques and the important algorithms available in this mining process.

Keywords : Data Stream Mining, Association Rule Mining, Data Mining, Online Data Mining.

I. Introduction

Once any company decided to use the data mining system for daily operations, management will be concerned with the system performance for their environment. If the mining take place on the historic data then the result could be used for the future strategic decision making[1]. But the amount of data collected over time is increased in daily basis in the database then that can reduce the accuracy of the result[2] of the mining process. This is where online data mining can play a vital role to improve the mining result and its accuracy[3].

a) Frequent Itemset Mining Approaches

The tentative nature of frequent itemset mining normally results in a large number of frequent itemset generations. The increase in the number of frequent itemset generated will result in the degradation of mining efficiency. The frequent closed itemset[9] mining is the solution for the above said problem. The FCI is a non redundant representation of the set of frequent itemsets[10]. The commendable reduction in the size of the result set leads to improved performance in the speed and memory usage. Different efficient FCI algorithms[8,11,12] are proposed by different authors. We noticed that the FCIs approach could not be applied over land mark window since the number of FCIs approaches that of frequent itemsets when the window becomes very large.

Author α : Computer Science and Information Technology Department at Ravishankar Shukla University, Raipur, C.G. and working as Associate Professor in Information Technology Department in Christian College of Engineering and Technology, Bhilai, C.G, India.
E-mail : pramodsnair@yahoo.com
Author σ : Professor in Indian Institute of Information Tecnology-Allahabad, U.P., India. E-mail : dropvyas@gmail.com

There is one another frequent item set mining technique called Frequent Maximal Item set[7]. Compare with the Frequent Closed Item set mining technique it will generate comparatively less number of item sets, due to this reason it is significantly more efficient [13] in terms of both CPU and memory. But the disadvantage of FMI mining is that it lose the frequency information of the subset of FMIs so the error bound will also increased. There are many concise representations of frequent item sets are proposed [14, 15, 16, 17, 18, 19], these are significantly saving memory space, CPU and shows better accuracy. This technique could be applied in stream mining with the efficient incremental technique and batch processing.

II. Windowing Approach to Data Stream Mining

One of the main issues in the stream data mining is to find out a model which will suit the extraction process of the frequent item set from the streaming in data. There are three stream data processing model[20] that are Landmark window, Damped window and Sliding window model. A transaction data stream is a sequence of incoming transactions and an excerpt of the stream is called a window. A window, W, can be either time-based or count-based, and either a landmark window or a sliding window. W is time-based if W consists of a sequence of fixed-length time units, where a variable number of transactions may arrive within each time unit. W is count-based if W is composed of a sequence of batches, where each batch consists of an equal number of transactions. W is a landmark window if $W = (T_1, T_2, ..., T)$; W is a sliding window if $W = (T_{T-w+1}, ..., T_T)$, where each T_i is a time unit or a batch, T_1 and T_T are the oldest and the current time unit or batch, and w is the number of time units or batches in the sliding window, depending on whether W is time-based or count-based. Note that a count-based window can also be captured by a time-based window by assuming that a uniform number of transactions arrive within each time unit.

The frequency of an item set, X, in W, denoted as freq(X), is the number of transactions in W support X. The support of X in W, denoted as sup(X), is defined as freq(X)/N, where N is the total number of transactions received in W. X is a Frequent Item set (FI) in W, if $sup(X) \geq \sigma$, where σ $(0 \leq \sigma \leq 1)$ is a user-specified minimum support threshold. X is a Frequent Maximal Item set (FMI) in W, if X is an FI in W and there exists no item set Y in W such that $X \subset Y$. X is a Frequent Closed Item set (FCI) in W, if X is an FI in W and there exists no item set Y in W such that $X \subset Y$ and freq(X) = freq(Y).

a) Landmark Window Concept

In this section we will discuss some of important land mark window algorithms. One of the algorithm proposed by Manku and Motwani[23] is a lossy counting approximation algorithm. It will compute the approximate set of frequent item sets over the entire stream so far. In this algorithm the stream is divided into sequence of buckets. The lossy counting algorithm processes a batch of transactions arriving at a particular time. In this paper they are maintaining the item set, the frequency of item set and the error as the upper bound of the frequency of the item set. This algorithm uses three different modules, Buffer, Trie and Set Gen. The Buffer module keeps filling the available memory with the incoming transactions. This module frequently computes the frequency of every item in the current transactions and prune if it is less than N. The Trie module maintains set D, as a forest of prefix trees. The Trie forest as an array of tuples (X, freq(X), err (X), level) that correspond to the pre-order traversal of the forest, where the level of a node is the distance of the node from the root. The Trie array is maintained as a set of chunks. On updating the Trie array, a new Trie array is created and chunks from the old Trie are freed as soon as they are not required.

All the item sets in the current batch having the support will be generated by the Set Gen module. The Apriori-like pruning[21] will help to avoid the generation of superset of an item set if the frequency less than β in the current batch. The Set Gen implemented with the help of Heap queue. Set Gen repeatedly processes the smallest item in Heap to generate a 1-itemset. If this 1-itemset is in Trie after the Add Entry or the Update Entry operation is utilized, Set Gen is recursively invoked with a new Heap created out of the items that follow the smallest items in the same transactions. During each call of Set Gen, qualified old item sets are copied to the new Trie array according to their orders in the old Trie array, while at the same time new item sets are added to the new Trie array in lexicographic order. When the recursive call returns, the smallest entry in Heap is removed and the recursive process continues with the next smallest item in Heap.

The quality of the approximation mining results by using the relaxed minimum support threshold ∈ leads to the extra usage of memory and the processing power. That is, the smaller relaxed minimum support leads to increase of number of sub-FIs generated, so the increase of memory and the extra usage of processing power. , if ∈ approaches σ, more false-positive answers will be included in the result, since all sub-FIs whose computed frequency is at least $(\sigma - \in)N \approx 0$ are displayed while the computed frequency of the sub-FIs can be less than their actual frequency by as much as σN. The same problem is in other mining algorithms [21, 22, 23, 24, 13, 4] that use a relaxed minimum support threshold to control the accuracy of the mining result.

One of the algorithm called DSM-FI developed by Li[13], is to mine an approximate set of FIs over the entire history of the stream. This algorithm is used a prefix-tree based in memory data structure. DSM-FI is also using the relaxed minimum support threshold and all the generated FIs are stored in the IsFI-forest. The DSM-FI consists of Header Table(HT) and Sub-Frequent Itemsets tree(SFI-tree). For every unique item in the set of sub-FIs it inserts an entry with frequency, batch id and head link, it increments otherwise. The DSM-FI frequently prunes the items that are not satisfied the minimum support.

One of the approximation algorithm developed by Lee[4] used the compressed prefix tree structure called CP-tree. The structure of the CP-tree is described as follows. Let D be the prefix tree used in estDec. Given a merging gap threshold δ, where $0 \leq \delta \leq 1$, if all the itemsets stored in a subtree S of D satisfy the following equation, then S is compressed into a node in the CP-tree.

$$\frac{freq_T(X) - freq_T(Y)}{N_T} \leq \delta$$

Where X is the root of S and Y is an item set in S. Assume S is compressed into a node v in the CP-tree. The node v consists of the following four fields: item-list, parent-list, freqTmax and freqTmin where v.item-list is a list of items which are the labels of the nodes in S, v. parent-list is a list of locations (in the CP-tree) of the parents of each node in S, v. freqTmax is the frequency of the root of S and freqTmin is the frequency of the right-most leaf of S.

The use of the CP-tree results in the reduction of memory consumption, which is important in mining data streams. The CP-tree can also be used to mine the FIs, however, the error rate of the computed frequency of the FIs, which is estimated from freqTmin and freqTmax, will be further increased. Thus, the CP-tree is more suitable for mining FMIs.

b) Sliding Window Concept

The sliding window model processes only the items in the window and maintains only the frequent item sets. The size of the sliding window can be decided according to the applications and the system resources. The recently generated transactions in the window will influence the mining result of the sliding windowing, otherwise all the items in the window to be maintained. The size of the sliding window may vary depends up on the applications it may use. In this section we will discuss some of the important windowing approaches for stream mining.

An in memory prefix tree based algorithm proposed by Chi[26, 22] following the windowing approach to incrementally update the set of frequent closed item sets over the sliding window . The data structure used for the algorithm is called as Closed Enumeration Tree (CET) to maintain the dynamically selected set of item set over the sliding window. This algorithm will compute the exact set of frequent closed item sets over the sliding window. The updation will be for each incoming transaction but not enough to handle the handle the high speed streams.

One another notable algorithm in the windowing concept is estWin[3]. This algorithm maintains the frequent item sets over a sliding window. The data structure used to maintain the item sets is prefix tree. The prefix tree holds three parameters for each items set in the tree, that are frequency of x in current window before x is inserting in the tree, that is freq(x). The second is an upper bound for the frequency of x in the current window before x is inserted in the tree, err(x). The third is the ID of the transaction being processed, tid(x).b. The item set in the tree will be pruned along with all supersets of the item set, we prune the item set X and the supersets if $tid(X) \leq tid_1$ and $freq(X) < \lceil \in N \rceil$, or (2) $tid(X) > tid_1$ and $freq(X) < \lceil \in (N - (tid(X) - tid_1)) \rceil$. The expression $tid(X) > tid_1$ means that X is inserted into D at some transaction that arrived within the current sliding window and hence the expression $(N-(tid(X)-tid_1))$returns the number of transactions that arrived within the current window since the arrival of the transaction having the ID tid(X). We note that X itself is not pruned if it is a 1-itemset, since estWin estimates the maximum frequency error of an itemset based on the computed frequency of its subsets [84] and thus the frequency of a 1-itemset cannot be estimated again if it is deleted.

c) Damped Window Concept

In this section we will discuss some of the notable Damped window algorithms. The estDec[5] algorithm proposed to reduce the effect of the old transactions on the stream mining result. They have used a decay rate to reduce the effect of the old transactions and the resulted frequent item sets are called recent frequent Item sets. The algorithm, for maintaining recent FIs is an approximate algorithm that adopts the mechanism to estimate the frequency of the item sets.

The use of a decay rate diminishes the effect of the old and obsolete information of a data stream on the mining result. However, estimating the frequency of an item set from the frequency of its subsets can produce a large error and the error may propagate all the way from the 2-subsets to the n-supersets, while the upper bound is too loose. Thus, it is difficult to formulate an error bound on the computed frequency of the resulting item sets and a large number of false-positive results will be returned, since the computed frequency of an item set may be much larger than its actual frequency. Moreover, the update for each incoming transaction (instead of a batch) may not be able to handle high-speed streams.

Another approximation algorithm[6] uses a tilted time window model . In this frequency FIs are kept in different time granularities such as last one hour, last two hours, last four hours and so on. The data structure used in this algorithm is called FP-Stream. There are two components in the FP-Stream which are pattern tree based prefix tree and tilted time window which is at the end node of the path. The pattern tree can be constructed using the FP-tree algorithm[25]. The tilted time window guarantees that the granularity error is at most T/2, where T is the time units.

The updation of the frequency record will be done by shifting the recent records to merge with the older records. To reduce the number of frequency records in the tilted-time windows, the old frequency records of an item set, X, are pruned as follows. Let freqj(X) be the computed frequency of X over a time unit Tj and Nj be the number of transactions received within Tj , where $1 \leq j \leq \tau$. For some m, where $1 \leq m \leq \tau$, the frequency records freq1(X), . . . ,freqm(X) are pruned if the following condition holds:

$$\exists n \leq \tau, \forall i, 1 \leq i \leq n, freqi(X) < \sigma Ni \text{ and}$$

$$\forall l, 1 \leq l \leq m \leq n, \sum_{j-1}^{l} \mathbf{freq}_j(x) < \in \sum_{j=1}^{j} N_J$$

The FP-stream mining algorithm computes a set of sub-FIs at the relaxed minimum support threshold, ∈ , over each batch of incoming transactions by using the FI mining algorithm, FP-growth [25]. For each sub-FI X obtained, FP-streaming inserts X into the FP-stream if X is not in the FP-stream. If X is already in the FP-stream, then the computed frequency of X over the current batch is added to its tilted-time window. Next, pruning is performed on the tilted-time window of X and if the window becomes empty, FP-growth stops mining supersets of X by the Apriori property [2]. After all sub-FIs mined by FP-growth are updated in the FP-stream, the FP-streaming scans the FP-stream and, for each item set X visited, if X is not updated by the current batch of transactions, the most recent frequency in X's tilted-time window is recorded as 0. Pruning is then performed on X. If the tilted-time window of some item set visited is empty (as a result of pruning), the item set is also pruned from the FP-stream.

The tilted-time window model allows us to answer more expressive time-sensitive queries, at the expense of some frequency record kept for each item set. The tilted-time window also places greater importance on recent data than on old data as does the

Global Journal of Computer Science and Technology (C) Volume XII Issue XI Version I Year 2012

sliding window model; however, it does not lose the information in the historical data completely. A drawback of the approach is that the FP-stream can become very large over time and updating and scanning such a large structure may degrade the mining throughput.

III. Conclusion

In this paper we have discussed some of the issues of the windowing concept for the online stream mining to develop an effective, performance oriented algorithm. We also discussed some of the important windowing algorithms in the different windowing concept and reviewed, for some extend, how the existing important algorithms could handle these different issues. The further study can be done on this field to develop an effective algorithm in the data stream mining. We have discussed the way the different algorithms handle the data stream mining so that the researchers can analyze and study further for the research work.

References Références Referencias

1. Fernando Crespoa, Richard Weberb. “A methodology for dynamic data mining based on fuzzy clustering”, Fuzzy Sets and Systems 150 (2005) 267–284.
2. David Hand, Heikki Mannila, Padhraic Smyth. “Principles of Data Mining”, ISBN: 026208290 MIT Press, Cambridge, MA, 2001.
3. Maria Halkidi, “Quality assessment and Uncertainty Handling in Data Mining Process” http://www.edbt2000. unikonstanz.de/phd-workshop/papers/Halkidi.pdf.
4. B. Liu, W. Hsu, and Y. Ma. Integrating Classification and Association Rule Mining. In Proc. of KDD, 1998.
5. J. H. Chang and W. S. Lee. Finding Recent Frequent Itemsets Adaptively over Online Data Streams. In Proc. of KDD, 2003.
6. C. Giannella, J. Han, J. Pei, X. Yan, and P. S. Yu. Mining Frequent Patterns in Data Streams at Multiple Time Granularities. In Kargupta et al.: Data Mining: Next Generation Challenges and Future Directions, MIT/AAAI Press, 2004.
7. D. Lee and W. Lee. Finding Maximal Frequent Itemsets over Online Data Streams Adaptively. In Proc. of ICDM, 2005.
8. Y. Chi, H. Wang, P. S. Yu and R. R. Muntz. Catch the Moment: Maintaining Closed Frequent Itemsets over a Data Stream Sliding Window. In KAIS, 10(3): 265-294, 2006.
9. N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal. Discovering Frequent Closed Itemsets for Association Rules. In Proc. of ICDT, 1999.
10. M. Zaki. Generating Non-Redundant Association Rules. In Proc. of KDD, 2000.
11. M. Zaki and C. J. Hsiao. CHARM: An Efficient Algorithm for Closed Itemset Mining. In Proc. of SDM, 2002.
12. J. Wang, J. Han, and J. Pei. CLOSET+: Searching for the Best Strategies for Mining Frequent Closed Itemsets. In Proc. of KDD, 2003.
13. K. Gouda and M. Zaki. Efficiently Mining Maximal Frequent Itemsets. In Proc. of ICDM, 2001.
14. T. Calders and B. Goethals. Mining All Non-derivable Frequent Itemsets. In Proc. Of PKDD, 2002.
15. J. F. Boulicaut, A. Bykowski and C. Rigotti. Free-Sets: a Condensed Representation of Boolean Data for the Approximation of Frequency Queries. In DMKD, 7(1):5-22, 2003.
16. J. Pei, G. Dong, W. Zou, and J. Han. Mining Condensed Frequent-Pattern Bases. In KAIS, 6(5): 570-594, 2004.
17. D. Xin, J. Han, X. Yan, and H. Cheng. Mining Compressed Frequent-Pattern Sets. In Proc. of VLDB, 2005.
18. F. Bonchi and C. Lucchese. On Condensed Representations of Constrained Frequent Patterns. In KAIS, 9(2): 180-201, 2005.
19. J. Cheng, Y. Ke, and W. Ng. δ-Tolerance Closed Frequent Itemsets. To appear in Proc. of ICDM, 2006.
20. R. Jin and G. Agrawal. An Algorithm for In-Core Frequent Itemset Mining on Streaming Data. In Proc. of ICDM, 2005.
21. LTC Bruce D. Caulkins, J.Lee, M.Wang, “A Dynamic Data Mining Technique for Intrusion Detection Systems, 43rd ACM Southeast Conference, March 18-20, 2005, Kennesaw, GA, USA. Copyright 2005 ACM 1-59593-059-0/05/0003.
22. Y. Chi, H. Wang, P. S. Yu and R. R. Muntz. Catch the Moment: Maintaining Closed Frequent Itemsets over a Data Stream Sliding Window. In KAIS, 10(3): 265-294, 2006.
23. H. Mannila, H. Toivonen, and A. I. Verkamo. Discovery of Frequent Episodes in Event Sequences. In DMKD, 1:259-289, 1997.
24. Graham Cormode, S.Muthukrishnan; What's Hot and What's Not: Tracking Most Frequent Items Dynamically; ACM Transactions on Database Systems; March 2005.
25. Cheqing Jin, Weining Qian, Chaofeng Sha, Jeffrey X. Yu, Aoying Zhou; Dynamically Maintaining Frequent Items over a Data Stream; Int'l Conf. on Information and Knowledge Management; 2003.
26. Hua-Fu Li, Suh-Yin Lee, and Man-Kwan Shan; An Efficient Algorithm for Mining Frequent Itemsets over the Entire History of Data Streams; Int'l Workshop on Knowledge Discovery in Data Streams; Sept. 2004.

Global Journal of Computer Science and Technology
Software & Data Engineering
Volume 12 Issue 11 Version 1.0 Year 2012
Type: Double Blind Peer Reviewed International Research Journal
Publisher: Global Journals Inc. (USA)
Online ISSN: 0975-4172 & Print ISSN: 0975-4350

An Extension of Description Logic Al

By F. Mamache

Université des Sciences et de la Technologie Houari Boumediene Alger Algérie

Abstract - The research in the domain of knowledge representation and reasoning has always concentrated on the methods that give a good description in the domain where they are able to be used to construct intelligent applications. Description Logics are a family of languages of knowledge representation which can be used to represent knowledge of a field of applications by clear, formal and structured means. In this paper, we give an overview of what are Description Logics and their actual applications in different fields and a brief idea of extensions of Description Logic AL, as we also introduce two operators, the operator less and operator more, which allow us to obtain a new extension of the Description Logic AL.

Keywords : Artificial Intelligence, Description Logics, Knowledge Representation, Semantic, Subsumption, Classification.

GJCST-C Classification : D.2.2

AN EXTENSION OF DESCRIPTION LOGIC AL

Strictly as per the compliance and regulations of:

An Extension of Description Logic AL

F. Mamache

Abstract - The research in the domain of knowledge representation and reasoning has always concentrated on the methods that give a good description in the domain where they are able to be used to construct intelligent applications. Description Logics are a family of languages of knowledge representation which can be used to represent knowledge of a field of applications by clear, formal and structured means. In this paper, we give an overview of what are Description Logics and their actual applications in different fields and a brief idea of extensions of Description Logic AL, as we also introduce two operators, the operator less and operator more, which allow us to obtain a new extension of the Description Logic AL.

Keywords : Artificial Intelligence, Description Logics, Knowledge Representation, Semantic, Subsumption, Classification.

I. Introduction

Research in the domain of knowledge representation and reasoning always concentrates on the methods that give a good description in the domain where they are able to be used to construct intelligent applications. By intelligent applications, we refer to systems able to find implicit consequences to represent knowledge explicitly.

Description Logic systems produce to their users' possibilities of varied inferences that deduct the implicit knowledge of the knowledge represented explicitly. Description Logics are a family of languages of knowledge representation which can be used to represent the knowledge of a field of applications by clear, formal and structured means.

These are logical formalisms of representation which distinguish themselves from Networks and Frames by their formal semantic that is based on logic.

In this paper, we give an overview of what are Description Logic and their applications in different fields. We notice several domains of applications, some include Software Engineering, Configuration, Medicine, Numeric libraries and Information Systems based on Web. there exists other domains of applications where the Description Logics have an significant role, as the field which include the Treatment of Natural Language and Management of Database. We give in this paper a brief idea of extensions of Description Logic AL, as we also introduce two operators, operator less and operator more, which allow us to obtain a new extension of the Description Logic AL.

II. Origin of Description Logics

Description Logics Dls or terminology logics are a family of languages of knowledge representation which can be used to represent the knowledge of a field of applications by clear, formal and structured means. Description Logics difier of their predecessors, such as Networks and Frames, given that they are equipped of formal logic based on semantic. We find three generations of systems. In the following, we will see their historic evolution.

a) *Pre-description logic systems*

Description Logics are formalisms of knowledge representation based on KL-One language. KL-One language is considered as root of the family of all languages. The Networks that are at the origin of the language KL-One, were introduced in 1966 as a representation of the basic concepts of the English words, and become a popular type of structures to represent a wide variety of concepts of the applications in Arti_cial Intelligence.

KL-One language introduced most key notions of Dls:

- Notion of concepts and roles
- Notions of restrictionvalue and the restrictionnumber that has an important role in the usage of the roles in the de_nition of the concepts and,
- Inference of subsumption and classi_cation. Kl−One is based on the subsumption : it's a system of structured inheritance and it is at the origin of a family of languages such as : KL-Two, Krypton, Loom, Kandor, Back, Nikl, Classic and Kriss.

b) *Description logics Systems*

The last pre-Description logics originate directly from KL-One that itself is a direct result from formal analysis. Description Logics systems that will follow as future generation will result from more theoretical research on terminology logics than of examination consequences of KLOne and of other latest systems. We can notice three approaches for the implementation of the reasoning services :

- The first one can be considered as limited and complete or as systems that are studied by restriction of the set of the concepts so that the subsumption can be calculated eficiently, possible in polynomial time. The system Classic is an example of this approach.
- The second approach designated as expressive and incomplete, since the idea is to furnish an expressive language and an effective reasoning. The inconvenience is, nevertheless, that the

LRIA Laboratoire de Recherche en Intelligence Arti_cielle. Faculté d'Electronique et d'Informatique. Département d'Informatique, USTHB, BP 32 El Alia, Bab Ezzouar, Alger (Email : amamache @usthb.dz).

algorithm of reasoning proves to be incomplete in these systems. An example of this system is the system Loom

- In the third approach, we have the characterized systems as being expressive and complete. They are not effective like those of the preceding approaches.

c) Current Description Logics systems

In the current generation of Knowledge Representation Systems based on the DLs (DLKRS), the need of complete algorithms of the expressive languages became focal points. The expressivity of the language of Description Logics is necessary to reason on the data models. The semi-structured data contributed to the identification of the most of the important extensions for practical applications.

III. Introduction to Description Logics

a) Introduction

A knowledge system is a program able to reason on an application domain to solve a particular problem, using knowledge related to the studied field. The knowledge of the domain is represented by entities which have syntactic descriptions which are associated to semantics. It does not exist any universal method to conceive such systems, but there is a stream of current and active research developed that were nourished by the studies carried out on the logic of the predicates, the networks semantic and the languages of Frames. This research gave rise to a family of languages of representation called Description Logics. In the formalism of Description Logics, a concept allows to represent a set of individuals, while a role represents a binary relation between individuals. A concept corresponds to a generic entity of an application domain and an individual to a particular entity, i.e, instance of a concept. Concepts, roles and individuals obey to the following principles:

- Concept and a role possess a structural description, elaborated from some constructors. A semantic is associated to each description of concept and role by an interpretation. The manipulations operated on the concepts and roles, are realized in agreement with this semantic.
- The knowledge are taken into account according to several levels : The representation and the manipulation of concepts and roles result from terminological level, the description and the manipulation of individuals result from factual level or assertions level. The terminological level is qualified by T-Box and the factual level by A-Box.
- Subsumption allows organizing concepts and roles by generality level: intuitively, a concept C subsumes a concept D if C is more general than D in the view where the set of the individuals represented by C contains the set of the individuals represented by D. A knowledge basis is composed of a hierarchy of concepts and of a hierarchy of roles.
- The operations which are at the basis of the terminological reasoning are the classification and instantiation. Classification applies to the concepts, if necessary to the roles and allows determining the position of a concept and of a role in their respective hierarchies. Instantiation allows finding the concepts of which an individual is susceptible to be an instance.

b) Basis of Description Logics

The basic sets that are defined and used in Description Logic are concepts and roles. Concept denotes a set of individuals and a role denotes a binary relation between individuals. Concept possesses a structured description which is constructed using a set of constructors introducing the roles associated to the concept and the restrictions attached to these roles. The restrictions carry generally on the co-domains of the role, which is the concept which the role establishes a relation, and the cardinality of the role, which fixes the minimal and maximal number of elementary values that, can take the role. The elementary values are instances of concepts or many values that result from basic types as integer, real, and chains of characters.

The concepts can be primitive or defined. The primitive concepts are comparable to atoms and are used as a basis for construction of the definite concepts. A role can be primitive or defined and can have a structural description, where appear the properties associated to the role.

The constructor and indicates that a concept is constructed from a conjunction of concepts that are the ascendants of the new concept- and the constructor all specifies the co-domain of a relation. The constructor not express the negation and does apply only to primitive constructors. The constructors at−last and at−most specify the cardinality of the role which they are associated and respectively indicate the minimum number and the maximum number of elementary values of the role.

The associated characteristics to a primitive concept are necessary: an individual x that is an instance of a primitive concept P possesses the characteristics of P. The associated characteristics to a defined concept D are necessary and sufficient: an individual x that is an instance of a defined concept D possesses the characteristics of D, and inversely, the fact that an individualy possesses the set of the associated characteristics to D suffices to infer that y is an instance of D. This distinction is at the basis of the classification process. Concepts are defined in a declarative manner (in a declaratory way) and the (installation) set up of the defined concepts in the hierarchy of the concepts is carried out under the check (control) of the classification process.

c) Description of concepts and roles : syntax

There is several description languages of concepts and roles. In follows, we introduce a minimal language called AL, which is enriched progressively by new constructors. The language AL is based on the languages FL and FL^- presented below, which are the languages for which were established the first theoretical results on the Dls.

C,D→	A|
Top|	⊤
Botto|⊥	
(and C D)|	C⊓D
(not A)|	¬A
(all r C)|	∀r.C
(some r)	∃r
Lispian*syntax*	Germany*syntax*

The grammar of the description language of AL, with Lispian and Germany syntaxes, C and D are concepts names, A a primitive concept name and r a primitive role name.

– The constructor Top (>) denotes the most general concept.
– The BOTTOM concept (?) denotes the least specific concept. Intuitively, the Top extension includes all possible individuals while that of BOTTOM is empty.
– The operator of conjunction: The operator and (u) allows us to build a new concept corresponding to conjunction of definite concepts. Example: The concept Person and Mother gives a new concept Female.
– The constructor not (¬) corresponds to the negation and relates only to the primitive concepts. Example: The concept Person and not Female can be expressed by: Person u ¬ Female.
– The operator of disjunction: The operator or (t) allows us to build a new concept corresponding to disjunction of definite concepts. Example: The concept person that are Male or Female can be represented by: Male t Female.
– Restrictions of roles: The connectors at−last, at−most and all are called restrictions of roles.

Restrictions of cardinality at−last (≥) and at−most (≤) specify the cardinality of role with which they are associated and indicate the minimal and maximum number of elementary values of the role. They limit the sets of values max and min of a role on a concept or an individual. Example: The concept: (≥3 has Child) ⊓ (≤2 has Female Relative) represent the concept: an individual having at − least 3 children and more 2 daughters.

To represent concepts like "In the system, there is less equations than unknowns ", and "an individual having more girls than boys" where the minimal number and the maximum number are not known, we thought to introduce others restrictions operators.

The constructors less and more indicate the cardinality of the role to which they are associated without specifying the minimal number or the maximum number of elementary values of the role. Example: The concept: (system (has (equations) < (unknowns)) (system (less (equations, unknowns))) represent the concept:" the system has less equations than unknowns ". Example: The concept: (has Child (daughters) > (sons)) (has Child (more (daughters, sounds))) represent the concept: "an individual having more daughters than sons ".

– The universal quantification all (∀r.c) specifies the co-field of role r. Example: The concept: (All children are female) is expressed by: ∀ has Child. Female.
– The existential quantification some(∃r) introduced the role r and affirms the existence of (less) one couple of individuals in relation via r. The operator of restriction of existential values: Allows to write the concept (an individual having a girl) like ' 9 has Child. Female'. Language AL = {> ?, u B, ¬ A, 8 r :C, 9 r} can be enriched by the following constructors:
– The negation of primitive or defined concepts, which is noted (not C) or ¬ C. The corresponding extension of AL is ALL = AL [{¬ C}.
– The disjunction of concepts, which is noted (or C D) or C t D. The corresponding extension of AL is ALU = AL [{C t D}.
– The typed existential quantification, noted (c − some r C) or 9 r: C. The corresponding extension of AL is ALE = AL [{9 r: C}.
– The typed existential quantification 9 r: C introduces a role r of co-field C and imposes the existence of less one couple of individuals (x, y) in relation by the role r, where C is the type of y.
– The cardinality on the roles is noted (at − leastnr) or ≤nr, and (at − mostnr) or ≥nr.The corresponding extension of AL is ALN = AL [{≤ nr, ≥ nr}.
– The constructors ≤ nr and ≥nr fix the cardinality minimum and maximum elementary values numbers of the role which they are associate. In particular, construction (∃r) is equivalent to construction (≥1 r).
– The comparison of the cardinality on the roles is noted r_1 less r_2 or $r_1 < r_2$, and (r_1 more r_2) or $r_1 > r_2$. The corresponding extension of AL is ALC = AL [{$r_1 < r_2$, $r_1 > r_2$}.
– The conjunction of roles is noted (and r_1 r_2) or $r_1 \setminus r_2$, the roles r_1 and r_2 being primitive. The corresponding extension of AL is ALR = AL [{$r_1 \setminus r_2$}.

Global Journal of Computer Science and Technology (C) Volume XII Issue XI Version I Year 2012

d) Concepts and roles description : Semantic

i. *Interpretation in ALLNRC*

A semantic is associated to descriptions of concepts and roles: Concepts are interpreted like subsets of a field of interpretation _ and roles like subsets of product $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$.

The concepts are interpreted like subsets of interpretation field $\Delta^{\mathcal{I}}$ and roles like subsets of product $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$.

For a concept C, CI corresponds to the subset of the elements of field $\Delta^{\mathcal{I}}$, and for a role r, rI corresponds to the subset of the couples of elements of product $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$.

The following definition is given within the framework of language ALCNRI Definition 1 (Interpretation) An interpretation I = (I,.I) is the data of a set called interpretation field and a interpretation function .I which fact of corresponding to a concept a subset of $\Delta^{\mathcal{I}}$ and to a role a subset of $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$, so that following equations are satisfied:

$$\top^{\mathcal{I}} = \Delta^{\mathcal{I}}$$
$$\bot^{\mathcal{I}} = \emptyset$$
$$(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$$
$$(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}$$
$$(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} - C^{\mathcal{I}}$$
$$(\forall\, r.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \;/\; \forall\, y : (x,y) \in r^{\mathcal{I}} \rightarrow y \in C^{\mathcal{I}}\}$$
$$(\exists\, r.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \;/\; \exists\, y : (x,y) \in r^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\}$$
$$(\geq nr)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \;/ |\{y \in \Delta^{\mathcal{I}} \;/\; (x,y) \in r^{I}\} \mid \geq n\}$$
$$(\leq nr)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \;/\; |\{y \in \Delta^{\mathcal{I}} \;/\; (x,y) \in r^{I}\} \mid \leq n\}$$
$$(r_1 > r_2)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \;/ |\{y \in \Delta^{\mathcal{I}} : (x,y) \in r_1^{\mathcal{I}}\} \,|>|\, \{z \in \Delta^{\mathcal{I}} : (x,z) \in r_2^{\mathcal{I}}\}|\}$$
$$(r_1 < r_2)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \;/ |\{y \in \Delta^{\mathcal{I}} : (x,y) \in r_1^{\mathcal{I}}\} \,|<|\, \{z \in \Delta^{\mathcal{I}} : (x,z) \in r_2^{\mathcal{I}}\}|\}$$
$$(r_1 \sqcap \cdots \sqcap r_n)^{\mathcal{I}} = r_1^{\mathcal{I}} \cap \cdots \cap r_n^{\mathcal{I}}$$

IV. Modelling in Description Logics

At the beginning, DLs were regarded particularly as effective for fields where knowledge could be organized in a hierarchical structure, based on the relation ' is-a '.

The ability to represent and reason on taxonomies in DLs, justified their use as language of modelling in the study and maintenance of organisms of structured knowledge in a hierarchical way as well as their adoption like language of representation for formal ontology.

So that the designers are able to use DLs to model applications, it is significant that the concepts of Description logic are easily understandable; this will facilitate the use of the effective tools.

There are two principal alternatives to grow the use of DLs like language of modelling:

i. To provide a syntax which be like the natural language,
ii. To implement interfaces where the user can specify the structures of representation through graphic operations.

To model in DLs requires of the designer to specify the concepts of the field of discussion, to characterize their relationships to the other concepts and to specify also individuals.

V. Applications Developed With Description Logic Systems

We notice several applicability, some including Software, Engineering, Configuration, Medicine, Numerical Libraries and Information systems based on Web. There is several other applicability where DLs play a significant role, as the fields which include Treatment of Natural Language and Management of the Data bases. Some applications, whose creation lasted several years, arrived only at the level of prototype, but several among have the totality of the industrial systems several projects on the treatment natural language based on DLs were undertaken; some reached the level of industrial applications. We will see now, briefly, some fields of research which have relation with DLs.

a) The natural language

The use of DLs in the treatment of the natural language for knowledge representation can be used to communicate the meaning of the sentences. This knowledge is typically concerned by the meaning of the words (dictionary), and by the context i.e. a representation of the situation and the field of dialogue. The expressivity of the natural language also carries out to investigations concerning the extensions of DLs, such as for example it reason by defect. Work on the natural language required construction ontology.

b) Management of Data Bases

Knowledge and reasoning systems based on Dls, DL – KRS, management of data bases systems DBMS are present and very useful. A DBMS takes care of the persistence of the data and the management of a broad quantity of these data, whereas a DL – KRS manages intentional knowledge by keeping the base of knowledge in memory DLs are equipped with tools of reasoning which can revive the phase of conceptual modelling of some advantages, compared with traditional languages whose role is limited, concerning modelling. The second aspect of the improvement of the DBMS with DLs requires the query language.

c) Software Engineering

The Software Engineering is one of the first applicability of DLs. The principal idea was to implement an information system Software or a system which could help the developer of the software to find information in a wide Software system. One of the most original applications of DLs is Lassiesystem. Lassiesystem had a considerable success but ended up falling because of difficulty of the maintenance of its knowledge base. The idea of an information Software system and use of DLs survived like particular application and was used later by others Systems.

d) Configuration

The task of the configuration is to find a set of components which can be suitably connected in order to carry out a system which satisfies a given specification. The task of the configuration appears in many industrial fields like telecommunication, car industry and constructions of buildings. By using DLs, we can exploit the capacity to classify the components and to organize in a taxonomy.

e) Medicine

Medicine is also a field where the expert systems were developed since 1980, however, the complexity of the medical field requires a variety in the use of the DL – KRS. The need to deal with large range for knowledge bases (100000 concepts) leads to development of specialized systems such as Galen.

VI. Conclusion

Description Logics are responsible for several basic concepts in Knowledge Representation and Reasoning. The most significant aspect of work on DLs was certainly the union between the theory and practice. Descriptions Logics are not only theoretical formalism reserved to the theorists of Knowledge Representation, research around Description Logics is very active and has practical and theoretical aiming. Thus, the construction of systems dealing with the real problems is in the center of the concerns of many research tasks. Description Logics are not fixed formalisms; they are sufficiently flexible to accept the introduction of new constructors, able to meet particular needs. In this paper, we introduced two new operators, the operator less and the operator more, who allowed us to obtain a new extension of the logic of description AL. These operators will find certainly an applicability in one of the fields quoted previously.

References Références Referencias

1. R.J. Brachman, J.G.Schmolze : An overview of the KL-ONE knowledge representation system Cognitive Science ,9, 171-216(1985).
2. R.J. Brachman, al : Living with Classic : When and how to use a KL-ONE like language PIn J. F. SOWA, Ed., Principles of semantic networks, chapter 14, 401-456 (1991).
3. Donini, al : Reasoning in description logics. In G. Brewka, editor, Principles of Knowledge Representation, CSLI Publications, Stanford (CA), USA, 191-236 (1996).
4. E. Franconi: Description Logics (http ://www.cs.man.ac.uk/òfranconi).
5. A. Napoli: Une brève introduction aux logiques de descriptions Rapport de Recherche RR-3314, INRIA, (1997).
6. D. Nardi, R.J.Brachman: An Introduction to Description Logics In the Description Logic Handbook, edited by F. Baader, D. Calvanese, D.L. McGuinness, D. Nardi, P.F. Patel-Schneider, Cambridge University Press, 5-44 (2002).
7. W.A Woods, J.G. Schmolze: The KL-ONE family Computers, Mathematics with Applications, 23(25) :133-177, (1992).

This page is intentionally left blank

Global Journal of Computer Science and Technology
Software & Data Engineering
Volume 12 Issue 11 Version 1.0 Year 2012
Type: Double Blind Peer Reviewed International Research Journal
Publisher: Global Journals Inc. (USA)
Online ISSN: 0975-4172 & Print ISSN: 0975-4350

Text Categorization and Machine Learning Methods: Current State of the Art

By Durga Bhavani Dasari & Dr . Venu Gopala Rao. K
G. Narayanamma Institute of Technology and Science, Hyderabad, AP, India

Abstract - In this informative age, we find many documents are available in digital forms which need classification of the text. For solving this major problem present researchers focused on machine learning techniques: a general inductive process automatically builds a classifier by learning, from a set of pre classified documents, the characteristics of the categories. The main benefit of the present approach is consisting in the manual definition of a classifier by domain experts where effectiveness, less use of expert work and straightforward portability to different domains are possible. The paper examines the main approaches to text categorization comparing the machine learning paradigm and present state of the art. Various issues pertaining to three different text similarity problems, namely, semantic, conceptual and contextual are also discussed.

Keywords : Text Mining, Text Categorization, Text Classification, Text Clustering.

GJCST-C Classification : D.2.2

TEXT CATEGORIZATION AND MACHINE LEARNING METHODS CURRENT STATE OF THE ART

Strictly as per the compliance and regulations of:

Text Categorization and Machine Learning Methods: Current State of the Art

Durga Bhavani Dasari [α] & Dr. Venu Gopala Rao. K [σ]

Abstract - In this informative age, we find many documents are available in digital forms which need classification of the text. For solving this major problem present researchers focused on machine learning techniques: a general inductive process automatically builds a classifier by learning, from a set of pre classified documents, the characteristics of the categories. The main benefit of the present approach is consisting in the manual definition of a classifier by domain experts where effectiveness, less use of expert work and straightforward portability to different domains are possible. The paper examines the main approaches to text categorization comparing the machine learning paradigm and present state of the art. Various issues pertaining to three different text similarity problems, namely, semantic, conceptual and contextual are also discussed.

Keywords : Text Mining, Text Categorization, Text Classification, Text Clustering.

I. Introduction

Text categorization, the activity of labeling natural language texts with thematic categories from a set arranged in advance has accumulated an important status in the information systems field, due to because of augmentation of availability of documents in digital form and the confirms need to access them in easy ways.. Currently text categorization is applied in many contexts, ranging from document indexing depending on a managing vocabulary, to document filtering, automated metadata creation, vagueness of word sense, population of and in general any application needs document organization or chosen and adaptive document execution. These days text categorization is a discipline at the crossroads of ML and IR, and it claims a number of characteristics with other tasks like information/ knowledge pulling from texts and text mining [39, 40]. "Text mining" is mostly used to represent all the tasks that, by analyzing large quantities of text and identifying usage patterns, try to extract probably helpful (although only probably correct) information. Concentrating on the above opinion, text categorization is an illustration of text mining. Along with the main point of the paper that is (i) the automatic assignment of documents to a predetermined set of categories, (ii) the automatic reorganization of such a set of categories [41], or (iii) the automatic identification of such a set of categories and the grouping of documents under each categories [42], a task generally called text clustering, or (iv) any activity of placing text items into groups, a task that has two text categorization and text clustering as certain illustrations [43]. The agile developments of online information, text categorization become one of the key techniques for dealing and arranging text data.

Text categorization techniques are helpful in to classifying news stories, discovering intriguing information on the WWW, and to guide a user's search through hypertext. Since constructing text classifiers manually is difficult and time-taking so it is beneficial of learning classifiers through instances.

Author α : Asst. professor, Sri Indu College of Engineering and Technology, Hyderabad A. P. , India.
E-mail : bhavaani. dasari@gmail. Com
Author σ : Professor, G. Narayanamma Institute of Technology and Science, Hyderabad, AP, India. E-mail : kvgrao1234@gmil. Com

II. Text Categorization

The main aim of text categorization is the classification of documents into a fixed number of predetermined categories. Every document will be either in multiple, or single, or no category at all. Utilizing machine learning, the main purpose is to learn classifiers through instances which perform the category assignments automatically. This is a monitored learning problem. Avoiding the overlapping of categories every category is considered as a isolated binary classification problem.

Coming to the process the first step in text categorization is to transform documents, which typically are strings of characters, into a representation opt for the learning algorithm and the classification task. The research in information retrieval advices that word stems performs like representation units where their ordering in a document is not a major for many tasks which leads to an attribute value representation of text. Every distinct word has a feature, with the number of times word occurs in the document as its value. For eliminating dispensable feature vectors, words are taken as features only if they occur in the training data at least 3 times and if they are not "stop-words" (like "and", "or", etc.).

The representation scheme giuides to very high-dimensional feature spaces consisting of more than 10000 dimensions. Many have recognized that the need for feature collection and choice is to make the use of conventional learning methods possible, to develop generalization accuracy, and to avoid "over fitting". The recommendation of [11], the information accumulated

Global Journal of Computer Science and Technology (C) Volume XII Issue XI Version I Year 2012

criterion are used in the paper to choose a subset of features.

Subsequently, from IR it is clear that scaling the dimensions of the feature vector with their inverse document frequency (IDF) [8] develops performance. At present the "tfc" variant is used. To abstract from different document lengths, each document feature vector is reduced to unit length.

III. Taxonomy of Text Classification Process

Sebastiani discussed a wonderful review of text classification domain [25]. Hence, in the present work along with the brief description of the text classification a few recent works than those in Sebastiani's article including few articles which are not mentioned by Sebastiani are also discussed. In Figure 1 the graphical representation of the Text Classification process is shown.

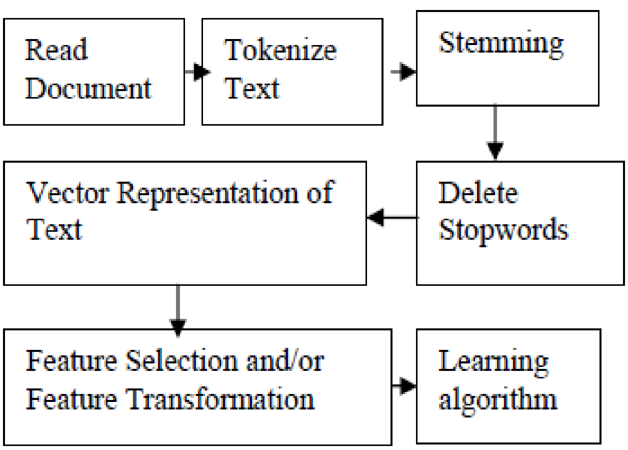


Fig. 1: Taxonomy of the Text Classification Process

The task of building a classifier for documents does not vary from other tasks of Machine Learning. The main point is the representation of a document [16]. One special certainty of the text categorization problem is that the number of features (unique words or phrases) reaches orders of tens of thousands flexibly. This develops big hindrances in applying many sophisticated learning algorithms to the text categorization, so dimension reduction methods are used which can be used either in choosing a subset of the original features [3], or transforming the features into new ones, that is, adding new features 10]. We checked the two in turn in Section 3 and Section 4. Upon completion of former phases a Machine Learning algorithm can be applied. Some algorithms have been proven to perform better in Text Classification tasks is often used as Support Vector Machines. In the present section a brief description of recent modification of learning algorithms in order to be applied in Text Classification is explained. Most of the methods that are using to examine the performance of a machine learning algorithms in Text Classification are expatiated in next section.

a) Tokenization

The process of breaking a stream of text up into tokens that is words, phrases, symbols, or other meaningful elements is called Tokenization where the list of tokens is input to the next processing of text classification.

Generally, tokenization occurs at the word level. Nevertheless, it is not easy to define the meaning of the "word". Where a tokenize process responds on simple heuristics, for instance:

All contiguous strings of alphabetic characters are part of one token; similarly with numbers. Tokens are divided by whitespace characters, like a space or line break, or by punctuation characters. Punctuation and whitespace may or may not be added in the resulting list of tokens. In languages like English (and most programming languages) words are separated by whitespace, this approach is straightforward. Still, tokenization is tough for languages with no word boundaries like Chinese. [1] Simple whitespace-delimited tokenization also shows toughness in word collocations like New York which must be considered as single token. Some ways to mention this problem are by improving more complex heuristics, querying a table of common collocations, or fitting the tokens to a language model that identifies collocations in a next processing.

b) Stemming

In linguistic morphology and information collection, stemming is the process for decreasing deviated (or sometimes derived) words to their stem, original form. The stem need not be identical to the morphological root of the word; it is usually enough if it is concern words map of similar stem, even if this stem is not a valid root. In computer science algorithms for stemming have been studied since 1968. Many search engines consider words with the similar stem as synonyms as a kind of query broadening, a process called conflation.

c) Stop word removal

Typically in computing, stop words are filtered out prior to the processing of natural language data (text) which is managed by man but not a machine. A prepared list of stop words do not exist which can be used by every tool. Though any stop word list is used by any tool in order to support the phrase search the list is ignored.

Any group of words can be selected as the stop words for a particular cause. For a few search machines, these is a list of common words, short function words, like the, is, at, which and on that create problems in performing text mining phrases that consist them. Therefore it is needed to eliminate stop words

Global Journal of Computer Science and Technology (C) Volume XII Issue XI Version I Year 2012

contains lexical words, like "want" from phrases to raise performance.

d) Vector representation of the documents

Vector denotation of the documents is an algebraic model for symbolizing text documents (and any objects, in general) as vectors of identifiers, like, for example, index terms which will be utilized in information filtering, information retrieval, indexing and relevancy rankings where its primary use is in the SMART Information Retrieval System.

A sequence of words is called a document [16]. Thus every document is generally denoted by an array of words. The group of all the words of a training group is called vocabulary, or feature set. Thus a document can be produced by a binary vector, assigning the value 1 if the document includes the feature-word or 0 if there is no word in the document.

e) Feature Selection and Transformation

The main objective of feature-selection methods is to decrease of the dimensionality of the dataset by eliminating features that are not related for the classification [6]. The transformation procedure is explained for presenting a number of benefits, involving tiny dataset size, tiny computational needs for the text categorization algorithms (especially those that do not scale well with the feature set size) and comfortable shrinking of the search space. The goal is to reduce the curse of dimensionality to yield developed classification perfection. The other advantage of feature selection is its quality to decrease over fitting, i. e. the phenomenon by which a classifier is tuned also to the contingent characteristics of the training data rather than the constitutive characteristics of the categories, and therefore, to augment generalization.

Feature Transformation differs considerably from Feature Selection approaches, but like them its aim is to decrease the feature set volume [10]. The approach does not weight terms in order to neglect the lower weighted but compacts the vocabulary based on feature concurrencies.

IV. Assortment of Machine Learning Algorithms For Text Classification

After feature opting and transformation the documents can be flexibly denoted in a form that can be utilized by a ML algorithm. Most of the text classifiers adduced in the literature utilizing machine learning techniques, probabilistic models, etc. They regularly vary in the approach taken are decision trees, naïve-Bayes, rule induction, neural networks, nearest neighbors, and lately, support vector machines. Though most of the approaches adduced, automated text classification is however a major area of research first due to the effectiveness of present automated text classifiers is not errorless and nevertheless require development.

Naive Bayes is regularly utilized in text classification applications and experiments due to its easy and effectiveness [14]. Nevertheless, its performance is reduced due to it does not model text. Schneider addressed the problems and display that they can be resolved by a few plain corrections [24]. Klopotek and Woch presented results of empirical evaluation of a Bayesian multinet classifier depending on a novel method of learning very large tree-like Bayesian networks [15]. The study advices that tree-like Bayesian networks are able to deal a text classification task in one hundred thousand variables with sufficient speed and accuracy.

When Support vector machines (SVM), are applied to text classification supplying excellent precision, but less recollection. Customizing SVMs means to develop recollect which helps in adjusting the origin associated with an SVM. Shanahan and Roma explained an automatic process for adjusting the thresholds of generic SVM [26] for improved results. Johnson et al. explained a fast decision tree construction algorithm that receives benefits of the sparse text data, and a rule simplification method that translates the decision tree into a logically equivalent rule set [9].

Lim introduced a method which raises performance of kNN based text classification by utilizing calculated parameters [18]. Some variants of the kNN method with various decision functions, k values, and feature sets are also introduced and evaluated to discover enough parameters.

For immediate document classification, Corner classification (CC) network, feed forward neural network is used. A training algorithm, TextCC is introduced in [34]. The complexity of of text classification tasks generally varies. As the number of different classes augments as of complexity and hence the training set size is required. In multi-class text classification task, unavoidable some classes are a bit harder than others to classify. Reasons for this are: very few positive training examples for the class, and lack of good forecasting features for that class.

When training a binary classifier per category in text categorization, we use all the documents in the training corpus that has the category as related training data and all the documents in the training corpus that are of the other categories are non related training data. It is a regular case that there is an overwhelming number of non related training documents specially when there is high number of categories with every allotted to a tiny documents, which is an “imbalanced data problem". This problem gives a certain risk to classification algorithms, which can accomplish perfection by simply classifying every example as negative. To resolve this problem, cost sensitive learning is required [5].

Global Journal of Computer Science and Technology (C) Volume XII Issue XI Version I Year 2012

A scalability analysis of a number of classifiers in text categorization is shown in [32]. Vinciarelli introduces categorization experiments performed over noisy texts [31]. With this noisy that any text got through an extraction process (affected by errors) from media other than digital texts (e.g. transcriptions of speech recordings extracted with a recognition system). The performance of the categorization system over the clean and noisy (Word Error Rate between ~10 and ~50 percent) versions of the similar documents is compared. The noisy texts are got through Handwriting Recognition and simulation of Optical Character Recognition where the results show less performance which is agreeable.
Other authors [36] also presented to parallelize and distribute the process of text classification. With such a procedure, the performance of classifiers can be developed in two ways that is accuracy and time complexity.

Of late in the area of Machine Learning the concept of combining classifiers is introduced as a new path for the development of the performance of single classifiers. Numerous methods advised for the creation of ensemble of classifiers. Mechanisms utilized to construct ensemble of classifiers consists of three issues. They are 1) Using various subset of training data with a one learning method, ii) Using various training parameters with a one training method (e. g. using different initial weights for each neural network in an ensemble), iii) Using various learning methods. In the context of combining multiple classifiers for text categorization, a number of researchers said that combination of various classifiers develops classification perfection [1], [29].

Comparison between the best individual classifier and the combined method, it is find that the performance of the combined method is greater [2]. Nardiello et al. [21] also presented algorithms in the family of "boosting"-based learners for automated text classification with good results.

V. Current State of the Art

Frunza, O et al[44] applied machine learning based text categorization for disease treatment relations titled "**A Machine Learning Approach for Identifying Disease-Treatment Relations in Short Texts**". With the reference of their proposal the authors debated that The Machine Learning (ML) field has won place in almost any domain of research and of lately become a reliable tool in the medical field. The empirical domain of automatic learning is used in tasks like medical decision support, medical imaging, protein-protein interaction, extraction of medical knowledge, and for total patient management care. ML is pursued as a tool by which computer-based systems can be combined with healthcare field in order to get a better, more efficient medical care.

The two tasks that are undertaken in presented model [44] supplied the basis for the design of an information technology framework has capacity to find and separate healthcare information. The first task made to find and extracts informative sentences on diseases and treatments topics, while the second one prepared to perform a finer grained classification of these sentences according to the semantic relations that presents between diseases and treatments.

The task of sentence selection discovers sentences from Medline published abstracts that talk about diseases and treatments. The task is sameto a scan of sentences contained in the abstract of an article in order to present to the user-only sentences that are found as including related information (disease-treatment information).

The task of relation identification has a deeper semantic dimension and it emphasized on finding disease-treatment relations in the sentences already choosen as being informative (e. g., task 1 is applied first). The training set is utilized to train the ML algorithm and the test set to test its performance.

Separately from the work of Rosario and Hearst [49], introduces [44] the annotations of the data set are utilizes to generate a hard task (task 1). It finds informative sentences that include information about diseases and treatments and semantic relations, versus non informative sentences. This permits to observe the excellence NLP and ML techniques can mingle with the task of discovering informative sentences, or in other words, they can remove out sentences that are related to medical diseases and treatments.

In this present model [44] the authors pointed on a few relations of interest and tried to find how the predictive model and representation technique work out good results. The task of discovering the semantic relations is as follows: Three models are constructed. Every model is focused on one relation and can distinguish sentences that contain the relation from sentences that do not. This setting is similar to a two-class classification task in which instances are labeled either with the relation in question *(Positive* label) or with non relevant information *(Negative* label);One model is built, to differentiate the three relations in a three-class classification task so that every sentence is named with one of the semantic relations. Utilizing the pipeline of tasks, we avoid some faults that can be proposed because of the truth that is considered uninformative sentences as potential data during classifying sentences directly into semantic relations. It is believed that this is a solution for discovering and separating related information made to a special semantic relation due to the second task is endeavoring to a finer grained classification of the sentences that already include information about the relations of interest.

Observation: Probabilistic models are standard and reliable for tasks performed on short texts in the medical domain. It is find potential developments in results when more information is brought in the representation technique for the task of classifying short medical texts. The second task that mentioned can be seen as a task that could get advantage from solving the first task first. Also, to perform a triage of the sentences (task 1) for a relation classification task is paramount step. Probabilistic models mixed with a representation technique bring the best results. This work seems to be quite effective text classification using machine learning to extract the relations semantically between the treatments. And it is quite clear that the model is not considering the context and conceptual issues to derive the relations between treatment relations.

For the preparation of text classifiers a new methodology which combines the distribution clustering of words and a learning technique was proposed by Al-Mubaid et al [45]. Al-Mubaid et al [50] opines that task of categorization becomes difficult if the content of the document has high dimensionality. He proposes that, this difficulty of high dimensionality can be resolved by feature clustering which is more effective than the current technique i. E feature selection. Thus the new method utilizes distributional clustering method (IB) to classify and cluster the given documents. And Lsquare is used for training text classifiers. From the experiments on few training texts As of the results those contrasted with SVM on correct experimental situation with a little number of training articles on three benchmark data grops *WebKB, 20Newsgroup,* and *Reuters- 21578*, the projected technique accomplished comparable classification accuracy*. The new method proposed is as follows*

This new model follows a good feature clustering techniques and a learning algorithm Lsquare which is logic based. This approach depends on the methodology where the text is presented by forming different clusters from the input data set and text classifiers are developed by using the Lsquare [51].

Word Features and Feature Clustering: In the vector representation every word in the text corresponds to a feature, henceforth leading to the high dimensionality of the document. By forming the clusters alike words i.e word clustering, high dimensionality of a text is minimized. Distributional clustering of words [52], [53], [54], [55], [56] is said to be the most successful to get the word clustering for TC. Every feature is a cluster alike words. For word feature techniques [53], feature clustering is more effective and useful when compared to the feature selection.

Since big quantity of lexis is brought into a group in the word clusters the necessity for feature selection automatically gets reduced. Since large number of words is brought into a group in the word clusters the necessity for feature selection automatically gets reduced. As lexis of text is brought into a cluster whole information of the text gets carried. Where as in feature selection there is a possibility to miss any information of the text.

Distributional Clustering Using the IB Method: Lexis Clusters formed by the clustering alike words is more efficient and easier when compared to feature selection [56]. In this new proposed model the common structure of Bottleneck a new technique is utilized to form the word clusters [53]. IB method traces the fully developed pertinent coding or the compact version of one variable X, given the joint distribution of two random variables P(X, Y), while the mutual information about the other variable Y is saved to the extent feasible. In the technique used in [53], X denotes the input lexis and variable Y denotes the class labels. In addition, they give a hierarchical top-down clustering process for generating the distributional IB clusters [53]. Initiating with one cluster that consists all the input data, the clusters divides in iterations with incrementing the annealing parameter β.

Observation: Recent developments in the techniques of feature clustering and dimension reduction are well utilized in the proposed in new model. The proposed TC approach combines these new advancements with logic-based learning techniques. The proposed method is experimented on all training-testing settings utilizing WebKB data set and on 0NG data set. These experiments proved that TC approach is more effective than that of SVM-based system. This technique of machine learning doesn't consider the semantic, theoretical and relative relations of the texts and the new model is tested under the same parameters. This is a disadvantage of the new approach and the feature research will be done in such a way that it recognizes all the semantic, theoretical and relative relations of the texts.

Sun, A. et al [46] opines that classification techniques that are utilizing top-down approach are competent enough to deal with changes to the category trees in text mining. Though these approaches are effective one common problem in all these methods is Blocking. It means rejection of the texts by the classifiers which cannot be sent to the classifiers at lower- levels. Thus Sun, A. et al [57] projected three methods to deal with the blocking problem, namely, *Threshold Reduction, Restricted Voting,* and *Extended Multiplicative.* The tests carried out utilizing Support Vector Machine (SVM) classifiers on the Reuters collection pointed out that all three projected models elaborated beneath could decrease blocking and advance the classification accuracy.

Threshold Reduction Method (TRM): Through The threshold reduction method many a documents can be send to the classifiers at the lower level if the sub tree classifiers are kept at the lower thresholds. In regular HTC, a manuscript d_i of group c_n is blocked by the sub

Global Journal of Computer Science and Technology (C) Volume XII Issue XI Version I Year 2012

tree classifier of any predecessor sub tree c_i, s of c_n when $\tau(c_i.s | d_j, \theta_{c_i.s}) = 0$. Hence TRM model concentrated on providing the right thresholds for sub tree classifiers. To provide the right thresholds number of thresholds to be considered must be few. It can be achieved when all the sub tree classifiers at the same time utilize the same threshold value..

Restricted Voting Method (RVM): In RVM methodology, it is made possible that sub tree classifiers of a node could get the documents from another sub tree classifiers of its grandparent node. This is made possible by creating the secondary channels. In this method secondary channel associates the secondary sub tree classifier or a secondary local classifier with the grandparent node thus enabling a direct connection between a node and its grandparent. $\tau'_{c_i.s}$ Categorizes articles that are approved by the sub-tree classifier or the secondary sub-tree classifier (if it exists) connected with c_{i-2}. $\tau'_{c_i.s}$ Accepts a document *dj* if $\tau'(c_i.s | d_j, \theta'_{c_i.s}) = 1$. Correspondingly, a secondary local classifier $\tau'_{c\ell}$ is connected with each leaf node $c\ell$ and classifies articles approved by the sub-tree classifier or the secondary sub-tree classifier connected with the grandparent node. '$\tau'_{c\ell}$' accepts a document *dj* if $\tau'(c_\ell | d_j, \theta'_{c_i.s}) = 1$. In TRM the thresholds of the sub tree classifiers are similar to the thresholds of the secondary classifiers. In RVM, though the secondary sub-tree (local) classifier and the sub-tree (local) classifier associated with a node are given the same decision task, they are trained with diverse sets of training articles.

Extended Multiplicative Method (EMM): The extended multiplicative method is an extension of the multiplicative method projected by Dumais and Chen [58]. The proposed new model will be able to handle category trees with more levels, where as the source method is limited only to the 3 level category trees. Like STTD, EMM links a local classifier with each leaf node and a sub-tree classifier with each non-leaf node. Let c_n be a leaf node at level n and the parent node be c_{n-1}. An article d_i is given to c_n if $P(c_n | d_j) \times P(c_{n-1}.s | d_j) \geq \theta_{c_{n(n-1)}}$, indicates a threshold. Likewise, d_i can be taken by the sub-tree classifier connected with c_{n-1} if $P(c_{n-1}.s | d_j) \times P(c_{n-2}.s | d_j) \geq \theta_{c_{n(n-1)(n-2)}}$. Thresholds are derived akin to those in TRM. EMM, in future research can be developed to reflect on the possibilities of more than two levels [10].

Observation: The challenge of Blocking in hierarchical text classification is mainly targeted in the proposed new model. Top-down approach is used to resolve the blocking problem. To differentiate the degree of blocking, we have established blocking factor as a new kind of classifier-centric performance measure. As a solution to the blocking challenge three methods were put forward namely, threshold reduction, restricted voting, and extended multiplicative methods. Of all the techniques restricted voting model is effective in bringing down the Blocking problem and has proved to be the best in terms of F_1^M measure too. But the disadvantage of this technology is it requires more classifiers thus demanding more time for training. Though they are few advantages, all the said models are not effective in summing-up the given document. Furthermore even these new models depend on term and document frequency and are unable to consider the contextual and semantic relations of the text. Thus further research will be focused on developing a model which recognizes semantic, conceptual and contextual relations of the texts thus enabling an effective precision. Text categorization methods that are utilizing machine learning techniques to bring on manuscript classifiers face a problem with very high computational costs that sometimes rise exponentially in the number of features because of the usage of the example manuscripts those can be part of the multiple classes. As a remedy to these raising costs, Sarinnapakorn, K et al[47] proposed a "baseline induction algorithm" which will be exclusively used for sub sets of features, where a set of classifiers are united. Along with the above said solutions Sarinnapakorn, K et al[47] proposed one more technique i. e alternative fusion techniques for the classifies that send back both class labels and confidences in these labels. This technique is developed from the Dempster-Shafer Theory.

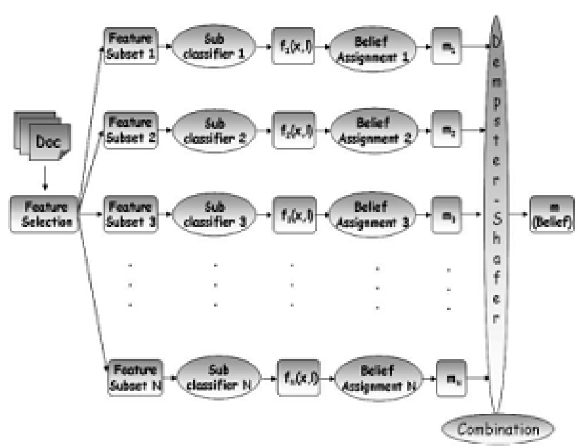


Fig. 2: We study a classification system where a simple mechanism based on the DST fuses the "testimories" of several subclassifiers that have been obtained by running a BIA on different feature subsets

Sarinnapakorn, K et al [47] examined a methodology that unites the outcome of a set of sub classifiers which are stimulated by a BIA every time from the same training examples depicted by a different feature subset. Each feature symbolizes the frequency of lexis.

Text categorization architecture is explained with picture in Fig 2. Whenever the example x is classified, the ranking function f_i(x; l) is given as an output by the ith classifier. Perfect real class labels of an example can be achieved by a methodology which can unite these out puts in such a way that it brings out a set Y ⊂ Y.

Every run of BIA stimulates a sub classifier that, for article x and class label l, returns f(x; l) ∈ (- ∞, ∞) that measures the sub classifier's confidence in l (higher f(x; l) designates higher confidence). A fusion methodology is required to unite these suggestions and confidence values. The instruction standardizes the function f(x;1) so as to ensure that its values commence in between the range of [0, 1]. If suppose range 1 is considered, the alteration between f(x;1) and the least belief of the classifier of any random label is elucidated, the resulting solution is then partitioned based on the high count obtained in the outputs of the sub-classifier. This is particularly done to ensure that the changed values can be considered as degrees of confidence, where values nearing 1 replicate their confidence in 1 while values nearing 0 replicate their robust incredulity in 1.

Step 2 utilizes the changed confidence values in the estimations of the BBAs that are closely related to the class labels. Refer the appendix for valid evidence that masses just estimated fulfill the requirements in (1). The Dempster-Shafer rule of arrangement is to blend the mass values restored by the various sub classifiers for all the four specified opportunities mentioned in every available class label.

Observation: Sarinnapakorn, K et al [47] designated a methodology to tackle forbidden computational charges of text-classification schemes wherein every individual file fits in the multiple classes at that point of time. The designated model specifically deals with the orientation mechanisms, whose training period increases in a linear fashion in accordance with the multiple features that are utilized for depicting critical hurdles in the case of text files. The feature called observation that the sub classifier amalgamation results in typical bursting of specialized computational reduction, exploiting the fact that the performance that was accomplished earlier can be still enhanced. The enhancement may probably occur if the chosen characteristic-selection mechanism utilizes provoked sub classifiers who harmonize amicably. The chosen box was a black one and hence the exact featured option of the BIA was not considered seriously.

Bell, D. A. et al [48] claims those results prove otherwise stating various text differentiation methodologies present various results. He also prescribed a methodology for merging the classifiers. Various techniques like support vector machine (SVM). Nearest fellow neighbors (kNN) and Rocchio were researched upon to unite the effects of two or more various categorization techniques in accordance with a sequential line of attack. A more refined version of the tactic to be employed is explained as follows:

Utilization of various confirmation techniques employs merging mechanisms like Dempster's rule or or the orthogonal sum [14] to resolve the Data Information Knowledge fusion issue. A more conventional way to substantial motive of explanation depends on the concept of statistical methodologies to present indicative assurance ie. The Dempster-Shafer (D-S) hypothesis that utilizes the quantitative data extracted from the classifiers.

Evidence Theory: The D-S hypothesis is an efficient technique realized for surviving the tentative expressions implanted in the confirmatory issues that are precariously used in the reasoning methods and it best ensembles with conclusion-based actions. This hypothesis is often considered as a simplification of Bayesian probability hypothesis by assisting in issuing a rational presentation for lack of evidence as also by abandoning the irrelevant and inadequate reasoning standards. A reasoning technique is devised as bits of evidence and specialize them to a stern formal mechanism so as to draw assumptions from a undisclosed evidence where it is expressed in the form of evidential functions. Few functions that are used frequently are mass functions, belief functions, doubt functions and plausibility functions. All these functions express the same data as the others.

Categorization-Specific Mass Function: The designated model contemplates the issue of calculating degrees of principle for the proof deduced from the text classifiers and the varied exact delineations of mass and belief terms for this specific field and then blend number of pieces of proofs to arrive at a conventional decision.

The 2-Points Focused Combination Method: Suppose that there exists a set of training data and a set of algorithms, where every individual algorithm produces one or more classifiers depending on the selected training set of data and then merge various outputs of various classifiers depending on the same testing files using Dempster's rule of merging to prepare the ultimate classification verdict.

Observation: Bell, D. A. et al [48] proposes a unique mechanism for presenting outputs obtained from various classifiers. A focal element triplet can be converted to a focal element quarter by expanding it. A consequential methodology implemented for a number of classifiers depending on the new structure was scrutinized as also modus operandi used for calculating

Global Journal of Computer Science and Technology (C) Volume XII Issue XI Version I Year 2012

triplets and quartets can be gained by evaluating the modus operandi implemented to gain values of other focal elements. The organization and related techniques and mechanisms invented in this experiment yield practical results for data evaluation and is quite unique to formulate. The designated model stipulates the responsibility of text content relational features like contextual and conceptual to incorporate results from various classifiers.

VI. Conclusion

This paper focuses on investigating the utilization of Machine learning mechanisms for ascertaining text classifiers and tries to generalize the specific properties of the recent trends in learning techniques with text data and recognize whether any of the stipulated models cited recently in current literature are judged as text analogous in terms of semantic, conceptual and contextual format. It is apparent from the statistics obtained that least count of models has been insinuated in recent times, focusing largely on reducing the computational density of the machine learning forms to enhance competence. Concerning recent literature, no recent work has been devised to focus on managing coherency of the files already classified.

References Références Referencias

1. Bao Y. and Ishii N., "Combining Multiple kNN Classifiers for Text Categorization by Reducts", LNCS 2534, 2002, pp. 340-347
2. Bi Y., Bell D., Wang H. , Guo G. , Greer K. , "Combining Multiple Classifiers Using Dempster's Rule of Combination for Text Categorization", MDAI, 2004, 127-138.
3. Brank J., Grobelnik M., Milic-Frayling N., Mladenic D., "Interaction of Feature Selection Methods and Linear Classification Models", Proc. of the 19th International Conference on Machine Learning, Australia, 2002.
4. Ana Cardoso-Cachopo, Arlindo L. Oliveira, An Empirical Comparison of Text Categorization Methods, Lecture Notes in Computer Science, Volume 2857, Jan 2003, Pages 183 - 196
5. Chawla, N. V. , Bowyer, K. W. , Hall, L. O. , Kegelmeyer, W. P. , "SMOTE: Synthetic Minority Over-sampling Technique, " Journal of AI Research, 16 2002, pp. 321-357.
6. Forman, G., An Experimental Study of Feature Selection Metrics for Text Categorization. Journal of Machine Learning Research, 3 2003, pp. 1289-1305
7. Fragoudis D., Meretakis D. , Likothanassis S., "Integrating Feature and Instance Selection for Text Classification", SIGKDD '02, July 23-26, 2002, Edmonton, Alberta, Canada.
8. Guan J., Zhou S., "Pruning Training Corpus to Speedup Text Classification", DEXA 2002, pp. 831-840
9. D. E. Johnson, F. J. Oles, T. Zhang, T. Goetz, "A decision-tree-based symbolic rule induction system for text categorization", IBM Systems Journal, September 2002.
10. Han X. , Zu G. , Ohyama W. , Wakabayashi T. , Kimura F. , Accuracy Improvement of Automatic Text Classification Based on Feature Transformation and Multi-classifier Combination, LNCS, Volume 3309, Jan 2004, pp. 463-468
11. Ke H., Shaoping M., "Text categorization based on Concept indexing and principal component analysis", Proc. TENCON 2002 Conference on Computers, Communications, Control and Power Engineering, 2002, pp. 51- 56.
12. Kehagias A. , Petridis V. , Kaburlasos V. , Fragkou P. , "A Comparison of Word- and Sense-Based Text Categorization Using Several Classification Algorithms", JIIS, Volume 21, Issue 3, 2003, pp. 227-247.
13. B. Kessler, G. Nunberg, and H. Schutze. Automatic detection of text genre. In Proceedings of the Thirty-Fifth ACL and EACL, pages 32–38, 1997.
14. Kim S. B. , Rim H. C. , Yook D. S. and Lim H. S. , "Effective Methods for Improving Naive Bayes Text Classifiers", LNAI 2417, 2002, pp. 414-423
15. Klopotek M. and Woch M., "Very Large Bayesian Networks in Text Classification", ICCS 2003, LNCS 2657, 2003, pp. 397-406
16. Leopold, Edda & Kindermann, Jörg, "Text Categorization with Support Vector Machines. How to Represent Texts in Input Space?", Machine Learning 46, 2002, pp. 423 - 444.
17. Lewis D., Yang Y., Rose T., Li F., "RCV1: A New Benchmark Collection for Text Categorization Research", Journal of Machine Learning Research 5, 2004, pp. 361-397.
18. Heui Lim, Improving kNN Based Text Classification with Well Estimated Parameters, LNCS, Vol. 3316, Oct 2004, Pages 516 - 523.
19. Madsen R. E., Sigurdsson S. , Hansen L. K. and Lansen J., "Pruning the Vocabulary for Better Context Recognition", 7th International Conference on Pattern Recognition, 2004
20. Montanes E., Quevedo J. R. and Diaz I., "A Wrapper Approach with Support Vector Machines for Text Categorization", LNCS 2686, 2003, pp. 230-237
21. Nardiello P., Sebastiani F., Sperduti A., "Discretizing Continuous Attributes in AdaBoost for Text Categorization", LNCS, Volume 2633, Jan 2003, pp. 320-334
22. Novovicova J., Malik A., and Pudil P., "Feature Selection Using Improved Mutual Information for Text Classification", SSPR&SPR 2004, LNCS 3138, pp. 1010– 1017, 2004

23. Qiang W., XiaoLong W., Yi G., "A Study of Semi-discrete Matrix Decomposition for LSI in Automated Text Categorization", LNCS, Volume 3248, Jan 2005, pp. 606-615.
24. Schneider, K., Techniques for Improving the Performance of Naive Bayes for Text Classification, LNCS, Vol. 3406, 2005, 682- 693.
25. Sebastiani F., "Machine Learning in Automated Text Categorization", ACM Computing Surveys, vol. 34 (1), 2002, pp. 1-47.
26. Shanahan J. and Roma N., Improving SVM Text Classification Performance through Threshold Adjustment, LNAI 2837, 2003, 361- 372
27. Soucy P. and Mineau G. , "Feature Selection Strategies for Text Categorization", AI 2003, LNAI 2671, 2003, pp. 505-509
28. Sousa P., Pimentao J. P. , Santos B. R. and Moura-Pires F., "Feature Selection Algorithms to Improve Documents Classification Performance", LNAI 2663, 2003, pp. 288-296
29. Sung-Bae Cho, Jee-Haeng Lee, Learning Neural Network Ensemble for Practical Text Classification, Lecture Notes in Computer Science, Volume 2690, Aug 2003, Pages 1032 – 1036.
30. Torkkola K., "Discriminative Features for Text Document Classification", Proc. International Conference on Pattern Recognition, Canada, 2002.
31. Vinciarelli A., "Noisy Text Categorization, Pattern Recognition", 17th International Conference on (ICPR'04) , 2004, pp. 554-557
32. Y. Yang, J. Zhang and B. Kisiel., "A scalability analysis of classifiers in text categorization", ACM SIGIR'03, 2003, pp 96- 103
33. Y. Yang. An evaluation of statistical approaches to text categorization. Journal of Information Retrieval, 1(1/2):67–88, 1999.
34. Zhenya Zhang, Shuguang Zhang, Enhong Chen, Xufa Wang, Hongmei Cheng, TextCC: New Feed Forward Neural Network for Classifying Documents Instantly, Lecture Notes in Computer Science, Volume 3497, Jan 2005, Pages 232 – 237.
35. Shuigeng Zhou, Jihong Guan, Evaluation and Construction of Training Corpuses for Text Classification: A Preliminary Study, Lecture Notes in Computer Science, Volume 2553, Jan 2002, Page 97-108.
36. Verayuth Lertnattee, Thanaruk Theeramunkong, Parallel Text Categorization for Multi-dimensional Data, Lecture Notes in Computer Science, Volume 3320, Jan 2004, Pages 38 - 41
37. Wang Qiang, Wang XiaoLong, Guan Yi, A Study of Semi-discrete Matrix Decomposition for LSI in Automated Text Categorization, Lecture Notes in Computer Science, Volume 3248, Jan 2005, Pages 606 – 615.
38. Zu G., Ohyama W., Wakabayashi T., Kimura F., "Accuracy improvement of automatic text classification based on feature transformation": Proc: the 2003 ACM Symposium on Document Engineering, November 20-22, 2003, pp. 118-120
39. KNIGHT, K. 1999. Mining online text. Commun. ACM 42, 11, 58–61.
40. PAZIENZA, M. T., ed. 1997. Information Extraction. Lecture Notes in Computer Science, Vol. 1299. Springer, Heidelberg, Germany. RILOFF. E. 1995. Little words can make a big difference for text classification. In Proceedings of SIGIR-95, 18th ACM International Conference on Research and Development in Information Retrieval (Seattle, WA, 1995), 130–136.
41. BORKO, H. AND BERNICK, M. 1963. Automatic document classification. J. Assoc. Comput. Mach. 10, 2, 151–161.
42. MERKL, D. 1998. Text classification with selforganizing maps: Some lessons learned. Neurocomputing 21, 1/3, 61–77.
43. MANNING, C. AND SCH¨UTZE, H. 1999. Foundations of Statistical Natural Language Processing. MIT Press, Cambridge, MA.
44. Frunza, O. ; Inkpen, D. ; Tran, T. ; , "A Machine Learning Approach for Identifying Disease-Treatment Relations in Short Texts, " Knowledge and Data Engineering, IEEE Transactions on , vol. 23, no. 6, pp. 801-814, June 2011, doi: 10. 1109/TKDE. 2010. 152, URL: http://ieeexplore. ieee. org/stamp/stamp.jsp?tp=&arnumber=5560656&isnumber=5753264
45. Al-Mubaid, H.; Umair, S. A. ;, "A New Text Categorization Technique Using Distributional Clustering and Learning Logic, " Knowledge and Data Engineering, IEEE Transactions on , vol. 18, no. 9, pp. 1156-1165, Sept. 2006, doi: 10. 1109/TKDE. 2006. 135.
46. Sun, A.; Lim, E. -P.; Ng, W. -K.; Srivastava, J.;, "Blocking reduction strategies in hierarchical text classification, " Knowledge and Data Engineering, IEEE Transactions on, vol. 16, no. 10, pp. 1305-1308, Oct. 2004, doi: 10. 1109/TKDE. 2004. 50.
47. Sarinnapakorn, K.; Kubat, M.;, "Combining Subclassifiers in Text Categorization: A DST-Based Solution and a Case Study, " Knowledge and Data Engineering, IEEE Transactions on , vol. 19, no. 12, pp. 1638-1651, Dec. 2007, doi: 10. 1109/TKDE. 2007. 190663
48. Bell, D. A. ; Guan, J. W. ; Bi, Y. ; , "On combining classifier mass functions for text categorization, " Knowledge and Data Engineering, IEEE Transactions on , vol. 17, no. 10, pp. 1307- 1319, Oct. 2005, doi: 10. 1109/TKDE. 2005. 167
49. P. Srinivasan and T. Rindflesch, "Exploring Text Mining from Medline, " Proc. Am. Medical Informatics Assoc. (AMIA) Symp. , 2002
50. H. Al-Mubaid and K. Truemper, "Learning to Find Context-Based Spelling Errors, " Data Mining and

Knowledge Discovery Approaches Based on Rule Induction Techniques, 2006
51. G. Felici and K. Truemper, "A Minsat Approach for Learning in Logic Domains, " Informs J. Computing, vol. 14, no. 1, winter 2002.
52. L. D. Baker and A. K. McCallum, "Distributional Clustering of Words for Text Classification, " Proc. Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval, 1998
53. R. Bekkerman, R. El-Yaniv, N. Tishby, and Y. Winter, "Distribu¬tional Word Clusters vs Words for Text Categorization, " J. Machine Learning Research, vol. 3, 2003
54. I. Dhillon, S. Mallela, and R. Kumar, "A Divisive Information- Theoretic Feature Clustering Algorithm for Text Classification, "J. Machine Learning Research, vol. 3, 2003
55. F. Pereira, N. Tishby, and L. Lee, "Distributional Clustering of English Words, " Proc. 31st Ann. Meeting of the ACL, pp. 183-190, 1993
56. N. Slonim and N. Tishby, "The Power of Word Clusters for Text Classification, "Proc. 23rd European Colloquium on Information Retrieval Research (ECIR-01), 2001
57. S. T. Dumais and H. Chen, "Hierarchical Classification of Web Content, " Proc. ACM SIGIR '00, pp. 256-263, July 2000.
58. A. Sun and E. -P. Lim, "Hierarchical Text Classification and Evaluation, " Proc. IEEE Int'l Conf. Data Mining (ICDM '01), pp. 521-528, Nov. 2001

Global Journal of Computer Science and Technology (C) Volume XII Issue XI Version I Year 2012

Global Journal of Computer Science and Technology
Software & Data Engineering
Volume 12 Issue 11 Version 1.0 Year 2012
Type: Double Blind Peer Reviewed International Research Journal
Publisher: Global Journals Inc. (USA)
Online ISSN: 0975-4172 & Print ISSN: 0975-4350

A Naive Based Approach for Mapping Two ADL Models

By Sai Bharath Kadati, K. K. Baseer, A. Rama Mohan Reddy & CH. Gowthami
JNIAS/JNTUA, University, Hyderabad. SVEC, Tirupati, and A.P., India

Abstract - In software engineering, we have identified and described the model correspondence problem. To Describe system architecture and artifacts uses models and diagrams. Models contains series of versions. To understand how versions correspondence are difficult. So, we designed a framework based on Search and Ammolite algorithms, which can cardinally finds the correspondence software models. Models are represented as graphs whose nodes have attributes (name, edge, label, connections). For a given diagram pair, it performs different individual matches such as pair-wise match, Split-Merge Match and Drop match and then combine all matches together to design a ADL model. Every ADL Model has its correspondence score for rating quality candidates. To find best Correspondence among the given ADL models uses Search and Ammolite Algorithms.

Keywords : Decision, Design Artifacts, Elements, Reasoning Principles, Semantic Information, Syntactic Information, Visual Information.

GJCST-C Classification : D.2.11

A NAIVE BASED APPROACH FOR MAPPING TWO ADL MODELS

Strictly as per the compliance and regulations of:

A Naive Based Approach for Mapping Two ADL Models

Sai Bharath Kadati [α], K. K. Baseer [σ], A. Rama Mohan Reddy [ρ] & CH. Gowthami [ω]

Abstract - In software engineering, we have identified and described the model correspondence problem. To Describe system architecture and artifacts uses models and diagrams. Models contains series of versions. To understand how versions correspondence are difficult. So, we designed a framework based on Search and Ammolite algorithms, which can cardinally finds the correspondence software models. Models are represented as graphs whose nodes have attributes (name, edge, label, connections). For a given diagram pair, it performs different individual matches such as pair-wise match, Split-Merge Match and Drop match and then combine all matches together to design a ADL model. Every ADL Model has its correspondence score for rating quality candidates. To find best Correspondence among the given ADL models uses Search and Ammolite Algorithms.

Keywords : Decision, Design Artifacts, Elements, Reasoning Principles, Semantic Information, Syntactic Information, Visual Information.

I. Introduction

An Architecture is defined as building for humans, and being an architect is having the spirit to build for humans. A framework is a collection of classes and applications, libraries of SDKs and APIs to help the different components all work together. In engineering discipline an essential part of quality is control of change. That dictates the need to review and understand changes prior to accept them. Models and Diagrams are a primary design artifacts in this environment, this means being able to compare diagrams to identify correspondence and discrepancies between them. In large-scale IT system development techniques have long existed for comparing textual artifacts, somewhat less work has been reported concerning comparisons of the diagrams and model that are common. The main problem of this paper is to correspondence between **a pair of diagrams** (a mapping between elements of one diagram and elements of the other) and **introduce a Bayesian approach to solve the problem.** The application which are in the central to modern IT systems development process includes structured representation of requirements, business process workflows, system overviews, architectural specifications of systems, network topologies, object designs, state transition diagrams, and control and data flow representation of code.

a) Scenarios

The system development life cycle has several application to find correspondence between models. A series of successive **revisions** of a model from design activity. There is a need to review and understand the nature of revisions as part of accepting them, rejecting them or merging them with other concurrent revisions and to identify correspondences and discrepancies is central to such activities. Model **variants** correspond is crucial for integration. Different collaboration may experiment with different paths of evolution of a model, resulting in a number of transient variants, with the intent that those branches deemed successful will be integrated back into a main stream. The use of multiple **views** of the architecture of the system by using many development approaches and methodologies[6]. The model we propose is made up of five main views [7].

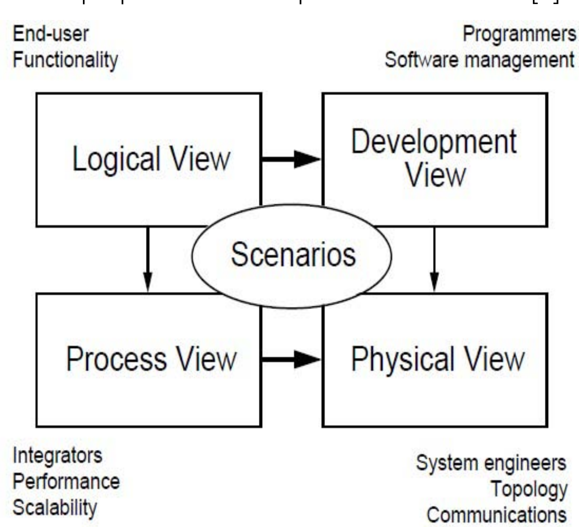


Fig. 1: The 4+1 View Model

- The ***logical* view**, which is the object model of the design (when an object-oriented design method is used),
- The ***process* view**, which captures the concurrency and synchronization aspects of the design,

Author α : Bachelor of Technology in Computer Science and Engineering and Department of Information Technology, SVEC, Tirupati, and A.P., India.
Author σ : Assistant Professor in department of Information Technology, JNIAS/JNTUA, University, Hyderabad. SVEC, Tirupati, and A.P., India.
Author ρ : Professor & HOD of Computer Science and Engineering, Sri Venkateswara University, Tirupati, A.P., India.
Author ω : B. Tech Computer Science and Engineering from JNTUK, Kakinada, Anantapur, India.

- The *physical* **view**, which describes the mapping(s) of the software onto the hardware and reflects its distributed aspect,
- The *development* **view**, which describes the static organization of the software in its development environment.

Traceability is the another important requirement for maintaining quality [10], [5]. Traceability between software artifact, such as requirements, design elements, code, test cases and defect reports. At finer level of granularity, traceability provides the ability to navigate between the elements of different artifacts such as individual software components, hardware nodes, requirements, non-functional requirements, and architectural decisions that reflects the design rationale for the system. The larger asset of reuse is the incorporation of **reference architecture** from a repository into a solution design.

b) Contribution of the Paper

In Present days, determining correspondences between models is a tedious, error-prone, time-consuming, manual process. The main goal is to achieve an automated means of determining the correspondences, similar to techniques for automated comparison of textual artifacts. This requires us to answer several questions:

- How do we represent models?
- Which features of models must be represented?
- What algorithms should be used to find correspondences?

In this paper, provide answers to these questions.

II. Diagram Features

We focus mainly on the problem of finding correspondences in the domain of IT architecture operational models [2], although the paper techniques have proven effective for other kinds of IT architecture models as well. Operational models are used by IBM Global Services architects as part of a development methodology for customized IT solutions. An operational model also includes model elements reflecting the key decisions constituting the rationale for the solution design.

The main features of an operational model diagram can be abstracted to elements found in many other kinds of diagrams:

- **Labeled nodes.** System components can be represented as textual or pictorial in a diagram. For example, an attribute may indicate whether the node is internal or external to the solution in an operational model diagram.
- **Edges.** A edge represents a relationship or association and it can indicate communication paths connection between nodes. Bandwidth, Technology, Security etc., are the attributes of the edge.
- **Containers.** A node that which contains other nodes is simply called **Container**. For example, In operational model diagram, a server may contain multiple software components or a region may contain multiple servers. Containers may be nested, current prototype only considers the nesting of servers within regions when correspondences.
- **Groups.**[8] Nodes are **grouped** together semantically. For instance, in operational models, servers located in the same building may be grouped within a common region. Like nodes, **groups** have labels and relationship. For example, regions have an adjacency relationship that indicates a connection.

Regions are discussed in greater detail below.

The information represented by system diagrams can be broadly classified into three types: 1) syntactic information (e.g., nodes, labels, containment, and edges), 2) semantic information (e.g., types, defined semantic attributes), and 3) visual information (e.g., position, shape, and color of diagram elements). Leveraging all of these kinds of information is one of the major challenges of diagram matching.

III. Model Correspondence Problem

The model correspondence problem is the problem of finding the "best" correspondence between the elements of two diagrams.

a) Semantics and Domain-Specific Knowledge as a Basis

The first issue is how to define "best." It may seem appealing to define "best" as the correspondence that preserves a specific semantic relationship between the two diagrams, but this definition would be difficult to apply in practice, for several reasons.

First, there are many possible semantic relationships between diagrams and it is hard to decide which applies. For example, in one case, we may have a diagram pair(E, E'), where E' is a revision of E, with the semantic relation "is a revision of." In another case, E may be a conceptual description of a system and D' a physical description, with the semantic relation "realizes."

Second, even if the semantic relationship is known, defining it in precise detail would be difficult, and even a precise definition may not have sufficient information to find the best correspondence.

Third, many diagrams found in practice have no formal semantics: They use informal notions of "boxes" and "lines" to convey context-specific architectural notions.

Either way, we conjecture that generic matching techniques can go a long way in finding correspondences between diagrams without having to incorporate knowledge of these kinds of semantic relationships or even knowledge of any of the deeper semantics of the various types of diagram.

b) Reasoning Principles for Recovering Traceability

Human experts can often identify good correspondences after careful examination of a pair of diagrams. Human experts did this by manually finding the best correspondences for some diagram pairs, and recording the reasoning principles used to find the correspondences

The following principles of reasoning about diagram pairs correspondences:

- Most decisions are made using ***evidence*** about which nodes from one diagram match which nodes from the other diagram.
- Every feature of the nodes in the diagrams can be important evidence, including text, connection and containment relationships, and geometric and pictorial attributes.

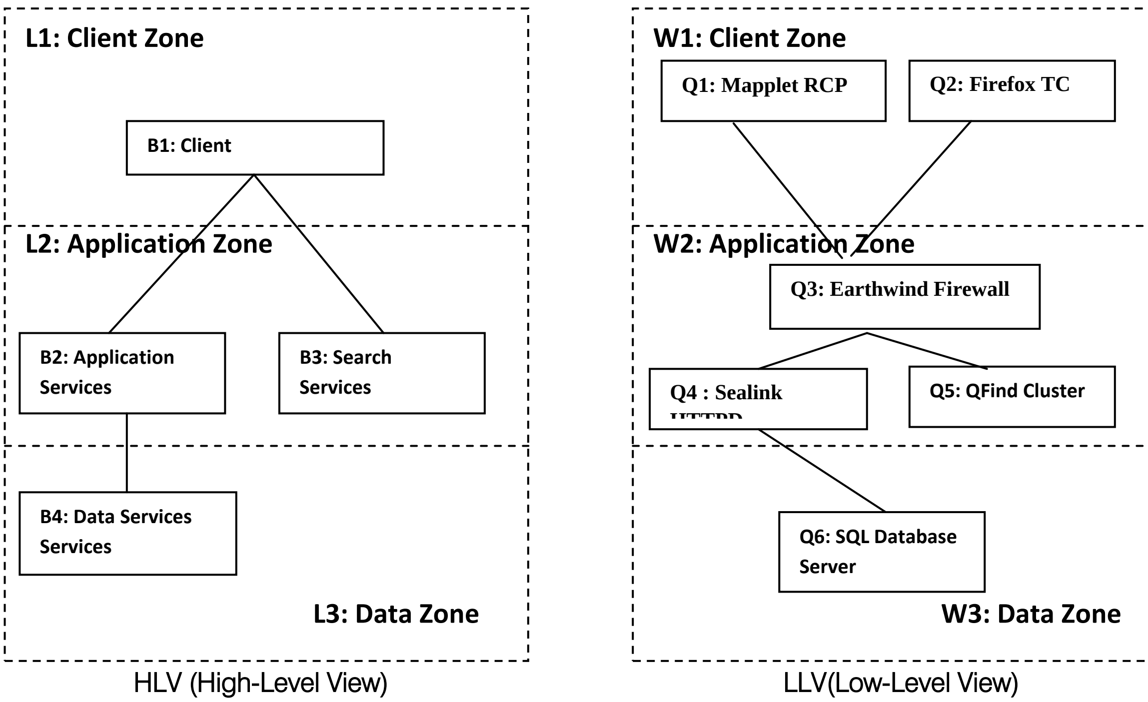


Fig. 2: Simple Example Diagram Pair

- Evidence takes the form of having similar or dissimilar features. For example, if two nodes have the same label, this is strong evidence that they match. If two nodes are at totally different positions in their respective diagrams, that is evidence that they do not match.
- For a node pair (n, n') sometimes there is some evidence that n and n' match and other evidence that n and n' do not match. Practitioners will use their experience to weigh the relative significance of the different pieces of evidence and decide whether or not n and n' match.
- The correspondence can be filled in by identifying one-to-one matches using evidence about node pairs. Other kinds of evidence help suggest non-one-to-one matches when necessary. For example, if diagram D has a node n labeled "Firewall and Access Control" and D' has node n'1 labeled "Firewall" and n'2 labeled "Access Control," the labels suggest that n matches to both n'1 and n'2. If n'1 and n'2 are both within the same container, this is further evidence that they may match to the same node in D.

IV. Solution Overview

An overview of our solution, and it serves as a road map to Bayesian correspondence, which gives the mathematical and gives a mathematical description an algorithm.

Our algorithm as Automated Matching of Models (AMMO). We explain the main ideas of the AMMO algorithm by tracing its behavior on a simple example diagram pair, HLV and LLV, as shown in Fig. 2. This diagram pair is highly simplified for presentation purposes, but it does exhibit some of the difficulties found in production models, such as non-obvious node matches and matches that are not one-to-one

The tags B1; B2; . . . ; Q1; Q2; . . .; L1; L2; . . . ; W1; W2; . . . are only for ease of reference in this discussion and are not part of the actual node labels. Also, note that regions, such as "L2: Application Zone," contain nodes, such as "B2: Application Services" and

Global Journal of Computer Science and Technology (C) Volume XII Issue XI Version I Year 2012

"B3: Search Services." Further note that the label of a node is not qualified by the label of the region that contains it.

a) Feature Similarity

Our algorithm begins by computing a number of similarity values for each possible node pair consisting of a node from one diagram and a node from the other diagram, i.e., $(x, x') \in HLV \times LLV$. A similarity value is computed for each feature from a predetermined set of features. For example, nodes with similar labels often match, so one of the features we work with is the textual label of a node, and one of the similarities we compute for a node pair is its label similarity—a value between 0 and 1 reflecting the string similarity between the node labels. A similarity value can be regarded as a "raw similarity score" for a particular feature for a node pair.

Table 1: Pair-wise Label Similarity for Fig. 2

	Q1	Q2	Q3	Q4	Q5	Q6
B1	0.118	0.125	0.083	0.211	0.316	0.095
B2	0.385	0.240	0.061	0.143	0.143	0.200
B3	0.190	0.200	0.286	0.348	0.174	0.240
B4	0.316	0.111	0.231	0.095	0.095	0.435

b) Match Probability from Feature Similarity

A similarity value in itself does not indicate whether pair of nodes match; that is, it is unclear whether a particular similarity is low or high with respect to the population. To transform a raw score consisting of a feature similarity value into a probability that a pair of nodes match. Given a probability distribution of the similarity values, based on similarities observed for matching and non-matching pairs in training data, Bayesian inference will convert the similarity of (x, x') into the probability that (x, x') match. From Table 1 to Table 2 the probabilities resulting from Bayesian inference given the similarities. One can see that the probability of node B4 matching to Q6 is much higher than the probability of B4 matching to any other node. One can also see that B2 is approximately twice as likely to match to Q1 as it is to match to any other node. Finally, one can see that the probabilities of B1 and B3 matching to any of the nodes in the second diagram are approximately equal, indicating that the label feature is inadequate in determining matches for these nodes.

Table 2 : Pairwise Match probabilities based on Label Similarity

	Q1	Q2	Q3	Q4	Q5	Q6
B1	0.100	0.100	0.104	0.108	0.155	0.102
B2	**0.225**	0.116	0.108	0.108	0.100	0.105
B3	0.104	0.105	0.135	0.182	0.102	0.116
B4	0.155	0.101	0.113	0.102	0.102	**0.308**

c) Multiple Evidencer

For some nodes, such as B1, label similarity does not help much in finding a match. In general, one evidencer is not usually enough to find the best match for a node. Thus, AMMO algorithm employs several evidencers. For example, it is noted previously that B2 appeared to correspond to Q1 based on label probabilities. However, a human expert would know intuitively that B2 should correspond to Q4, because both appear to be in similar positions in the two diagrams. For multiple evidencers need a mechanism for combining one kind of evidence with another. AMMO combines evidence using Bayesian inference on a joint probability distribution over all of the kinds of evidence. The results of combining the label and position evidence. Note that B2 now matches to Q4 with probability five times greater than any other node. Note as well that the possibilities concerning matches for other nodes have been narrowed down considerably.

Table 3 : Probabilities based on Position similarity

	Q1	Q2	Q3	Q4	Q5	Q6
B1	**0.728**	**0.844**	**0.255**	0.003	0.009	0.000
B2	0.010	0.001	**0.313**	0.913	0.022	**0.407**
B3	0.002	0.015	**0.275**	0.022	0.917	**0.238**
B4	0.000	0.000	0.087	**0.659**	0.033	**0.741**

d) Simple Evidencer and Complex Evidencer

The evidencers combining obtained by both the label and position evidencers yielded a very probable match for B2. Beyond this, there is additional evidence that makes this match even more probable. B4, which is a neighbor of B2, matches Q6, which is a neighbor of Q4—having matching neighbors is additional evidence that B2 matches Q4. Our implementation includes a "connection evidencer" that provides such evidence. Evidencers such as the label or position evidencers simple evidencers because they use only information about the given pair of nodes. In contrast, call evidencers like the connection evidencer complex evidencers because they use more than just information about a given pair of nodes to compute the similarity for that pair of nodes—they also use information about other pairs of nodes (in this case: neighboring nodes) that have already been determined to match.

Table 4 : Probabilities based on Both Label and Position Similarity

	Q1	Q2	Q3	Q4	Q5	Q6
B1	**0.230**	**0.375**	0.038	0.000	0.002	0.000
B2	0.003	0.000	0.053	**0.561**	0.003	0.075
B3	0.000	0.002	0.056	0.005	**0.555**	0.039
B4	0.000	0.000	0.012	0.181	0.00	**0.560**

e) Splits and Merges

HLV has four nodes and LLV has six, clearly not every node of LLV can participate in a one-to-one match. It is possible that a node from one diagram matches no nodes from the other diagram. Another possibility is that a node from one diagram matches a combination of nodes from the other diagram. Splits (one-to-many matches) and merges (many-to-one matches) are common in practice. Experts identify splits and merges by combining several pieces of evidence.

For example, an expert might note the following characteristics of HLV and LLV:

- o C1 is close in position to each of P1, P2, and P3.
- o P1, P2, and P3 are interconnected.
- o The combination of P1, P2, and P3 taken together has connections to P4 and to P5, and these connections appear to match the connections from C1 to C2 and to C3.

These characteristics, when taken together, indicate that C1 is likely to have split into P1, P2, and P3, i.e., that C1 matches P1, P2, and P3.

f) Drops

It is also possible that a node in one diagram does not match any node in the other diagram. The probability that a node is dropped as the drop probability, denoted as P_DROP. This probability is determined empirically based on training data.

g) Correspondence Score

The entire correspondence between the two diagrams from individuals matches between nodes. A Naïve approach to find "best" correspondence between two diagrams would be to include the node pairs with the highest pair probabilities. Table 5 below shows the results of combining all evidence about pairs for above example.

Table 3.6 : Pair-wise Match Probabilities based on all evidence

	P1	P2	P3	P4	P5	P6
C1	**104.4**	**189.6**	5.371	0.000	0.001	0.000
C2	0.415	0.019	30.82	**787.0**	2.787	0.037
C3	0.082	0.642	79.93	5.572	**783.0**	0.019
C4	0.000	0.000	1.021	0.100	0.002	**786.3**

If only to consider the pair probabilities shown in Table 5 determine the "best" correspondence to be Corr1 = {(B1, Q2), (B2, Q4), (B3, Q5), (B4, Q6)}. However, this approach fails to yield the optimal correspondence for several reasons. First, although it might result in dropped nodes (when none of the chosen pairs involve a given node), it does not take into consideration the probability of those drops. For example, correspondence Corr1 does not include a match for Q1, and thus, Q1 is a dropped node (as we have defined that above). However, if the probability of a node being dropped is extremely low, it might have been better for Corr1 to include a split (as that was defined above) involving Q1, resulting in a correspondence which is more likely overall. Second, although it might result in splits and merges (when more than one of the chosen pairs involves a given node), this approach does not take into account the probability of these splits and merges. Third, greedily choosing the best pairs, one after the other, does not take into account the fact that choosing a particular pair match can raise or lower the probability of other pair matches, due to complex evidencers such as the connection evidencer.

h) Complexity

A correspondence using only simple node pair evidencers such as label and position, and restrict ourselves to correspondences in which all node matches are one-to-one, then need to find the maximum score correspondence using a polynomial-time algorithm based on maximum-weight bipartite matching. Using complex evidencers and allowing correspondences that are not one-to-one, the problem of identifying the maximum score correspondence is NP-hard.

V. Bayesian Correspondence Model

a) Correspondences and Matches

Let E and E' be diagrams whose nodes are sets N and N', respectively. Our core notion is the diagram correspondence, which equates sets of nodes in N with sets of nodes in N', but also allows nodes to be left out. Formally, Q is a $partial\ partition$ of a set U iff $P = \{a1, a2, ..\}$, where each $a_i \subseteq U$ and $a_i \cap a_j = \emptyset$; for all $i \neq j$. A diagram correspondence for nodes N and N' of two diagrams is a tuple $C = (S, S', f)$, where

S is a partial partition of N;
S' is a partial partition of N';
$f : S \rightarrow S'$ is one to one:

b) Evidencers

Evidencers provide the basis for determining the probability that a pair of nodes match, based on one kind of evidence. Informally, an evidencer consists of three parts: 1) a definition of a node feature (e.g., a node's label), 2) a function that measures the similarity of two nodes based on that feature, and 3) a probability distribution of node pair similarity values in cases where the two nodes match, and a probability distribution of node pair similarity values in cases where the two nodes do not match.

Formally, an evidencer consists of a similarity function e_i and probability functions a_i and b_i.

The similarity function is a function$e_i(x, x')$, where (x, x') is a node pair from (E, E'), where E' is a diagram derived from E by an unspecified procedure $\mathcal{D}$. We model $\mathcal{D}$ by asserting that $e_i(x, x')$ is a random variable. The range of e_i is arbitrary: The set of values used to measure similarity can be chosen to suit the evidencer. For example, the label evidence similarity function $e_l(x, x') = textsim(label(x), label(x))$ returns a real number in the interval $[0,1]$ ($textsim$ is a function that returns a similarity value for two strings: our prototype used a function implemented in the Python standard libraries).

c) Correspondence Probability

In order to use the evidence to find the best correspondence, model the best correspondence as a random variable c that can take any diagram correspondence as its value. Estimation of the best correspondence is the one that has the highest probability given in the evidence.

$$\hat{c} = \arg\max_{c} P(c|e).$$

d) Singular Correspondence Probability Model

The singular correspondence probability model defines the probability of a singular correspondence conditional on the observed evidence.

Let (S, S', f) be a singular correspondence for diagrams containing nodes n and n0. We will use $\mathcal{N}(S)$ to refer to the set of nodes in the partial partition S.We use the notation (x, ϕ) to mean that the node x in the first diagram does not match any node in the second diagram, and similarly for (ϕ, x'). Then,

$$pairs(c) \equiv \{(x, f(x)) | x \in \mathcal{N}(S)\} \cup \{(x,\phi) | x \in N \backslash \mathcal{N}(S)\} \cup \{(\phi, x') | x' \in N' \backslash \mathcal{N}(S)\}$$

Conditional independence allows us to define the correspondence probability as the product of the probability of the pairs:

$$P(c|e) = \prod_{(x,x')\in pairs\,(c)} P(\langle x,x'\rangle | e(x,x'))$$

One-to-none match probability. We assume simply that a node maps to nothing with fixed probability $P(\langle x, \emptyset\rangle) = P(\langle \emptyset, x\rangle) = y_0$. Choose the numerical value of y_0 based on the empirical frequency of one-to-none pairs observed in training data. It may improve accuracy to develop a model of the probability that n maps to nothing based on the features of x. However, in this paper have not implemented such models.

One-to-one match probability model. By adopting a Bayesian model of the probability that one node matches another conditional on the evidence:

$$P(\langle x,x'\rangle | e(x,x')) = \frac{P(\langle x,x'\rangle)P(e(x,x')|\langle x,x'\rangle)}{P(e(x,x'))}$$

Because $\langle x,x'\rangle$ and $\cancel{\langle x,x'\rangle}$ are mutually exclusive events and exhaustive of the space of all possible outcomes with respect to (x,x'), the denominator can be rewritten using a standard normalization technique to get:

$$P(\langle x,x'\rangle | e(x,x')) = \frac{P(\langle x,x'\rangle)P(e(x,x')|\langle x,x'\rangle)}{P(\langle x,x'\rangle)\cdot P(e(x,x')|\langle x,x'\rangle) + P(\cancel{\langle x,x'\rangle})\cdot P(e(x,x')|\cancel{\langle x,x'\rangle})}$$

Assuming that e_i is independent of e_j for all $i \neq j$,

$$P(e(x,x')|\langle x,x'\rangle) = \prod_i P(e_i(x,x')|\langle x,x'\rangle)$$

(and similarly for $\cancel{\langle x,x'\rangle}$), so rewrite once more to get:

$$P(\langle x,x'\rangle | e(x,x')) = \frac{p(1)}{p(1)+p(0)},$$

Where

$$p(1) = P(\langle x,x'\rangle)\prod_i P(e_i(x,x')|\langle x,x'\rangle)$$

$$p(0) = P(\cancel{\langle x,x'\rangle})\prod_i P(e_i(x,x')|\cancel{\langle x,x'\rangle})$$

The factors $P(e_i(x,x')|\langle x,x'\rangle)$ and $P(e_i(x,x')|\cancel{\langle x,x'\rangle})$ are the values that are computed by the probability functions a_i and b_i defined earlier for evidencers.

The factor $P(\langle x,x'\rangle)$ is referred to as a prior. a_i and b_i the prior by decomposing the match event into simpler events, and then, applying commonly used principles of prior selection. First, In this paper notice that the event $\langle x,x'\rangle$ decomposes into two events: E, the event that x matches to some node (i.e., x is not dropped), and F, the event that x matches specifically to x'. Thus, $P(\langle x,x'\rangle) = P(E)P(F|E)$. For $P(E)$, we use a simple empirical prior: $P(E) \equiv 1 - y_0$, where y_0 is the Probability that a node is dropped, as observed in training. For $P(F|E)$, we use an indifference prior: Knowing only that x matches to some node in N', we assume that all nodes are equally likely, so $P(F|E) = 1/|N'|$. This gives us our complete prior: $P(\langle x,x'\rangle) = (1-y_0)/|N'|$.

e) Split-Merge Correspondence Probability Model

The split-merge correspondence probability model is like the singular correspondence probability model, except that paper deal with pairs of sets of nodes rather than pairs of individual nodes decompose a split-merge correspondence $c = (S, S', f)$ into set pairs as follows:

$$spairs(c) \equiv \{(s, f(s)) | s \in S\} \cup \{(\{x\}, \emptyset) | x \in N \backslash \mathcal{N}(S)\} \cup \{(\emptyset, \{x'\}) | x' \in N' \backslash \mathcal{N}(S')\},$$

One-to-many match probability model. For the one-to many case can use a Bayesian model similar to that for the one-to-one case:

$$P(\langle s,s'\rangle | e(s,s')) = \frac{P(\langle s,s'\rangle)P(e(s,s')|\langle s,s'\rangle)}{P(e(s,s'))},$$

and proceed similarly to the one-to-one case, ultimately arriving at the need to compute factors $P(e_w(s,s')|\langle s,s'\rangle)$ and $P(e_w(s,s')|\cancel{\langle s,s'\rangle})$, where the e_w are similar to the e_i of the one-to-one case, except that they deal with sets rather than individual nodes. As well, this need to compute a prior $P(\langle s, s' \rangle)$.

Several issues in computing the factor $P(e_w(x, \{x_1', x_2'\}) | \langle x, \{x_1', x_2'\} \rangle)$, which is the probability according to one kind of evidence (e_w) that the node x matches the set consisting of nodes x_1' and x_2', i.e., that "x splits into x_1' and x_2'", or conversely that "x_1' and x_2' merge into x."

One way that we address these two issues is to define a new kind of evidence based on the evidence about the merge node matching each of the split nodes individually. That is, consider evidence about pairs of nodes, each pair consisting of the merge node and one of the split nodes.

It define

$$P(e_w(x, \{x_1', x_2', \ldots, x_k'\}) | \langle x, \{x_1', x_2', \ldots, x_k'\} \rangle) \equiv P\big(e_i(x, x_j') \big| \langle x, x_j' \rangle\big),$$

Where

$$j = arg \min_{l=1 \ldots k} (e_i(x, x_l')),$$

This define the prior for the one-to-many case as follows: We notice that the event $\langle x, \{x_1', \ldots, x_k'\} \rangle$ decomposes into two events: G, the event that x matches a set of k nodes, and the event H that n matches specifically to $\{x_1', \ldots, x_k'\}$. Thus, $P(\langle x, \{x_1', \ldots, x_k'\} \rangle) = P(G) | P(H|G)$. For $P(G)$, we use the fixed empirical prior, m_k, the observed probability that a node x will match exactly k nodes. For $P(H|G)$, we use an indifference prior: Knowing only that n matches to a set of k nodes in N', we assume that any of the k nodes is equally likely. This yield:

$$P(\langle x, \{x_1', \ldots, x_k'\} \rangle) = \frac{y_k}{\binom{|N'|}{k}}.$$

f) The Maximization Problem

The previous sections showed how to compute $P(c|e)$ for a given correspondence c and evidence e. To complete the algorithm, one should describe how to find the c with maximal $P(c|e)$.

Computing the score of such correspondences using only simple evidencers can be done in polynomial time (ideally constant time per node pair, quadratic overall). To find the maximum probability correspondence in this case, construct a graph which has as its nodes the union of the nodes in the two diagrams, $N \cup N'$. Place an edge from every node n in N to every node n' in N' with edge weight $w(x, x') = P(\langle x, x' \rangle | e(x, x'))$. Now find the maximum probability correspondence in polynomial time using maximum-weight bipartite matching [4].

i. *Greedy Search*

The simplest search algorithm is greedy search. In greedy search, we keep track of only one piece of information, the current state. On each step, we examine all states reachable by a single transition from the current state, and move to the state with the greatest probability. And there is no backtracking—In this paper, only consider transitions that add a node pair to the correspondence, not those that remove a pair. If there is no next state with greater probability than the current state, the search stops.

```
GreedySearch:

BestCorr   : = emptyCorrespondence
/* Initialize the best correspondence to one  in which no node
has a corresponding match in the order diagram */
BestScore: = Score (BestCorr)
FoundBetter: = True
While foundBetter do
            FoundBetter: = False
            BestFoundSoFar: = BestCorr
            BestScoreSoFar: = BestScore
            for each pair <n,n'> that can be added to BestCorr
do
               newCorr  :=  addPairToCorr(BestCorr,<n,n'>)
               newScore  :=  Score(newCorr)
               if newScore > bestScoreSoFar then
                        BestFoundSoFar: = newCorr
                        BestScoreSoFar: = newScore
                        FoundBetter   : = TRUE
               end if
            end for
            if foundBetter then
                BestCorr: = bestFoundSoFar
                BestScore   : = bestScoreSoFar
            end if
end while
return (BestCorr, BestScore)
end GreedySearch
```

Fig. 3 : Greedy Search

Fig. 3 gives a high-level description of the greedy search algorithm for our problem. We assume that, before this algorithm is called, for any nodes n and n' in the two diagrams, we have already computed $p(\langle n, n' \rangle | e(n, n'))$, the probability that they match, based upon the various simple evidencers.

ii. *Complexity Analysis*

Let's assume that the total number of nodes in the diagrams is $O(N)$. Then, the naive implementation of the greedy search algorithm has complexity $O(N^4)$. The outer while loop will be executed at most $O(N)$ times since each iteration removes at least one node of the diagrams from future consideration. The for loop is executed $O(N^2)$, as there are at most N^2 pairs to consider. the naïve implementation, computing Score(newCorr) at line "*" costs $O(N)$ time due to the connection evidencer, which requires $P(\langle y, y' \rangle | e(y, y'))$ to be recomputed for each pair hm;m0i in the correspondence. The connection evidencer will return different values for the probability of $\langle y, y' \rangle$. Hence, the total complexity of the algorithm is $O(N^4)$.

Year 2012 53 Global Journal of Computer Science and Technology (C) Volume XII Issue XI Version I

Global Journal of Computer Science and Technology (C) Volume XII Issue XI Version I Year 2012

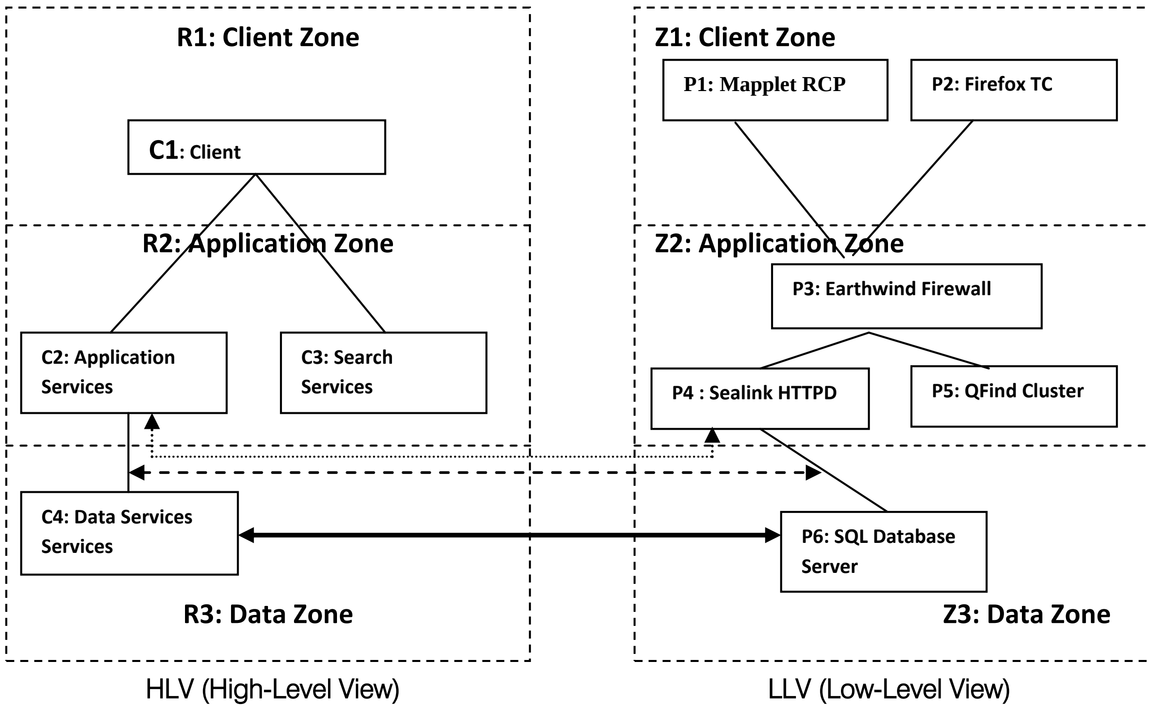


Fig. 4 : Connection matching on the example

iii. *Incremental Algorithm*

It is possible to implement the Score() function so that it takes time proportional to the number of neighbors of the added nodes—probability is only recomputed for pairs that might possibly be affected by a newly added pair. Assuming bounded degree graphs, this *incremental* version takes complexity $O(N^3)$.

VI. Prototype Evidencer

It Describes the set of evidencers that was designed and implemented as part of prototype implementation of the AMMO algorithm. The prototype evidencers can calculate

a) Simple Evidencer

Label Evidencer measures the similarity between text labels of a node pair. Python standard library function difflib.Sequence- Matcher.ratio().

Region Evidencer, A region may have a name, a set of neighboring regions, and a set of nodes that are located within it.

Type Evidencer. Some diagrams have nodes typed as being hardware components or infrastructure software components or application software components (or EJBs or ManagedComponents), while other diagrams have nodes typed as being actors or information flows or use cases or systems.

Position Evidencer Similarity values returned by position evidencer and expect the euclidean distance between matching nodes to be small.

b) Complex Evidencer

A complex evidencer to be an evidence which requires information from more than just the node pair for which it is finding a similarity value. In addition to that node pair, it also takes as input a partial correspondence between the two diagrams.

Connection evidencer. The Connection evidencer is based on the connections, or edges, that each node has to its immediate neighbors.

Fig.3 illustrates connection similarity computation for the pair (B2, Q4) in our sample diagram pair. In this figure, the solid curved line indicates that at this point in the search, the match $\langle B4, Q6\rangle$ is already part of the correspondence. The dotted curved line indicates that we are considering the node pair (B2, Q4). By virtue of the facts that B2 has two neighbors (B1 and B4), Q4 has two neighbors (Q3 and Q6), and one of B2's two neighbors (B4) matches one of Q4's two neighbors (Q6), as indicated by the dashed line, the connection similarity for (B2, Q4) is $avg\left(\frac{1}{2},\frac{1}{2}\right) = 0.5$. Ultimately, connections turn out to be strong evidence that B2 and Q4 match.

c) Split Evidencer

A Split-Merge Model which defined the probability of a split-merge correspondence conditional on the observed evidence. Recall that a split-merge correspondence is one containing split-merge matches—matches between one node and a set of nodes. Further, recall that, to evaluate the probability of such correspondences, two types of evidencers are used: simple (pair) evidencers and split evidencers. The simple evidencers that were implemented as part of our prototype, and this section describes the split evidencers of our prototype.

The motivation for creating special-purpose split evidencers arose out of the observation that split-merge correspondences exhibited different characteristics than singular correspondences and that these characteristics were not taken into account by the simple evidencers.

Label Sim evidencer. The similarity determined by the Label Sim evidencer is the minimum similarity among the labels of the nodes.

Label Intersect evidencer. The similarity determined by the Label Intersect evidencer is the similarity between the label of x and the longest suffix or prefix commonto the labels of the x_i' nodes.

Label Concat evidencer. The Label Concat Evidencer similarity function uses the Label Evidencer similarity function to obtain the similarity between the label of x and the concatenation of the labels of the x_i' nodes.

Inner Connect evidencer. This is a discrete measure of similarity based on whether or not all of the x_i' nodes are connected to each other.

Outer Connect evidencer. This is a continuous measure of connection similarity between x and the cluster of x_i' nodes taken as a whole.

VII. Ammo-Lite: Improving Performance and Scalability

Although the greedy search algorithm described performed well for diagrams with dozens of nodes, it was not practical for diagrams with hundreds of nodes. the major scalability problem with AMMO is that every time it has to decide which node pair to add next, it must compute an exact probability for each possible correspondence that would result from adding one more node pair. Our incremental version of greedy search helps avoid some of this recomputation, but not enough to be practical for larger-scale diagrams. To solve this problem, we designed a new algorithm, AMMO-LITE, which approximates AMMO's behavior but uses a simpler search that is driven by pair probabilities rather than correspondence probabilities. This approach avoids repeated calculation of correspondence probabilities and, in practice, achieves much better performance with only a small loss of precision.

```
AMMO-LITE
Use simple evidencers to precompute and store
probabilities of all pairs < n,  m >
PotentialPairs := list of all pairs < n,  m > in descending
order of     probability
Corr := emptyCorrespondence
Done = False
While PotentialPairs is not Empty and
        Prob(first(PotentialPairs)) > threshold do
       <n,  m> :=  removeFirst (PotentialPairs)
       if <n, m> can be added to Corr then
              must_re_sort := False
              Corr := addPairToCorr(Corr, <n,  m>)
              for each pair <<nn, mm> in PotentialPairs do
                  if nn == n or mm == m then
    use split evs to update the probability of <nn, mm>
                     must_re_sort  : =  True
                  else if nn is neighbor of n and
                     mm is a neighbor of m then
    use connect ev to update the probability of <<nn,mm>
                     must_re_sort  : =  True
                  end if
              end for
              if must_re_sort then
                  PotentialPairs  : = re-sort(PotentialPairs)
              end if
        end if
end while
```

Fig. 5: Ammo-Lite

a) Algorithm Description

It uses the probabilities of the pairs to determine the order in which pairs should be added to the correspondence. This is done as follows:

As in the case of AMMO, the first thing that the algorithm does is to precompute probabilities of all possible node pairs, using the simple evidencers. It then creates a sorted list Potential Pairs, which contains the node pairs sorted in descending order by probability.

The main loop of AMMO-LITE goes through Potential-Pairs, adding the highest probability pair (the one at the head of the list) to the correspondence, provided that it is permissible to add that pair. It is not permissible to add a pair. It is not permissible to add a pair to the correspondence if that would result in a many-to-many match. Each time a new pair $\langle x, y\rangle$ is added to the correspondence, the algorithm goes through the list again, in order to determine if the precomputed probability of any remaining pair $\langle xx, yy\rangle$ has been affected. The probability of pair $\langle xx, yy\rangle$ in PotentialPairs will be affected in two different circumstances:

- If xx or yy is one of the nodes in the pair we just added, then adding $\langle xx, yy\rangle$ would result in a split/merge. Thus, we change the precomputed probability stored for $\langle xx, yy\rangle$ to be the probability of the split/merge that would result from adding $\langle xx, yy\rangle$ to the correspondence.
- If xx and yy are neighbors of x and y, respectively, then adding $\langle x, y\rangle$ to the correspondence will affect the connectivity similarity of $\langle xx, yy\rangle$. Thus, $P(\langle xx, yy\rangle)$ must be recomputed, this time using the connection evidencer as well as the simple evidencers.

After going through PotentialPairs, if any probabilities have been recomputed, the list is resorted. The algorithm then continues with another iteration of the main loop to add another pair to the correspondence. The algorithm terminates when either the list is empty or the probability of the pair at the head of the list is less than some threshold value. This value is determined by experimentation with training data, and can be easily changed. In our implementation, this threshold is m_0, the empirically determined probability that a node does not correspond to any node in the other diagram.

b) Complexity Analysis

Let the total number of nodes in a diagram be $O(N)$, as in the analysis of AMMO. Depending on the value of threshold, the outer while loop could be executed $O(N^2)$ times, once for every possible node pair. However, the outer if statement (immediately within the while loop) will only be true $O(N)$ times since each pair added must add at least one new node to the correspondence, due to the many-to-many restriction, and hence, add at most $O(N)$ pairs. Thus, the nested for loop will be reached on only $O(N)$ iterations of the while loop. Each time the for loop is reached, it will execute $O(|PotentialPairs|) = O(N^2)$ iterations. The resulting total complexity is $O(N^3)$. Similarly, like the nested for loop, the statement resort(PotentialPairs) will be reached at most $O(N)$ times. Sorting being $O(NlogN)$, each sort of the $O(N^2)$ items in PotentialPairs will have complexity $O(N^2logN)$. Thus, the resulting total complexity of the algorithm due to all sorting is $O(N^3logN)$. That dominates the $O(N^3)$ of the nested for loop, and therefore, the overall worst-case total complexity of the AMMO-LITE algorithm is $O(N^3logN)$.

Table 7: Experiment Results:
Average Algorithm Recall, Precision and Runtime
(in Seconds)

Algorithm	Recall %	Precision %	Time
Baseline (non-Bayesian)	75	70	3
AMMO (all evidencers)	82	85	82
AMMO-LITE (all evidencers)	80	84	3

To see why AMMO-LITE performs better than AMMO in practice, consider the following: In AMMO-LITE, each timewe add a pair $\langle x, y\rangle$ and make a pass through the list PotentialPairs. Although this list can be $O(N^2)$, it is a "quick" pass over the list—most of the pairs are just skipped. "Real" computation only takes place if $\langle xx, yy\rangle$ meets certain criteria in which case, we recompute its associated probability. So, in practice, our performance is better than $O(N^3logN)$ would suggest.

In fact, employing a priority queue along with an incremental approach to updating pair probabilities, and assuming a bounded-degree graph, we could achieve an overall total complexity of $O(N^2logN)$ as follows: This can implement PotentialPairs as a priority queue in which pairs are ordered according to their probability, there by obviating the need for separate explicit sorts. Initially, we construct PotentialPairs by inserting all of the $O(N^2)$ pairs into it. With a priority queue implementation for which insert, get_max, and delete are $O(logN)$, the complexity of constructing PotentialPairs is $O(N^2logN)$. In that way, we avoid having to reexamine all of the $O(N^2)$ remaining pairs in PotentialPairs. Assuming bounded-degree graphs, with the number of neighbors of a pair $\langle x, y\rangle$ being bounded by a constant k, the number of pairs whose probability must be recomputed due to connectivity is k. Whenever we recomputed the probability of a pair and delete it from Potential-Pairs and reinsert it with its new probability (or we could simply do a change_priority operation). With delete and insert being $O(logN)$, the total complexity due to recomputing probabilities of neighbors of all of the $O(N)$ added pairs is $O(N * k * logN) = O(NlogN)$. Similarly, when a pair $\langle x, y\rangle$ is added to the correspondence, the number of pairs $\langle xx. yy\rangle$ whose probability must be recomputed because they would now result in splits or merges is at most $O(N)$ because there are at most $O(N)$ pairs for which $x = xx$ or $y = yy$. Hence, the total complexity due to recomputing probabilities due to split/ merge considerations is $O(N * N * logN) = O(N^2logN)$. Thus, the overall total complexity of the algorithm would be $O(N^2logN)$.

c) AMMO-LITE Experiments

Table 7 compares the results of running AMMO-LITE against those obtained by running AMMO and

Baseline. The results include accuracy as well as the runtime (in seconds) required by each algorithm.

Table 8 : Experimental Results: AMMO-LITE versus AMMO Runtimes (in seconds) for Various Diagram Sizes

Pair	Nodes	Edges	AMMO-LITE	AMMO	Ratio
Pair 1	9	13	0.83	2.74	3.3
Pair 2	12	12	0.96	3.68	3.8
Pair 3	15	24	2.63	12.19	4.6
Pair 4	22	41	5.11	41.49	8.1
Pair 5	35	31	11.30	98.66	8.7
Pair 6	41	68	15.51	257.27	16.6
Pair 7	637	968	6175.00	-	-

The values in the table were obtained by averaging the Recall, Precision, and Time metrics for each algorithm across all of our model pairs.

The AMMO-LITE algorithm did not do quite as well as AMMO in terms of both Recall and Precision, but it still did significantly better than the non-Bayesian approach. Furthermore, if one examines the cases where AMMO-LITE did poorly in comparison to AMMO, most of these cases involved complex correspondences with a number of challenging matches and multiple split/merges.

VIII. Results

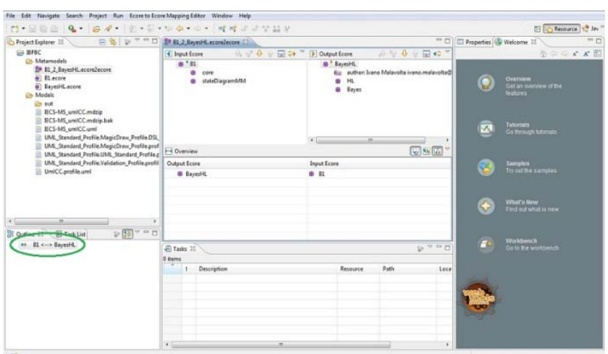


Fig. 6 :(a). Mapping between two ADL Models

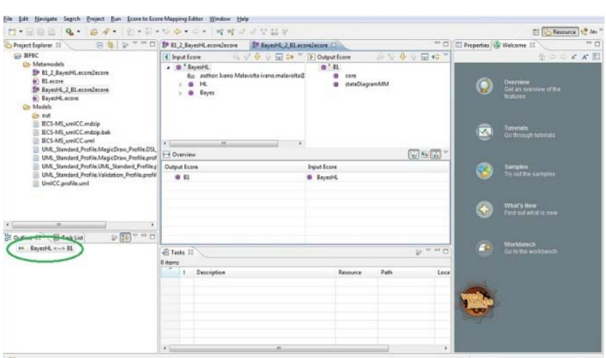


Fig. 6 :(b). Mapping between two ADL Models

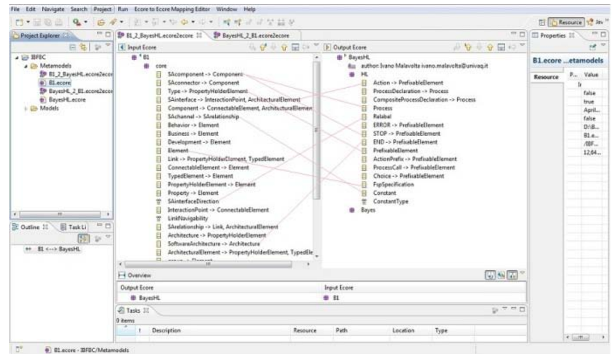


Fig. 7 :(a). Mapping the objects between two ADL Models

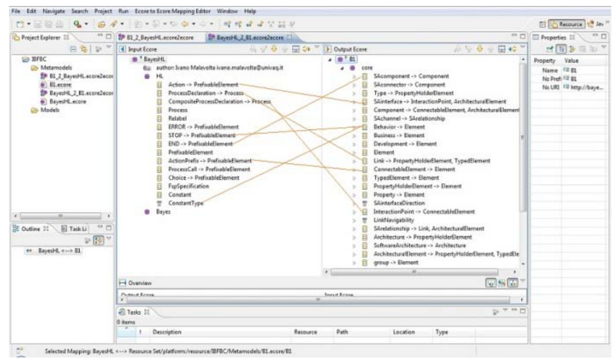


Fig. 7 :(b). Mapping the objects between two ADL Models

IX. Conclusion

We have identified and described the model correspondence problem, an important problem in software engineering. We have designed a Bayesian framework that supports the reasoning needed to solve the model correspondence problem. And we have implemented and tested a matching algorithm based on our framework, finding that it achieved high accuracy on a set of test diagram pairs. We believe that this work holds great promise for the future.

References Références Referencias

1. Garland and R. Anthony, Large-Scale Software Architecture. John Wiley and Sons, 2003.
2. R. Youngs, D. Redmond-Pyle, P. Spaas, and E. Kahan, “A Standard for Architecture Description,” IBM Systems J., vol. 38, no. 1, pp. 32-50, 1999.
3. Z. Xing and E. Stroulia, “Umldiff: An Algorithm for Object-Oriented Design Differencing,” Proc. 20th IEEE/ACM Int’l Conf. Automated Software Eng., pp. 54-65, 2005.
4. D.E. Tarjan, Data Structures and Network Algorithms. SIAM, Nov. 1983.
5. B. Ramesh and M. Jarke, “Towards a Reference Model for Requirements Traceability,” IEEE Trans. Software Eng., vol. 27, no. 1, pp. 58-93, Jan. 2001.
6. Jossic, M.D.D. Fabro, J.-P. Lerat, J. Bezivin, and F. Jouault, “Model Integration with Model Weaving: A

Case Study in System Architecture," Proc. Int'l Conf. Systems Eng. and Modeling, pp. 79-84, 2007.

7. P. Kruchten, "Architectural Blueprints-The 4 + 1 View Model of Software Architecture," IEEE Software, vol. 12, no. 6, pp. 42-50, Nov. 1995.
8. D. Mandelin, D. Kimelman, and D.M. Yellin, "A Bayesian Approach to Diagram Matching with Application to Architectural Models," Proc. 28th Int'l Conf. Software Eng., May 2006.
9. N. Rozanski and E. Woods, Software Systems Architecture: Working with Stakeholders Using Viewpoints and Perspectives. Addison- Wesley, 2005.
10. G. Antoniol, G. Canfora, G. Casazza, and A.D. Lucia, "Maintaining Traceability Links during Object-Oriented Software Evolution," Software-Practice and Experience, vol. 31, pp. 331-355, 2001.
11. Handbook on Architectures of Information Systems, pp. 669- 692. Springer, 2006.
12. IBM Insurance Application Architecture-Executive Summary,http://www.03.ibm.com/industries/insurance/us/detail/solution/P669447B27619A15.html, 2009.
13. TM Forum-Information Framework (SID). http://www.tmforum.org/InformationFramework/1684/home.html, 2009.
14. NGOSS Shared Information/Data Model, http://en.wikipedia.org/wiki/NGOSS_Shared_Information/Data_Model, 2009.
15. WebSVN-Diplomarbeit,http://surprise.wh-stuttgart.de/websvn/log.php?repname=diplomarbeit**path=%2Ftrunk%2Fdesign%2FMiddlewareUML.xmi**rev=87**sc=1**isdir=0, 2009.
16. S. Abrams, B. Bloom, P. Keyser, D. Kimelman, E. Nelson, W. Neuberger, T. Roth, I. Simmonds, S. Tang, and J. Vlissides, "Architectural Thinking and Modeling with AWB: The Architects Workbench," IBM Systems J., vol. 45, no. 3, pp. 481-500, 2006.

Global Journals Inc. (US) Guidelines Handbook 2012

www.GlobalJournals.org

FELLOW OF ASSOCIATION OF RESEARCH SOCIETY IN COMPUTING (FARSC)

- 'FARSC' title will be awarded to the person after approval of Editor-in-Chief and Editorial Board. The title 'FARSC" can be added to name in the following manner. eg. **Dr. John E. Hall, Ph.D., FARSC or William Walldroff Ph. D., M.S., FARSC**
- Being FARSC is a respectful honor. It authenticates your research activities. After becoming FARSC, you can use 'FARSC' title as you use your degree in suffix of your name. This will definitely will enhance and add up your name. You can use it on your Career Counseling Materials/CV/Resume/Visiting Card/Name Plate etc.
- 60% Discount will be provided to FARSC members for publishing research papers in Global Journals Inc., if our Editorial Board and Peer Reviewers accept the paper. For the life time, if you are author/co-author of any paper bill sent to you will automatically be discounted one by 60%
- FARSC will be given a renowned, secure, free professional email address with 100 GB of space eg.johnhall@globaljournals.org. You will be facilitated with Webmail, Spam Assassin, Email Forwarders, Auto-Responders, Email Delivery Route tracing, etc.
- FARSC member is eligible to become paid peer reviewer at Global Journals Inc. to earn up to 15% of realized author charges taken from author of respective paper. After reviewing 5 or more papers you can request to transfer the amount to your bank account or to your PayPal account.
- Eg. If we had taken 420 USD from author, we can send 63 USD to your account.
- FARSC member can apply for free approval, grading and certification of some of their Educational and Institutional Degrees from Global Journals Inc. (US) and Open Association of Research, Society U.S.A.
- After you are FARSC. You can send us scanned copy of all of your documents. We will verify, grade and certify them within a month. It will be based on your academic records, quality of research papers published by you, and 50 more criteria. This is beneficial for your job interviews as recruiting organization need not just rely on you for authenticity and your unknown qualities, you would have authentic ranks of all of your documents. Our scale is unique worldwide.
- FARSC member can proceed to get benefits of free research podcasting in Global Research Radio with their research documents, slides and online movies.
- After your publication anywhere in the world, you can upload you research paper with your recorded voice or you can use our professional RJs to record your paper their voice. We can also stream your conference videos and display your slides online.
- FARSC will be eligible for free application of Standardization of their Researches by Open Scientific Standards. Standardization is next step and level after publishing in a journal. A team of research and professional will work with you to take your research to its next level, which is worldwide open standardization.

- FARSC is eligible to earn from their researches: While publishing his paper with Global Journals Inc. (US), FARSC can decide whether he/she would like to publish his/her research in closed manner. When readers will buy that individual research paper for reading, 80% of its earning by Global Journals Inc. (US) will be transferred to FARSC member's bank account after certain threshold balance. There is no time limit for collection. FARSC member can decide its price and we can help in decision.

MEMBER OF ASSOCIATION OF RESEARCH SOCIETY IN COMPUTING (MARSC)

- 'MARSC' title will be awarded to the person after approval of Editor-in-Chief and Editorial Board. The title 'MARSC" can be added to name in the following manner. eg. Dr. John E. Hall, Ph.D., MARSC or William Walldroff Ph. D., M.S., MARSC
- Being MARSC is a respectful honor. It authenticates your research activities. After becoming MARSC, you can use 'MARSC' title as you use your degree in suffix of your name. This will definitely will enhance and add up your name. You can use it on your Career Counseling Materials/CV/Resume/Visiting Card/Name Plate etc.
- 40% Discount will be provided to MARSC members for publishing research papers in Global Journals Inc., if our Editorial Board and Peer Reviewers accept the paper. For the life time, if you are author/co-author of any paper bill sent to you will automatically be discounted one by 60%
- MARSC will be given a renowned, secure, free professional email address with 30 GB of space eg.johnhall@globaljournals.org. You will be facilitated with Webmail, Spam Assassin, Email Forwarders, Auto-Responders, Email Delivery Route tracing, etc.
- MARSC member is eligible to become paid peer reviewer at Global Journals Inc. to earn up to 10% of realized author charges taken from author of respective paper. After reviewing 5 or more papers you can request to transfer the amount to your bank account or to your PayPal account.
- MARSC member can apply for free approval, grading and certification of some of their Educational and Institutional Degrees from Global Journals Inc. (US) and Open Association of Research, Society U.S.A.
- MARSC is eligible to earn from their researches: While publishing his paper with Global Journals Inc. (US), MARSC can decide whether he/she would like to publish his/her research in closed manner. When readers will buy that individual research paper for reading, 40% of its earning by Global Journals Inc. (US) will be transferred to MARSC member's bank account after certain threshold balance. There is no time limit for collection. MARSC member can decide its price and we can help in decision.

ANNUAL MEMBER

- Annual Member will be authorized to receive e-Journal GJCST for one year (subscription for one year).
- The member will be allotted free 1 GB Web-space along with subDomain to contribute and participate in our activities.
- A professional email address will be allotted free 500 MB email space.

PAPER PUBLICATION

- The members can publish paper once. The paper will be sent to two-peer reviewer. The paper will be published after the acceptance of peer reviewers and Editorial Board.

Process of submission of Research Paper

The Area or field of specialization may or may not be of any category as mentioned in 'Scope of Journal' menu of the GlobalJournals.org website. There are 37 Research Journal categorized with Six parental Journals GJCST, GJMR, GJRE, GJMBR, GJSFR, GJHSS. For Authors should prefer the mentioned categories. There are three widely used systems UDC, DDC and LCC. The details are available as 'Knowledge Abstract' at Home page. The major advantage of this coding is that, the research work will be exposed to and shared with all over the world as we are being abstracted and indexed worldwide.

The paper should be in proper format. The format can be downloaded from first page of 'Author Guideline' Menu. The Author is expected to follow the general rules as mentioned in this menu. The paper should be written in MS-Word Format (*.DOC,*.DOCX).

The Author can submit the paper either online or offline. The authors should prefer online submission.Online Submission: There are three ways to submit your paper:

(A) (I) First, register yourself using top right corner of Home page then Login. If you are already registered, then login using your username and password.

(II) Choose corresponding Journal.

(III) Click 'Submit Manuscript'. Fill required information and Upload the paper.

(B) If you are using Internet Explorer, then Direct Submission through Homepage is also available.

(C) If these two are not convenient, and then email the paper directly to dean@globaljournals.org.

Offline Submission: Author can send the typed form of paper by Post. However, online submission should be preferred.

Preferred Author Guidelines

MANUSCRIPT STYLE INSTRUCTION **<u>(Must be strictly followed)</u>**

Page Size: 8.27" X 11'"

- Left Margin: 0.65
- Right Margin: 0.65
- Top Margin: 0.75
- Bottom Margin: 0.75
- Font type of all text should be Swis 721 Lt BT.
- Paper Title should be of Font Size 24 with one Column section.
- Author Name in Font Size of 11 with one column as of Title.
- Abstract Font size of 9 Bold, “Abstract” word in Italic Bold.
- Main Text: Font size 10 with justified two columns section
- Two Column with Equal Column with of 3.38 and Gaping of .2
- First Character must be three lines Drop capped.
- Paragraph before Spacing of 1 pt and After of 0 pt.
- Line Spacing of 1 pt
- Large Images must be in One Column
- Numbering of First Main Headings (Heading 1) must be in Roman Letters, Capital Letter, and Font Size of 10.
- Numbering of Second Main Headings (Heading 2) must be in Alphabets, Italic, and Font Size of 10.

You can use your own standard format also.
Author Guidelines:

1. General,

2. Ethical Guidelines,

3. Submission of Manuscripts,

4. Manuscript’s Category,

5. Structure and Format of Manuscript,

6. After Acceptance.

1. GENERAL

Before submitting your research paper, one is advised to go through the details as mentioned in following heads. It will be beneficial, while peer reviewer justify your paper for publication.

Scope

The Global Journals Inc. (US) welcome the submission of original paper, review paper, survey article relevant to the all the streams of Philosophy and knowledge. The Global Journals Inc. (US) is parental platform for Global Journal of Computer Science and Technology, Researches in Engineering, Medical Research, Science Frontier Research, Human Social Science, Management, and Business organization. The choice of specific field can be done otherwise as following in Abstracting and Indexing Page on this Website. As the all Global

Journals Inc. (US) are being abstracted and indexed (in process) by most of the reputed organizations. Topics of only narrow interest will not be accepted unless they have wider potential or consequences.

2. ETHICAL GUIDELINES

Authors should follow the ethical guidelines as mentioned below for publication of research paper and research activities.

Papers are accepted on strict understanding that the material in whole or in part has not been, nor is being, considered for publication elsewhere. If the paper once accepted by Global Journals Inc. (US) and Editorial Board, will become the copyright of the Global Journals Inc. (US).

Authorship: The authors and coauthors should have active contribution to conception design, analysis and interpretation of findings. They should critically review the contents and drafting of the paper. All should approve the final version of the paper before submission

The Global Journals Inc. (US) follows the definition of authorship set up by the Global Academy of Research and Development. According to the Global Academy of R&D authorship, criteria must be based on:

1) Substantial contributions to conception and acquisition of data, analysis and interpretation of the findings.

2) Drafting the paper and revising it critically regarding important academic content.

3) Final approval of the version of the paper to be published.

All authors should have been credited according to their appropriate contribution in research activity and preparing paper. Contributors who do not match the criteria as authors may be mentioned under Acknowledgement.

Acknowledgements: Contributors to the research other than authors credited should be mentioned under acknowledgement. The specifications of the source of funding for the research if appropriate can be included. Suppliers of resources may be mentioned along with address.

Appeal of Decision: The Editorial Board's decision on publication of the paper is final and cannot be appealed elsewhere.

Permissions: It is the author's responsibility to have prior permission if all or parts of earlier published illustrations are used in this paper.

Please mention proper reference and appropriate acknowledgements wherever expected.

If all or parts of previously published illustrations are used, permission must be taken from the copyright holder concerned. It is the author's responsibility to take these in writing.

Approval for reproduction/modification of any information (including figures and tables) published elsewhere must be obtained by the authors/copyright holders before submission of the manuscript. Contributors (Authors) are responsible for any copyright fee involved.

3. SUBMISSION OF MANUSCRIPTS

Manuscripts should be uploaded via this online submission page. The online submission is most efficient method for submission of papers, as it enables rapid distribution of manuscripts and consequently speeds up the review procedure. It also enables authors to know the status of their own manuscripts by emailing us. Complete instructions for submitting a paper is available below.

Manuscript submission is a systematic procedure and little preparation is required beyond having all parts of your manuscript in a given format and a computer with an Internet connection and a Web browser. Full help and instructions are provided on-screen. As an author, you will be prompted for login and manuscript details as Field of Paper and then to upload your manuscript file(s) according to the instructions.

To avoid postal delays, all transaction is preferred by e-mail. A finished manuscript submission is confirmed by e-mail immediately and your paper enters the editorial process with no postal delays. When a conclusion is made about the publication of your paper by our Editorial Board, revisions can be submitted online with the same procedure, with an occasion to view and respond to all comments.

Complete support for both authors and co-author is provided.

4. MANUSCRIPT'S CATEGORY

Based on potential and nature, the manuscript can be categorized under the following heads:

Original research paper: Such papers are reports of high-level significant original research work.

Review papers: These are concise, significant but helpful and decisive topics for young researchers.

Research articles: These are handled with small investigation and applications

Research letters: The letters are small and concise comments on previously published matters.

5.STRUCTURE AND FORMAT OF MANUSCRIPT

The recommended size of original research paper is less than seven thousand words, review papers fewer than seven thousands words also.Preparation of research paper or how to write research paper, are major hurdle, while writing manuscript. The research articles and research letters should be fewer than three thousand words, the structure original research paper; sometime review paper should be as follows:

Papers: These are reports of significant research (typically less than 7000 words equivalent, including tables, figures, references), and comprise:

(a)Title should be relevant and commensurate with the theme of the paper.

(b) A brief Summary, "Abstract" (less than 150 words) containing the major results and conclusions.

(c) Up to ten keywords, that precisely identifies the paper's subject, purpose, and focus.

(d) An Introduction, giving necessary background excluding subheadings; objectives must be clearly declared.

(e) Resources and techniques with sufficient complete experimental details (wherever possible by reference) to permit repetition; sources of information must be given and numerical methods must be specified by reference, unless non-standard.

(f) Results should be presented concisely, by well-designed tables and/or figures; the same data may not be used in both; suitable statistical data should be given. All data must be obtained with attention to numerical detail in the planning stage. As reproduced design has been recognized to be important to experiments for a considerable time, the Editor has decided that any paper that appears not to have adequate numerical treatments of the data will be returned un-refereed;

(g) Discussion should cover the implications and consequences, not just recapitulating the results; conclusions should be summarizing.

(h) Brief Acknowledgements.

(i) References in the proper form.

Authors should very cautiously consider the preparation of papers to ensure that they communicate efficiently. Papers are much more likely to be accepted, if they are cautiously designed and laid out, contain few or no errors, are summarizing, and be conventional to the approach and instructions. They will in addition, be published with much less delays than those that require much technical and editorial correction.

The Editorial Board reserves the right to make literary corrections and to make suggestions to improve briefness.

It is vital, that authors take care in submitting a manuscript that is written in simple language and adheres to published guidelines.

Format

Language: The language of publication is UK English. Authors, for whom English is a second language, must have their manuscript efficiently edited by an English-speaking person before submission to make sure that, the English is of high excellence. It is preferable, that manuscripts should be professionally edited.

Standard Usage, Abbreviations, and Units: Spelling and hyphenation should be conventional to The Concise Oxford English Dictionary. Statistics and measurements should at all times be given in figures, e.g. 16 min, except for when the number begins a sentence. When the number does not refer to a unit of measurement it should be spelt in full unless, it is 160 or greater.

Abbreviations supposed to be used carefully. The abbreviated name or expression is supposed to be cited in full at first usage, followed by the conventional abbreviation in parentheses.

Metric SI units are supposed to generally be used excluding where they conflict with current practice or are confusing. For illustration, 1.4 l rather than 1.4 × 10-3 m3, or 4 mm somewhat than 4 × 10-3 m. Chemical formula and solutions must identify the form used, e.g. anhydrous or hydrated, and the concentration must be in clearly defined units. Common species names should be followed by underlines at the first mention. For following use the generic name should be constricted to a single letter, if it is clear.

Structure

All manuscripts submitted to Global Journals Inc. (US), ought to include:

Title: The title page must carry an instructive title that reflects the content, a running title (less than 45 characters together with spaces), names of the authors and co-authors, and the place(s) wherever the work was carried out. The full postal address in addition with the e-mail address of related author must be given. Up to eleven keywords or very brief phrases have to be given to help data retrieval, mining and indexing.

Abstract, used in Original Papers and Reviews:

Optimizing Abstract for Search Engines

Many researchers searching for information online will use search engines such as Google, Yahoo or similar. By optimizing your paper for search engines, you will amplify the chance of someone finding it. This in turn will make it more likely to be viewed and/or cited in a further work. Global Journals Inc. (US) have compiled these guidelines to facilitate you to maximize the web-friendliness of the most public part of your paper.

Key Words

A major linchpin in research work for the writing research paper is the keyword search, which one will employ to find both library and Internet resources.

One must be persistent and creative in using keywords. An effective keyword search requires a strategy and planning a list of possible keywords and phrases to try.

Search engines for most searches, use Boolean searching, which is somewhat different from Internet searches. The Boolean search uses "operators," words (and, or, not, and near) that enable you to expand or narrow your affords. Tips for research paper while preparing research paper are very helpful guideline of research paper.

Choice of key words is first tool of tips to write research paper. Research paper writing is an art.A few tips for deciding as strategically as possible about keyword search:

- One should start brainstorming lists of possible keywords before even begin searching. Think about the most important concepts related to research work. Ask, "What words would a source have to include to be truly valuable in research paper?" Then consider synonyms for the important words.
- It may take the discovery of only one relevant paper to let steer in the right keyword direction because in most databases, the keywords under which a research paper is abstracted are listed with the paper.
- One should avoid outdated words.

Keywords are the key that opens a door to research work sources. Keyword searching is an art in which researcher's skills are bound to improve with experience and time.

Numerical Methods: Numerical methods used should be clear and, where appropriate, supported by references.

Acknowledgements: Please make these as concise as possible.

References

References follow the Harvard scheme of referencing. References in the text should cite the authors' names followed by the time of their publication, unless there are three or more authors when simply the first author's name is quoted followed by et al. unpublished work has to only be cited where necessary, and only in the text. Copies of references in press in other journals have to be supplied with submitted typescripts. It is necessary that all citations and references be carefully checked before submission, as mistakes or omissions will cause delays.

References to information on the World Wide Web can be given, but only if the information is available without charge to readers on an official site. Wikipedia and Similar websites are not allowed where anyone can change the information. Authors will be asked to make available electronic copies of the cited information for inclusion on the Global Journals Inc. (US) homepage at the judgment of the Editorial Board.

The Editorial Board and Global Journals Inc. (US) recommend that, citation of online-published papers and other material should be done via a DOI (digital object identifier). If an author cites anything, which does not have a DOI, they run the risk of the cited material not being noticeable.

The Editorial Board and Global Journals Inc. (US) recommend the use of a tool such as Reference Manager for reference management and formatting.

Tables, Figures and Figure Legends

Tables: Tables should be few in number, cautiously designed, uncrowned, and include only essential data. Each must have an Arabic number, e.g. Table 4, a self-explanatory caption and be on a separate sheet. Vertical lines should not be used.

Figures: Figures are supposed to be submitted as separate files. Always take in a citation in the text for each figure using Arabic numbers, e.g. Fig. 4. Artwork must be submitted online in electronic form by e-mailing them.

Preparation of Electronic Figures for Publication

Even though low quality images are sufficient for review purposes, print publication requires high quality images to prevent the final product being blurred or fuzzy. Submit (or e-mail) EPS (line art) or TIFF (halftone/photographs) files only. MS PowerPoint and Word Graphics are unsuitable for printed pictures. Do not use pixel-oriented software. Scans (TIFF only) should have a resolution of at least 350 dpi (halftone) or 700 to 1100 dpi (line drawings) in relation to the imitation size. Please give the data for figures in black and white or submit a Color Work Agreement Form. EPS files must be saved with fonts embedded (and with a TIFF preview, if possible).

For scanned images, the scanning resolution (at final image size) ought to be as follows to ensure good reproduction: line art: >650 dpi; halftones (including gel photographs) : >350 dpi; figures containing both halftone and line images: >650 dpi.

Color Charges: It is the rule of the Global Journals Inc. (US) for authors to pay the full cost for the reproduction of their color artwork. Hence, please note that, if there is color artwork in your manuscript when it is accepted for publication, we would require you to complete and return a color work agreement form before your paper can be published.

Figure Legends: Self-explanatory legends of all figures should be incorporated separately under the heading 'Legends to Figures'. In the full-text online edition of the journal, figure legends may possibly be truncated in abbreviated links to the full screen version. Therefore, the first 100 characters of any legend should notify the reader, about the key aspects of the figure.

6. AFTER ACCEPTANCE

Upon approval of a paper for publication, the manuscript will be forwarded to the dean, who is responsible for the publication of the Global Journals Inc. (US).

6.1 Proof Corrections

The corresponding author will receive an e-mail alert containing a link to a website or will be attached. A working e-mail address must therefore be provided for the related author.

Acrobat Reader will be required in order to read this file. This software can be downloaded

(Free of charge) from the following website:

www.adobe.com/products/acrobat/readstep2.html. This will facilitate the file to be opened, read on screen, and printed out in order for any corrections to be added. Further instructions will be sent with the proof.

Proofs must be returned to the dean at dean@globaljournals.org within three days of receipt.

As changes to proofs are costly, we inquire that you only correct typesetting errors. All illustrations are retained by the publisher. Please note that the authors are responsible for all statements made in their work, including changes made by the copy editor.

6.2 Early View of Global Journals Inc. (US) (Publication Prior to Print)

The Global Journals Inc. (US) are enclosed by our publishing's Early View service. Early View articles are complete full-text articles sent in advance of their publication. Early View articles are absolute and final. They have been completely reviewed, revised and edited for publication, and the authors' final corrections have been incorporated. Because they are in final form, no changes can be made after sending them. The nature of Early View articles means that they do not yet have volume, issue or page numbers, so Early View articles cannot be cited in the conventional way.

6.3 Author Services

Online production tracking is available for your article through Author Services. Author Services enables authors to track their article - once it has been accepted - through the production process to publication online and in print. Authors can check the status of their articles online and choose to receive automated e-mails at key stages of production. The authors will receive an e-mail with a unique link that enables them to register and have their article automatically added to the system. Please ensure that a complete e-mail address is provided when submitting the manuscript.

6.4 Author Material Archive Policy

Please note that if not specifically requested, publisher will dispose off hardcopy & electronic information submitted, after the two months of publication. If you require the return of any information submitted, please inform the Editorial Board or dean as soon as possible.

6.5 Offprint and Extra Copies

A PDF offprint of the online-published article will be provided free of charge to the related author, and may be distributed according to the Publisher's terms and conditions. Additional paper offprint may be ordered by emailing us at: editor@globaljournals.org .

the search? Will I be able to find all information in this field area? If the answer of these types of questions will be "Yes" then you can choose that topic. In most of the cases, you may have to conduct the surveys and have to visit several places because this field is related to Computer Science and Information Technology. Also, you may have to do a lot of work to find all rise and falls regarding the various data of that subject. Sometimes, detailed information plays a vital role, instead of short information.

2. Evaluators are human: First thing to remember that evaluators are also human being. They are not only meant for rejecting a paper. They are here to evaluate your paper. So, present your Best.

3. Think Like Evaluators: If you are in a confusion or getting demotivated that your paper will be accepted by evaluators or not, then think and try to evaluate your paper like an Evaluator. Try to understand that what an evaluator wants in your research paper and automatically you will have your answer.

4. Make blueprints of paper: The outline is the plan or framework that will help you to arrange your thoughts. It will make your paper logical. But remember that all points of your outline must be related to the topic you have chosen.

5. Ask your Guides: If you are having any difficulty in your research, then do not hesitate to share your difficulty to your guide (if you have any). They will surely help you out and resolve your doubts. If you can't clarify what exactly you require for your work then ask the supervisor to help you with the alternative. He might also provide you the list of essential readings.

6. Use of computer is recommended: As you are doing research in the field of Computer Science, then this point is quite obvious.

7. Use right software: Always use good quality software packages. If you are not capable to judge good software then you can lose quality of your paper unknowingly. There are various software programs available to help you, which you can get through Internet.

8. Use the Internet for help: An excellent start for your paper can be by using the Google. It is an excellent search engine, where you can have your doubts resolved. You may also read some answers for the frequent question how to write my research paper or find model research paper. From the internet library you can download books. If you have all required books make important reading selecting and analyzing the specified information. Then put together research paper sketch out.

9. Use and get big pictures: Always use encyclopedias, Wikipedia to get pictures so that you can go into the depth.

10. Bookmarks are useful: When you read any book or magazine, you generally use bookmarks, right! It is a good habit, which helps to not to lose your continuity. You should always use bookmarks while searching on Internet also, which will make your search easier.

11. Revise what you wrote: When you write anything, always read it, summarize it and then finalize it.

12. Make all efforts: Make all efforts to mention what you are going to write in your paper. That means always have a good start. Try to mention everything in introduction, that what is the need of a particular research paper. Polish your work by good skill of writing and always give an evaluator, what he wants.

13. Have backups: When you are going to do any important thing like making research paper, you should always have backup copies of it either in your computer or in paper. This will help you to not to lose any of your important.

14. Produce good diagrams of your own: Always try to include good charts or diagrams in your paper to improve quality. Using several and unnecessary diagrams will degrade the quality of your paper by creating "hotchpotch." So always, try to make and include those diagrams, which are made by your own to improve readability and understandability of your paper.

15. Use of direct quotes: When you do research relevant to literature, history or current affairs then use of quotes become essential but if study is relevant to science then use of quotes is not preferable.

16. Use proper verb tense: Use proper verb tenses in your paper. Use past tense, to present those events that happened. Use present tense to indicate events that are going on. Use future tense to indicate future happening events. Use of improper and wrong tenses will confuse the evaluator. Avoid the sentences that are incomplete.

17. Never use online paper: If you are getting any paper on Internet, then never use it as your research paper because it might be possible that evaluator has already seen it or maybe it is outdated version.

18. **Pick a good study spot:** To do your research studies always try to pick a spot, which is quiet. Every spot is not for studies. Spot that suits you choose it and proceed further.

19. Know what you know: Always try to know, what you know by making objectives. Else, you will be confused and cannot achieve your target.

20. Use good quality grammar: Always use a good quality grammar and use words that will throw positive impact on evaluator. Use of good quality grammar does not mean to use tough words, that for each word the evaluator has to go through dictionary. Do not start sentence with a conjunction. Do not fragment sentences. Eliminate one-word sentences. Ignore passive voice. Do not ever use a big word when a diminutive one would suffice. Verbs have to be in agreement with their subjects. Prepositions are not expressions to finish sentences with. It is incorrect to ever divide an infinitive. Avoid clichés like the disease. Also, always shun irritating alliteration. Use language that is simple and straight forward. put together a neat summary.

21. Arrangement of information: Each section of the main body should start with an opening sentence and there should be a changeover at the end of the section. Give only valid and powerful arguments to your topic. You may also maintain your arguments with records.

22. Never start in last minute: Always start at right time and give enough time to research work. Leaving everything to the last minute will degrade your paper and spoil your work.

23. Multitasking in research is not good: Doing several things at the same time proves bad habit in case of research activity. Research is an area, where everything has a particular time slot. Divide your research work in parts and do particular part in particular time slot.

24. Never copy others' work: Never copy others' work and give it your name because if evaluator has seen it anywhere you will be in trouble.

25. Take proper rest and food: No matter how many hours you spend for your research activity, if you are not taking care of your health then all your efforts will be in vain. For a quality research, study is must, and this can be done by taking proper rest and food.

26. Go for seminars: Attend seminars if the topic is relevant to your research area. Utilize all your resources.

27. Refresh your mind after intervals: Try to give rest to your mind by listening to soft music or by sleeping in intervals. This will also improve your memory.

28. Make colleagues: Always try to make colleagues. No matter how sharper or intelligent you are, if you make colleagues you can have several ideas, which will be helpful for your research.

29. **Think technically:** Always think technically. If anything happens, then search its reasons, its benefits, and demerits.

30. Think and then print: When you will go to print your paper, notice that tables are not be split, headings are not detached from their descriptions, and page sequence is maintained.

31. Adding unnecessary information: Do not add unnecessary information, like, I have used MS Excel to draw graph. Do not add irrelevant and inappropriate material. These all will create superfluous. Foreign terminology and phrases are not apropos. One should NEVER take a broad view. Analogy in script is like feathers on a snake. Not at all use a large word when a very small one would be sufficient. Use words properly, regardless of how others use them. Remove quotations. Puns are for kids, not grunt readers. Amplification is a billion times of inferior quality than sarcasm.

32. Never oversimplify everything: To add material in your research paper, never go for oversimplification. This will definitely irritate the evaluator. Be more or less specific. Also too, by no means, ever use rhythmic redundancies. Contractions aren't essential and shouldn't be there used. Comparisons are as terrible as clichés. Give up ampersands and abbreviations, and so on. Remove commas, that are, not necessary. Parenthetical words however should be together with this in commas. Understatement is all the time the complete best way to put onward earth-shaking thoughts. Give a detailed literary review.

33. Report concluded results: Use concluded results. From raw data, filter the results and then conclude your studies based on measurements and observations taken. Significant figures and appropriate number of decimal places should be used. Parenthetical remarks are prohibitive. Proofread carefully at final stage. In the end give outline to your arguments. Spot out perspectives of further study of this subject. Justify your conclusion by at the bottom of them with sufficient justifications and examples.

34. After conclusion: Once you have concluded your research, the next most important step is to present your findings. Presentation is extremely important as it is the definite medium though which your research is going to be in print to the rest of the crowd. Care should be taken to categorize your thoughts well and present them in a logical and neat manner. A good quality research paper format is essential because it serves to highlight your research paper and bring to light all necessary aspects in your research.

Informal Guidelines of Research Paper Writing

Key points to remember:

- Submit all work in its final form.
- Write your paper in the form, which is presented in the guidelines using the template.
- Please note the criterion for grading the final paper by peer-reviewers.

Final Points:

A purpose of organizing a research paper is to let people to interpret your effort selectively. The journal requires the following sections, submitted in the order listed, each section to start on a new page.

The introduction will be compiled from reference matter and will reflect the design processes or outline of basis that direct you to make study. As you will carry out the process of study, the method and process section will be constructed as like that. The result segment will show related statistics in nearly sequential order and will direct the reviewers next to the similar intellectual paths throughout the data that you took to carry out your study. The discussion section will provide understanding of the data and projections as to the implication of the results. The use of good quality references all through the paper will give the effort trustworthiness by representing an alertness of prior workings.

Writing a research paper is not an easy job no matter how trouble-free the actual research or concept. Practice, excellent preparation, and controlled record keeping are the only means to make straightforward the progression.

General style:

Specific editorial column necessities for compliance of a manuscript will always take over from directions in these general guidelines.

To make a paper clear

· Adhere to recommended page limits

Mistakes to evade

- Insertion a title at the foot of a page with the subsequent text on the next page

- Separating a table/chart or figure - impound each figure/table to a single page
- Submitting a manuscript with pages out of sequence

In every sections of your document

· Use standard writing style including articles ("a", "the," etc.)

· Keep on paying attention on the research topic of the paper

· Use paragraphs to split each significant point (excluding for the abstract)

· Align the primary line of each section

· Present your points in sound order

· Use present tense to report well accepted

· Use past tense to describe specific results

· Shun familiar wording, don't address the reviewer directly, and don't use slang, slang language, or superlatives

· Shun use of extra pictures - include only those figures essential to presenting results

Title Page:

Choose a revealing title. It should be short. It should not have non-standard acronyms or abbreviations. It should not exceed two printed lines. It should include the name(s) and address (es) of all authors.

Abstract:

The summary should be two hundred words or less. It should briefly and clearly explain the key findings reported in the manuscript--must have precise statistics. It should not have abnormal acronyms or abbreviations. It should be logical in itself. Shun citing references at this point.

An abstract is a brief distinct paragraph summary of finished work or work in development. In a minute or less a reviewer can be taught the foundation behind the study, common approach to the problem, relevant results, and significant conclusions or new questions.

Write your summary when your paper is completed because how can you write the summary of anything which is not yet written? Wealth of terminology is very essential in abstract. Yet, use comprehensive sentences and do not let go readability for briefness. You can maintain it succinct by phrasing sentences so that they provide more than lone rationale. The author can at this moment go straight to shortening the outcome. Sum up the study, with the subsequent elements in any summary. Try to maintain the initial two items to no more than one ruling each.

- Reason of the study - theory, overall issue, purpose
- Fundamental goal
- To the point depiction of the research
- Consequences, including definite statistics - if the consequences are quantitative in nature, account quantitative data; results of any numerical analysis should be reported
- Significant conclusions or questions that track from the research(es)

Approach:

- Single section, and succinct
- As a outline of job done, it is always written in past tense
- A conceptual should situate on its own, and not submit to any other part of the paper such as a form or table
- Center on shortening results - bound background information to a verdict or two, if completely necessary
- What you account in an conceptual must be regular with what you reported in the manuscript
- Exact spelling, clearness of sentences and phrases, and appropriate reporting of quantities (proper units, important statistics) are just as significant in an abstract as they are anywhere else

Introduction:

The **Introduction** should "introduce" the manuscript. The reviewer should be presented with sufficient background information to be capable to comprehend and calculate the purpose of your study without having to submit to other works. The basis for the study should be offered. Give most important references but shun difficult to make a comprehensive appraisal of the topic. In the introduction, describe the problem visibly. If the problem is not acknowledged in a logical, reasonable way, the reviewer will have no attention in your result. Speak in common terms about techniques used to explain the problem, if needed, but do not present any particulars about the protocols here. Following approach can create a valuable beginning:

- Explain the value (significance) of the study
- Shield the model - why did you employ this particular system or method? What is its compensation? You strength remark on its appropriateness from a abstract point of vision as well as point out sensible reasons for using it.
- Present a justification. Status your particular theory (es) or aim(s), and describe the logic that led you to choose them.
- Very for a short time explain the tentative propose and how it skilled the declared objectives.

Approach:

- Use past tense except for when referring to recognized facts. After all, the manuscript will be submitted after the entire job is done.
- Sort out your thoughts; manufacture one key point with every section. If you make the four points listed above, you will need a least of four paragraphs.
- Present surroundings information only as desirable in order hold up a situation. The reviewer does not desire to read the whole thing you know about a topic.
- Shape the theory/purpose specifically - do not take a broad view.
- As always, give awareness to spelling, simplicity and correctness of sentences and phrases.

Procedures (Methods and Materials):

This part is supposed to be the easiest to carve if you have good skills. A sound written Procedures segment allows a capable scientist to replacement your results. Present precise information about your supplies. The suppliers and clarity of reagents can be helpful bits of information. Present methods in sequential order but linked methodologies can be grouped as a segment. Be concise when relating the protocols. Attempt for the least amount of information that would permit another capable scientist to spare your outcome but be cautious that vital information is integrated. The use of subheadings is suggested and ought to be synchronized with the results section. When a technique is used that has been well described in another object, mention the specific item describing a way but draw the basic principle while stating the situation. The purpose is to text all particular resources and broad procedures, so that another person may use some or all of the methods in one more study or referee the scientific value of your work. It is not to be a step by step report of the whole thing you did, nor is a methods section a set of orders.

Materials:

- Explain materials individually only if the study is so complex that it saves liberty this way.
- Embrace particular materials, and any tools or provisions that are not frequently found in laboratories.
- Do not take in frequently found.
- If use of a definite type of tools.
- Materials may be reported in a part section or else they may be recognized along with your measures.

Methods:

- Report the method (not particulars of each process that engaged the same methodology)
- Describe the method entirely
- To be succinct, present methods under headings dedicated to specific dealings or groups of measures
- Simplify - details how procedures were completed not how they were exclusively performed on a particular day.
- If well known procedures were used, account the procedure by name, possibly with reference, and that's all.

Approach:

- It is embarrassed or not possible to use vigorous voice when documenting methods with no using first person, which would focus the reviewer's interest on the researcher rather than the job. As a result when script up the methods most authors use third person passive voice.
- Use standard style in this and in every other part of the paper - avoid familiar lists, and use full sentences.

What to keep away from

- Resources and methods are not a set of information.
- Skip all descriptive information and surroundings - save it for the argument.
- Leave out information that is immaterial to a third party.

Results:

The principle of a results segment is to present and demonstrate your conclusion. Create this part a entirely objective details of the outcome, and save all understanding for the discussion.

The page length of this segment is set by the sum and types of data to be reported. Carry on to be to the point, by means of statistics and tables, if suitable, to present consequences most efficiently.You must obviously differentiate material that would usually be incorporated in a study editorial from any unprocessed data or additional appendix matter that would not be available. In fact, such matter should not be submitted at all except requested by the instructor.

Content

- Sum up your conclusion in text and demonstrate them, if suitable, with figures and tables.
- In manuscript, explain each of your consequences, point the reader to remarks that are most appropriate.
- Present a background, such as by describing the question that was addressed by creation an exacting study.
- Explain results of control experiments and comprise remarks that are not accessible in a prescribed figure or table, if appropriate.
- Examine your data, then prepare the analyzed (transformed) data in the form of a figure (graph), table, or in manuscript form.

What to stay away from

- Do not discuss or infer your outcome, report surroundings information, or try to explain anything.
- Not at all, take in raw data or intermediate calculations in a research manuscript.

- Do not present the similar data more than once.
- Manuscript should complement any figures or tables, not duplicate the identical information.
- Never confuse figures with tables - there is a difference.

Approach

- As forever, use past tense when you submit to your results, and put the whole thing in a reasonable order.
- Put figures and tables, appropriately numbered, in order at the end of the report
- If you desire, you may place your figures and tables properly within the text of your results part.

Figures and tables

- If you put figures and tables at the end of the details, make certain that they are visibly distinguished from any attach appendix materials, such as raw facts
- Despite of position, each figure must be numbered one after the other and complete with subtitle
- In spite of position, each table must be titled, numbered one after the other and complete with heading
- All figure and table must be adequately complete that it could situate on its own, divide from text

Discussion:

The Discussion is expected the trickiest segment to write and describe. A lot of papers submitted for journal are discarded based on problems with the Discussion. There is no head of state for how long a argument should be. Position your understanding of the outcome visibly to lead the reviewer through your conclusions, and then finish the paper with a summing up of the implication of the study. The purpose here is to offer an understanding of your results and hold up for all of your conclusions, using facts from your research and generally accepted information, if suitable. The implication of result should be visibly described. Infer your data in the conversation in suitable depth. This means that when you clarify an observable fact you must explain mechanisms that may account for the observation. If your results vary from your prospect, make clear why that may have happened. If your results agree, then explain the theory that the proof supported. It is never suitable to just state that the data approved with prospect, and let it drop at that.

- Make a decision if each premise is supported, discarded, or if you cannot make a conclusion with assurance. Do not just dismiss a study or part of a study as "uncertain."
- Research papers are not acknowledged if the work is imperfect. Draw what conclusions you can based upon the results that you have, and take care of the study as a finished work
- You may propose future guidelines, such as how the experiment might be personalized to accomplish a new idea.
- Give details all of your remarks as much as possible, focus on mechanisms.
- Make a decision if the tentative design sufficiently addressed the theory, and whether or not it was correctly restricted.
- Try to present substitute explanations if sensible alternatives be present.
- One research will not counter an overall question, so maintain the large picture in mind, where do you go next? The best studies unlock new avenues of study. What questions remain?
- Recommendations for detailed papers will offer supplementary suggestions.

Approach:

- When you refer to information, differentiate data generated by your own studies from available information
- Submit to work done by specific persons (including you) in past tense.
- Submit to generally acknowledged facts and main beliefs in present tense.

Administration Rules Listed Before Submitting Your Research Paper to Global Journals Inc. (US)

Please carefully note down following rules and regulation before submitting your Research Paper to Global Journals Inc. (US):

Segment Draft and Final Research Paper: You have to strictly follow the template of research paper. If it is not done your paper may get rejected.

- The **major constraint** is that you must independently make all content, tables, graphs, and facts that are offered in the paper. You must write each part of the paper wholly on your own. The Peer-reviewers need to identify your own perceptive of the concepts in your own terms. NEVER extract straight from any foundation, and never rephrase someone else's analysis.
- Do not give permission to anyone else to "PROOFREAD" your manuscript.
- Methods to avoid Plagiarism is applied by us on every paper, if found guilty, you will be blacklisted by all of our collaborated research groups, your institution will be informed for this and strict legal actions will be taken immediately.)
- To guard yourself and others from possible illegal use please do not permit anyone right to use to your paper and files.

Criterion for Grading a Research Paper (Compilation) by Global Journals Inc. (US)

Please note that following table is only a Grading of "Paper Compilation" and not on "Performed/Stated Research" whose grading solely depends on Individual Assigned Peer Reviewer and Editorial Board Member. These can be available only on request and after decision of Paper. This report will be the property of Global Journals Inc. (US).

Topics	Grades		
	A-B	**C-D**	**E-F**
Abstract	Clear and concise with appropriate content, Correct format. 200 words or below	Unclear summary and no specific data, Incorrect form Above 200 words	No specific data with ambiguous information Above 250 words
Introduction	Containing all background details with clear goal and appropriate details, flow specification, no grammar and spelling mistake, well organized sentence and paragraph, reference cited	Unclear and confusing data, appropriate format, grammar and spelling errors with unorganized matter	Out of place depth and content, hazy format
Methods and Procedures	Clear and to the point with well arranged paragraph, precision and accuracy of facts and figures, well organized subheads	Difficult to comprehend with embarrassed text, too much explanation but completed	Incorrect and unorganized structure with hazy meaning
Result	Well organized, Clear and specific, Correct units with precision, correct data, well structuring of paragraph, no grammar and spelling mistake	Complete and embarrassed text, difficult to comprehend	Irregular format with wrong facts and figures
Discussion	Well organized, meaningful specification, sound conclusion, logical and concise explanation, highly structured paragraph reference cited	Wordy, unclear conclusion, spurious	Conclusion is not cited, unorganized, difficult to comprehend
References	Complete and correct format, well organized	Beside the point, Incomplete	Wrong format and structuring

Index

Q

R

S

T

U

V

W

save our planet

Global Journal of Computer Science and Technology

Visit us on the Web at www.GlobalJournals.org | www.ComputerResearch.org
or email us at helpdesk@globaljournals.org

ISSN 9754350