Journal of Computer Science 5 (1): 39-48, 2009
ISSN 1549-3636

Current State of Component-Oriented Software Development Practice in Malaysia: Towards Identifying Its Potential and Research Areas

[1]Hazleen Aris and [2]Siti Salwah Salim
[1]College of Information Technology, University Tenaga National,
Km. 7, Jalan Kajang-Puchong, 43009 Kajang, Selangor, Malaysia
[2]Faculty of Computer Science and Information Technology, University of Malaya,
Lembah Pantai, 50603 Kuala Lumpur, Malaysia

Abstract: Problem statement: A study was conducted to investigate the current state of software development practice in Malaysia with regard to Component Oriented Software Development (COSD). The objectives of the study were to identify its potential application in Malaysia and to determine research areas that can be explored further to promote its application. **Approach:** Information required for the study was obtained through a survey, questionnaires were distributed to the software developers who are working at various software development companies in Malaysia. Software developers were generally divided into two categories; component users and non-component users. Information on current practices in software development, current practices of the component users, problems faced by the component users in applying COSD and reasons that hinder the non-component users from applying COSD was sought. **Results:** Results showed that current practice of software development in Malaysia is inclined towards software reuse, which is in line with the aim of COSD. Even though the component users face problems in applying COSD, they still believe that COSD is a better approach to develop software and majority of them will continue to apply COSD. The non-component users also believe that COSD is a better approach to develop software and majority of them will apply COSD in their future software development projects. **Conclusion:** Study concluded that the potential of COSD application amongst the software developers in Malaysia is high. List of COSD research areas formulated from the study can serve as a basis for the researchers to pursue research in this area that will further increase the potential.

Key words: Component Oriented Software Development (COSD), COSD state, COSD application

INTRODUCTION

The idea of Component-Oriented Software Development (COSD) is as old as the software engineering itself. Its inception dates back to the late 1960's. Since then, it has become one of the central research topics in the field of software engineering and its evolution has taken place in many different forms and from various aspects. The main motivation behind these research is the list of advantages expected to be brought about by the successful implementation of COSD and the success stories of components reuse from other engineering fields, particularly in mechanical and electrical engineering. Despite the presence of obstacles along the path in adopting COSD, researchers are still optimist that these obstacles, or rather, challenges, can be overcome as COSD matures and 'disappears' i.e., no longer become noteworthy[1]. Hence, research on COSD is becoming more and more intense with each addressing one or more challenges imposed.

One of these areas is the study on the current state of COSD application in the software development industries. Review made on related literatures discovered at least two other research that study the state of COSD application. The first research studied the situation at Kingston city of Jamaica[2]. This study, which was done based on the response from eight prominent software development companies in Kingston, covered the level of components reuse in software development, the quality of software systems created with components reuse,

Corresponding Author: Hazleen Aris, College of Information Technology, University Tenaga National, Km. 7, Jalan Kajang-Puchong, 43009, Kajang, Selangor, Malaysia Tel: +603-89212368 Fax: +603-89212161

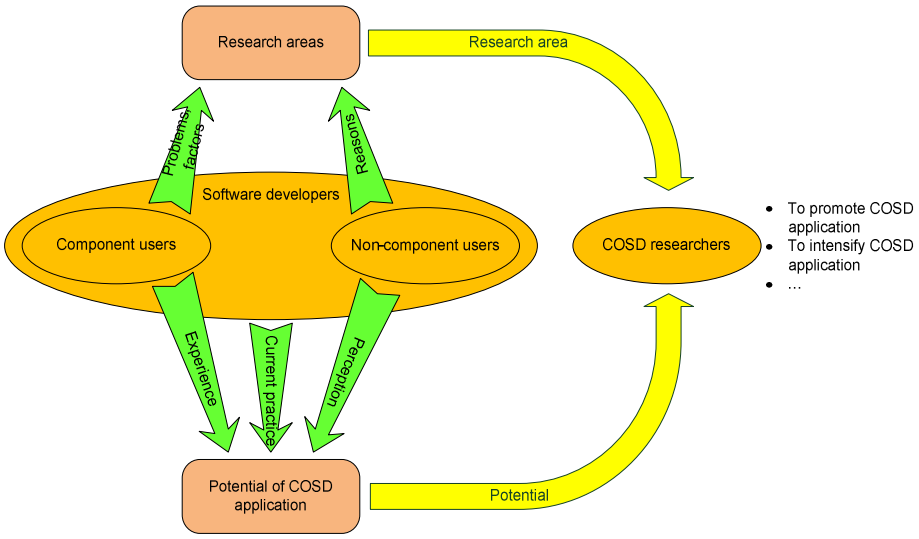

Fig. 1: Survey model for the study

the average number of software systems created per year and the cost associated with components reuse. In particular, it concluded that all of the companies involved in the case study have experienced the benefits of COSD and that the main success factor of COSD was attributed to the existence of good components repositories. The second research performed similar study covering three European countries; Norway, Italy and Germany[3]. This study focused on the process models used to develop software using Off-The-Shelf (OTS) components and the selection process of the OTS components. It concluded that the process models used are typically variations of well-known traditional process models mixed with OTS specific activities and the selections of OTS components are based on two processes; familiarity-based and internet search-based.

The situation in Malaysia, however, is not quite known as there are not many research done thus far to study the current state of COSD application in software development in this country. This has triggered us to perform similar study that aims at finding out the current state of COSD application amongst the software developers in Malaysia. Questionnaires entitled 'A Survey to Investigate the Current State of the Application of Component-oriented Software Development (COSD) amongst the Software Developers in Malaysia, were distributed to the software developers who are working at various software development companies across Malaysia. Two main objectives of the survey are:

- To determine the potential of COSD application in Malaysia
- To identify the appropriate research areas in COSD that should be addressed in order to promote and enhance its application

Figure 1 shows the survey model used to guide the study. As can be seen from the model, software developers in this study are broadly categorized into component users and non-component users. Component users are those who use components in their software development projects and non-component users are those who do not use components in their software development projects. To meet the first objective, the following information is sought:

- Current practice in software development from both component users and non-component users
- Experience in applying COSD from the component users
- Perception towards COSD from the non-component users

To meet the second objective of the study, the following information is sought from the component users:

- Problems faced in applying COSD
- Factors believed to play significant roles towards the success of COSD

From the non-component users, information on their reasons for not applying COSD in software development is sought. The questionnaire was therefore designed to contain questions that can capture these information.

MATERIALS AND METHODS

In the next three paragraphs, the design of the questions that make up the questionnaire, population sampling of the target respondents and data collection method are explained.

Questionnaire design: The questionnaire comprises a total of 19 questions that are divided into four main sections; section A, section B, section C and section D. Section A contains three questions (Q1-Q3) asking for the background information of the respondents and section B contains five questions (Q4-Q8) to find out the respondents' current practice in developing software. The last question in section B (Q8) asked about whether or not the respondent is using components in developing software. Based on the respondents' answers to this question, they would have to either proceed to section C or section D. Section C of the questionnaire was targeting at the component users, i.e., those who use components in their software development. This section, which contains eight questions (Q9-Q16), was looking for information on the nature of components used, problems in using them and ways to improve their use. Finally, section D that contains three questions (Q17-Q19) was meant for the non-component users, i.e., those who do not use components in their software development. The questions asked for information on the perception on COSD and possibility of using it in future software development projects.

Population determination: The questionnaires were distributed to the software developers in the Multimedia Super Corridor (MSC) status companies, which are clustered mainly at these four cybercities; Kuala Lumpur city centre, Technology Park Malaysia, UPM-MTDC and Cyberjaya[4]. Our main source of information to estimate the total number of software development companies in Malaysia is the Multimedia Super Corridor (MSC) portal[5] where a list of Information and Communication Technology (ICT) related companies with the MSC status from a number of sectors is made publicly available. At present, there is a total of 1,511 ICT related companies being granted the MSC status as shown according to their respective sector, in Table 1.

Data collection: Prior to the actual questionnaire distribution, a pilot test was performed using paper-based questionnaire involving a total of eleven software developers, which were chosen based on convenient sampling. The pilot test was conducted with the aim of identifying the possible problems or confusions in understanding the questions in the questionnaire.

Table 1: ICT related companies with MSC status

Sector	No. of companies
Software development	803
Creative multimedia	149
Support services	143
Internet-based business	183
Hardware design	133
Shared services and outsourcing	100

In the actual questionnaires distribution, a total of 400 questionnaires were distributed to 400 software development companies, which were randomly selected from the list of MSC status companies under the software development sector. Some of the questionnaires did not reach their target recipients due to:

- Invalid addresses-the address published in the MSC portal is no longer used and so do the contact numbers
- Overseas addresses-the development work for the companies are actually done overseas
- Branch company-the companies are branches of larger companies and therefore we only sent questionnaires to the parent companies

In answering the questionnaires, the respondents were given the choice to return the paper-based questionnaires or to submit the online version of the questionnaires.

When the information gathering exercise concluded in October 2007, a total of 104 responses were received, making up 26% response rate. Of these, 31 respondents returned the paper-based questionnaires and the balance of 73 respondents answered the web-based questionnaires. The main reason given by those who refused to participate was time constraint. Other than this time constraint, quite a number of the companies' representatives contacted during follow up to non-response said that they do not have any software developers in the companies i.e., they are just software resellers.

RESULTS

From the total of 104 responses received, three paper-based responses were deemed invalid for analysis due to incomplete information. Thus, the following presentation of results is based on 101 valid responses.

Demographics of respondents: Table 2 shows the information on the respondents' job nature. System analyst, system designer and programmer are positions that are directly related to software development.

Table 2: Respondents' job nature (Q1)

Job Nature	Frequency	Percentage (%)
System analyst	25	24.75
System designer	2	1.98
Programmer	36	35.64
Subtotal	63	62.37
Others	38	37.62

Table 3: Respondents experience in the position (Q2)

Duration in the Position	Frequency	Percentage (%)
More than 5 years	39	38.61
Between 2 and 5 years	29	28.71
Subtotal	68	67.32
Less than 2 years	33	32.67

Table 4: Users of the software produced (Q3)

Software product user	Frequency	Percentage (%)
Both	43	42.57
External	27	26.73
Subtotal	70	69.30
Internal	30	30.69

Table 5: Programming languages used by developers (Q4)

Languages	Frequency	Percentage (%)
C	26	25.74
C++	28	27.72
Java	42	41.58
Visual Basic	41	40.59
Others	56	55.45

Therefore, from Table 2, 62.37% respondents are holding positions that are directly related to software development. The balance of 37.62% respondents who chose 'Other' range from software engineers to technical information technology supports. From Table 3, it can be seen that 67.32% of the respondents have more than two years experience in their position. From Table 4, we can see that nearly three quarter of the respondents (69.30%) are involved in developing software for other companies.

Current practice: On the current practice, the questionnaire aimed at obtaining information on the programming languages used, software development methodology used, amount of code reused from the previous projects, familiarity and understanding of the term component. Information on the portion of the software developers who are using components in their software development projects is also obtained. Table 5 shows the distribution of the programming languages used by developers. 'Others' languages used by the developers range from COBOL to a complete development framework. Shown in bold is the programming language used by most of the software developers, i.e., Java.

Table 6: Methodology applied in software development (Q5)

Methodology	Frequency	Percentage (%)
Conventional	53	52.48
Object-oriented	66	65.35
Component-oriented	30	29.70
Others	9	8.91

Table 7: Amount of codes reused from previous projects (Q6)

Reuse amount	Frequency	Percentage (%)
100%	5	4.95
Around 80%	22	21.78
Around 50%	49	48.51
Less than 20%	20	19.80
None	5	4.95
Reusers	96	95.05
Reusers of > 50%	76	79.17

Table 8: Familiarity versus the use of component

	Familiar with the term component	Not familiar with the term component
Use component	49	9
Not using component	16	27

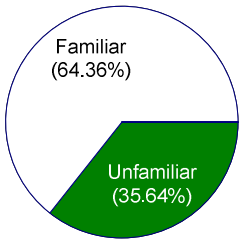

Fig. 2: Familiarity of the term component (Q7)

From Table 6, it can be seen that the methodology applied, as claimed by most software developers is object-oriented (65.35%), followed by conventional methodology (52.48%) in the second place and component-oriented (29.70%) in the third. Table 7 shows 95.05% of the software developers reuse code from previous projects with majority of them reuse around 50% of the code. The percentage of reusers who reuse more than 50% of code from previous projects is 79.17%.

When asked whether or not they are familiar with the term component, 65 (64.36%) of the respondents are familiar with it and the balance of 36 (35.64%) respondents are not as shown in Fig. 2.

In response to the question that asks about components usage, 58 (57.43%) respondents admit that they use components in their software development and the balance of 43 (42.57%) say that they do not use components as shown in Fig. 3. A cross tabulation analysis shows that out of 65 respondents who are familiar with the term component, only 49 are actually using it in their software development projects as can be seen from Table 8. This represents 48.51% of the total respondents.

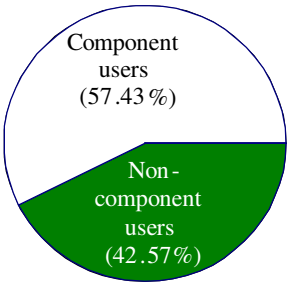

Fig. 3: Percentage of component users and non-component users (Q8)

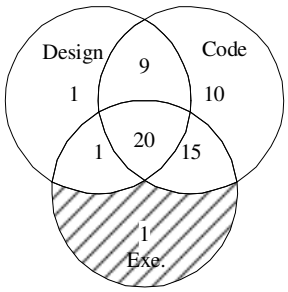

Fig. 4: Types of components used (Q10)

Table 9: Adherence to component requirements (Q11)

	Independent of any vendors	Dependent on vendors
Available in executable form	28	10
Not available in executable form	14	6

Table 10: Self developed versus third party components (Q9)

	Using self developed components	Not using self developed components
Using third party components	20	17
Not using third party components	20	1

Next, the analysis done on the data gathered from the component users is presented, which is able to give more information on the application of COSD amongst Malaysian software developers.

Component users: From the total of 58 respondents who use components in their software development projects, 28 respondents, representing 48.28% of the component users claimed that the components used met both requirements of a component as stated by Szyperski[6]. This cross tabulation result is shown in Table 9. The questionnaire then further asked a few more questions on sources of components and forms of components used to investigate this claim. Result shows that only 20 component users use a mixture of self developed and third party components as shown in Table 10. This further reduces the number of component users using components that meet both requirements of a component to only 34.48%. However, only 1 (1.72%) component user use component solely in executable form as can be seen from the shaded area in the Venn diagram shown in Fig. 4.

Next, the questionnaire intended to find out the problems encountered by the component users. These problems, as shown abbreviated in Fig. 5, are:

- Lack of tools to support the development process (Tools availability)

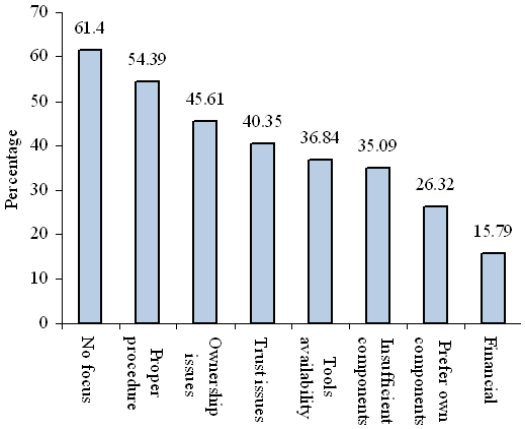

Fig. 5: Problems faced in using components (Q12)

- Insufficient components to be used and reused (Insufficient component)
- No proper procedures for developing, depositing and retrieving components from the repository (Proper procedure)
- Focus not given to COSD (No focus)
- Insufficient funding in shifting to COSD (Financial)
- Trust in using components developed by other people (Trust issues)
- Prefer to develop the whole application afresh (Prefer own component)
- Ownership issues of the shared components (Ownership issues)

Figure 5 also shows the feedback received from the developers on the problem that they faced in using components in percentage form. Of the 36.84% software developers who claimed that lack of tools as one of their problems in using components, the questionnaire further asked about the type of tools needed and the response is shown in Fig. 6. From Fig. 6, it shows that the most needed tools,

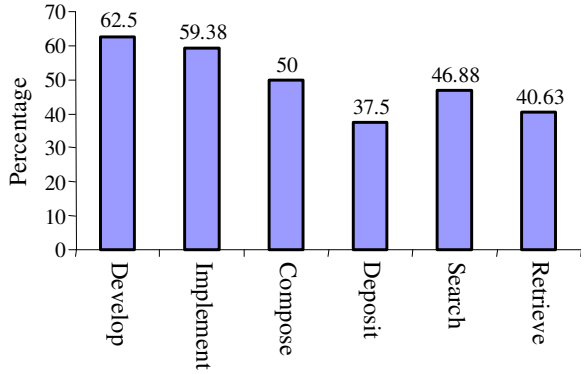

Fig. 6: Types of tools needed to support COSD (Q13)

in descending priority order, are tools to develop (62.50%), implement (59.38%) and compose (50.00%) components respectively. The component users were also asked on the factors that they think play important roles towards successful implementation of COSD. These factors are:

- Availability of tools to support COSD (Tools availability)
- Practical hands-on training on the application of COSD (Training)
- Sufficient knowledge on component technology (Knowledge on COSD)
- Strong support from the management to shift to COSD (Management support)
- Strong financial support or funding to start up COSD (Financial)
- Willingness to learn and adopt new skills and technology (Willingness to learn)
- Proper procedures on components ownership, transfer et cetera in place (Proper procedure)
- Willingness to share components with other developers (Willingness to share)

The result also varies as shown in Fig. 7.

The advantages of COSD as stated in many literatures are higher software product quality, lower production time and lower production cost. For those who have been applying COSD in their software development projects, these benefits may have been materialized. For others, they remain a theory. Our interest here is to find out the percentage of component users who see these advantages from their experience in using components. Since quality is somewhat intangible, it is further refined into three contributing attributes; number of errors found, testing time and number of complaints from users.

Table 11: Opinion on COSD as a better way to develop software (Q14)

Opinion	Frequency	Percentage (%)
Agree	40	94.83
Disagree	3	5.17

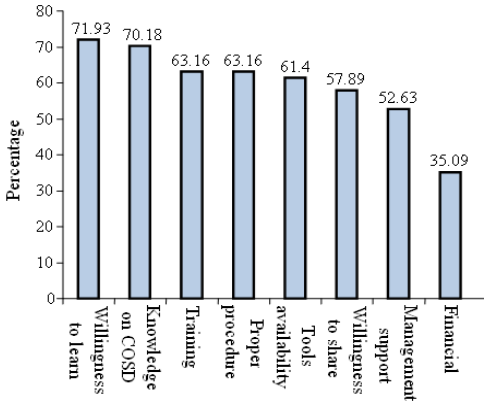

Fig. 7: Factors towards successful implementation of COSD (Q16)

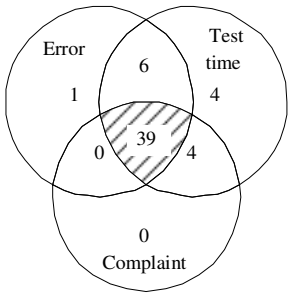

Fig. 8: Seen quality attributes of COSD (Q15 a))

Component users are regarded as to have seen the increase in quality of the software produced when they agree to all these attributes, as shown shaded in the Venn diagram of Fig. 8. Therefore, 39 (67.24%) component users actually see or experience an increase in the quality of the software produced when using components.

Out of these 39 component users, more than half of them (74.36%) also agree to the reduced time to market the software and lower cost in producing them as shown in the Venn diagram of Fig. 9. As such, it can be concluded that a total of 29 (50.00%) component users actually see the oft-mentioned advantages of COSD based on their experience.

Interestingly, despite all the problems faced by these component users, 55 (94.83%) out of 58 of them still believe that COSD is a better way to develop software as shown in Table 11. Next, the analysis done on the feedback received from the non-component users is presented.

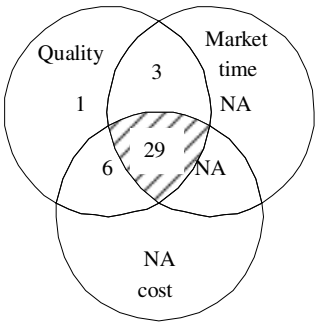

Fig. 9: Seen advantages of COSD (Q15)

Table 12: Chances of using component in future software development projects (Q18)

Chances	Frequency	Percentage (%)
Will use	39	90.70
Will not use	4	9.30

Non-component users: 43 out of 101 respondents, which accounts for 42.57% of the total respondents do not use components in their software development projects. Various reasons are given for not using components. These are:

- Lack of knowledge about components (Knowledge on COSD)
- Not exposed to the component technology (Exposure to COSD)
- Lack of available supporting tools (Tools availability)
- Lack of support from the management (Management support)
- Insufficient funding to shift to COSD (Financial)
- Do not feel the need to change the current practice (Need for change)
- Not willing to change the current development practice (Willingness to change)
- Not willing to learn new technology (Willingness to learn)
- COSD is not the common development practice in Malaysia (Uncommon practice)

The top three reasons for not using components are exposure to COSD, knowledge on COSD and tools availability as shown in Fig. 10. When asked about their willingness to use components in future software development projects, if given the opportunity to do so, 90.70% of them will adopt COSD in the future if given the chance. Furthermore, 93.02% of the non-component users agree that COSD is the way to go in developing software as shown in Table 12 and 13 respectively.

Table 13: Opinion in COSD as the way to go in developing software (Q19)

Opinion	Frequency	Percentage
Agree	40	93.02%
Disagree	3	6.98%

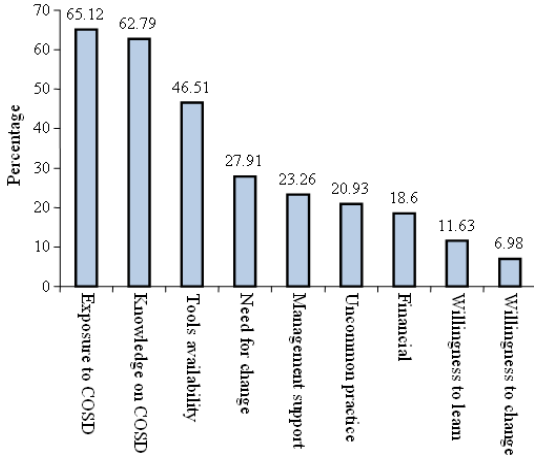

Fig. 10: Reasons for not using components (Q17)

DISCUSSION

Next, analyses are performed to interpret the results and tie them back to objectives of this survey as shown in the survey model of Fig. 1.

Determining the potential: The potential of COSD application can be determined from the responses given by the software developers in general, from the component users and from the non-component users as previously shown in Fig. 1. Therefore, the subsequent three paragraphs will discuss and conclude about the potential from the viewpoints of the three groups of respondents respectively.

From the current software development practice of all software developers, it can be concluded that the inclination is towards software reuse. This is showed by the types of programming languages used, which are mainly object-oriented programming languages; the development methodology applied, which are mainly object-oriented; and the amount of code being reused where majority of the software developers reuse code from previous projects with the mod of reuse amount is around 50%. Almost 80% of those who reuse code from previous projects code reuse more than 50% of the code. COSD is designed to support reuse and this inclination is in line with its objectives. Therefore, it can be concluded that from the viewpoint of the software developers in general, the potential of COSD application is high. The fact that majority of the

software developers are already familiar with the term component further increases this potential.

Even though problems are faced by the component users while using components, 94.83% of the existing component users still believe that COSD is a better way to develop software. This belief is most likely supported by the fact that half of them experienced all the advantages of COSD while using it. This means that the possibility of the component users to continue using this development approach is there. Therefore, from the viewpoint of the current component users, it can be concluded that the potential of COSD to be applied in their software development projects is also high.

From the side of the non-component users, the prospect is also very encouraging. 93.02% of the non-component users agree that COSD is the way to go in developing software despite not applying them in their software development projects, with 90.70% of them will apply the approach if given the chance to do so. Therefore, we can also conclude that the potential of COSD to be applied by the non-component users is also high. With these, we conclude that the potential of COSD application in Malaysia is high.

On top of the potential for COSD application, outcomes from the survey are also able to discover the research potential in the area of COSD. From the total of 58 component users, only 28 (27.72%) of them admit that the components used meet both requirements of being independent of any vendors and available in executable form, two requirements of ideal components as stated by Szyperski[6]. This claim, when further investigated shows that only 34.48% of the component users use a mixture of self developed and third party components, i.e., an indication of being independent of vendors. Furthermore, majority of the component users use components in a mixture of design, code and executable forms. There is only 1 (0.99%) component user who uses components solely in executable form, i.e., an indication of meeting the second requirement of an ideal component. Thus, it can be concluded that the current COSD practice is still far from maturity and hence, provides spacious room for research opportunities in this area. Therefore, the possible research areas, which were identified from the survey results, will be discussed next.

Identifying research areas: From the analysis shown in Fig. 5, 7 and 10, it can be seen that the priority order of problems, factors and reasons affecting the use of COSD vary between the component users and non-component users. To the component users, the top three problems faced are:

- Focus not given to COSD (No focus)
- No proper procedures for developing, depositing and retrieving components from the repository (Proper procedure) Ownership issues of the shared components (Ownership issues)

As for the factors affecting the use of COSD, they priorities on:

- Willingness to learn and adopt new skills and technology (willingness to learn)
- Sufficient knowledge on component technology (knowledge on COSD)
- Practical hands-on training on the application of COSD (training)

For the non-component users on the other hand, the first three reasons given by them are:

- Not exposed to the component technology (Exposure to COSD)
- Lack of knowledge about components (knowledge on COSD)
- Lack of available supporting tools (tools availability)

Because of this differing priority, it is therefore not feasible to address a single problem, factor or reason that will give the most impact on the application of COSD to both component users and non-component users alike. Therefore, it is suggested that each problem, factor or reason is treated equally by venturing into appropriate research area. For this purpose, four possible research areas, labelled S_1 to S_4, are proposed to address the problems, factors and reasons. Overlapping problems, factors and reasons are merged in the process of formulating the research areas. Details of the merging is explained in Aris and Salim[7]. These research areas are categorized into primary and secondary as shown in Fig. 11. Primary research areas can be initiated to directly solve a specific problem and factor under it. Secondary research areas depend on the outcomes of primary research areas and cannot be directly initiated.

In the primary research areas category, three research areas are identified; S_1, S_2 and S_3. S_1 is concerned with the provision of trainings and workshops related to the application of COSD in software development. Issues that are directly influenced by this solution can be divided into two as shown separated by dashed line in Fig. 11. Issues above the dashed line are those that directly affect the software developers.

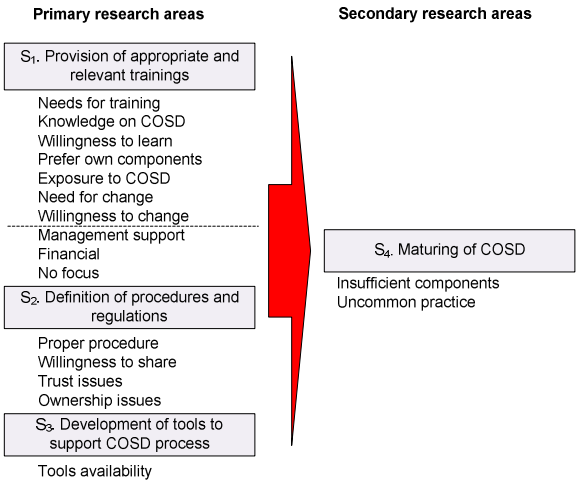

Fig. 11: Proposed research areas to address problems facing and factors affecting the application

Meanwhile, issues below the dashed line are those that affect the management of a software company. For the software developers, two categories of trainings and workshops can be further identified; soft-skill training and hard-skill training. Soft-skill trainings focus on information dissemination to instil awareness on the benefits of using components while hard-skill trainings that focus on the use of tools, techniques and model to facilitate the development of software using COSD approach.

S_2 is concerned with definition of procedures and regulations pertaining to the use of components. This includes the procedures to store components into the repositories that may include the issues of labeling and tagging a component, procedures to retrieve components from the repositories that may include the issues of acknowledgment, royalty, copyrights and the extent of which a component can be modified. Regulations that govern the use of third party components, including accountability issues, should also be defined. Proper procedures and regulations in place help to resolve issues on the lack of proper procedure, willingness to share, trust and ownership issues as shown in Fig. 11.

The third primary research area, S_3, is on the development of tools to support COSD process is the research area where most of the current research on COSD is focusing on[8]. This solution concerns with the production of tools that can be used to facilitate the software developers in developing software using COSD approach. The tools can support individual or a group of processes in COSD. Tools can also include framework or models that can be applied by the software developers to assist the development work.

Research results from primary research areas will contribute to the maturing of COSD, S_4 that will solve the problems and factors of insufficient components and uncommon practice. However, it is important to note here that these research areas are not necessarily mutually exclusive of one another. For example, proper procedures and regulations are needed in governing the exchange of components (S_2). However, this can only be achieved with the exposure of COSD being made through provision of appropriate and relevant trainings (S_1).

Threats to validity: While every care has been taken to ensure the reliability of the information gathered, its representativeness cannot be 100% guaranteed as the data are obtained from sampled population. However, the following measures have been taken to mitigate the possible threats to data validity[9].

On construct validity, a pilot test was performed prior to the actual questionnaire distribution to ensure that the software developers share common understanding on the terms used in the questionnaire. As a result, a number of questionnaires were rearranged and rephrased for clarity.

On external validity, the data collected indicated that most of the respondents are experienced enough for us to rely on the accuracy of the information given, with majority having more than two years experience in their job. Most of the respondents also develop software for both internal and external use, which indicates that the software development practice applied is not only confined to their companies but general enough to represent software development practice in Malaysia. Needless to mention, most of the respondents' job nature is also directly related to software development.

On internal validity, as can be seen in this article, only basic descriptive statistics is used in analyzing the results, which are derived directly from the raw data gathered. Finally, on reliability, detailed descriptions on the survey method and questionnaire structure have been included.

CONCLUSION

Knowing and understanding the current state of COSD application in the software development industries are important in determining research areas in COSD that will help promoting its use by the software developers. In this study, the results of a study done on investigating the current state of COSD application amongst Malaysian software developers are presented. The study was accomplished through questionnaire distribution to the software developers

working in various software development companies in Malaysia. Analysis of the responses received shows that the opportunity of COSD application in future software development amongst Malaysian software developers is high. On top of the opportunity, this study also discovers a number of problems associated with its application, which if not treated, would become hindrances to the application of COSD, despite its high prospect. Therefore, combined with the factors affecting the application of COSD, which is another finding from this study, these problems are turned into possible research areas that can be explored in order to encourage its application amongst Malaysian software developers.

REFERENCES

1. Prieto-Diaz, R., 1994. The disappearance of software reuse. Proceeding of the 3rd International Conference on Software Reuse, Nov. 1-4, IEEE Computer Society, USA., pp: 255. DOI: 10.1109/ICSR.1994.365792
2. Pyne, R., S. McNamarah, M. Bernard, D. Hines, G. Lawrence and D. Barton, 2005. An evaluation on the state of component-based software engineering in Jamaica. Proceeding of the IEEE Southeast Conference, Apr. 8-10, IEEE Computer Society, USA., pp: 570-575. DOI: 10.1109/SECON.2005.1423307
3. Li, J., M. Torchiano, R. Conradi, O.P.N. Slyngstad and C. Bunse, 2006. A state-of-the-practice survey of off-the-shelf component-based development processes. Lecture Notes Comput. Sci., 4039: 16-28. http://www.springerlink.com/content/184316177150k2qn/
4. Seta, F., T. Onishi and T. Kidokoro, 2001. Study about locational tendency of IT companies in city centers and suburbs-case study of Malaysia. Proceeding of the International Symposium on Urban Planning, (ISUP'01), Taejong, Korea, pp: 257-266. http://www.regionalplanning.net/seta/study/0108myit.pdf
5. MDec, 2007. List of MSC Status Companies. http://www.mscmalaysia.my/topic/Company+Directory
6. Szyperski, C., 2002. Component Software Beyond Object-Oriented Programming. 2nd Edn., Pearson Education Limited, London, ISBN: 0-201-74572-0.
7. Aris, H. and S.S. Salim, 2008. Issues on the application of component-oriented software development: Formulation of research areas. Inform. Technol. J., 7: 1149-1155. http://www.scialert.net/pdfs/itj/2008/1149-1155.pdf
8. Crnkovic, I. and M. Larsson, 2002. Building Reliable Component-Based Software Systems. 1st Edn., Artech House Publishers, ISBN: 1-58053-327-2, pp: 454.
9. Easterbrook, J., M.A. Storey and D. Damian, 2007. Selecting Empirical Methods for Software Engineering. In: Guide to Advanced Empirical Software Engineering, Shull, F. and D.I.K. Sjoberg (Eds.). Springer, London, ISBN: 978-1-84800-044-5, pp: 285-311.