Maturity Model for Analyzing Risks in Cots Components

S.Nirenjena
Teaching Fellow,
Department of Information Technology,
Madras Institute of Technology Chennai
nirenjena25@gmail.com

N.Sakthipriya
Assistant Professor,
Department of CSE,
Bharath University, Chennai
priya.sakthi43@gmail.com

P.Sathiyanarayanan
Assistant Professor
Department of CSE
Manakula Vinayagar Institute of Technology, Puducherry,
Sathiyanarayanan89@gmail.com

Abstract:- **Commercial-off the shelf components has gained popularity in many IT industries due to increase in its functionality and decrease in development cost. They are obtained as black box components by the third party vendors therefore various risks are involved while integrating COTS software into our system. Risks are nothing but probability of occurrence, so the industry purchasing the COTS has to be aware of the risks involved in it due to their investments on it. This paper provides a risk maturity model which analyses risks involved in requirements, incoming and outgoing interactions, components criticality, usability and security. Metrics are included to evaluate the risk percentage of the components. Checklists is also present where thirty to forty questions along with three options each ,they help in evaluating the risks involved is high, medium or low. This model will surely help the end users in evaluating the risk factors depending on their domain.**

I. INTRODUCTION

Risks in COTS can be classified in both theoretical and industrial perspective. Much researchprovides solution to risks by providing mitigation strategies for each development stages of the software. They can provide solution only upto a certain extent and cannot be applied to all domains therefore we have to analyze the risks based on industrial performance of COTS. Theoretical classification of risks involves performance, cost, schedule and support. They have their sub-classification too.

Classification of risks

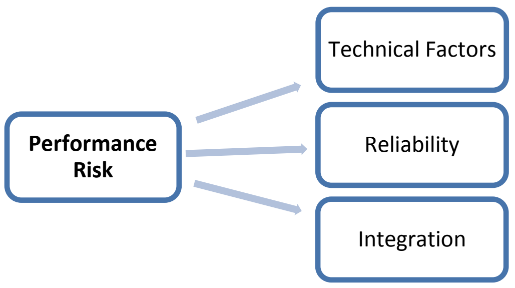


Fig 1(a): Performance risk classification

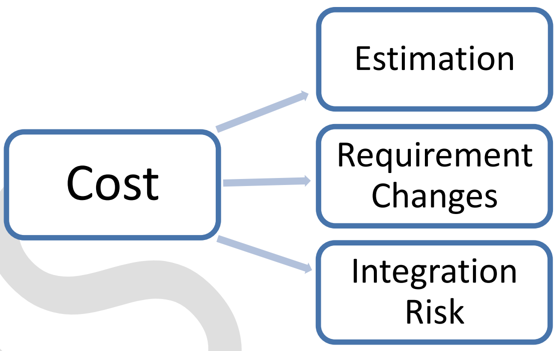


Fig 1(b): Risks involved due to cost

Practical perspective is nothing but performance of COTS in industries. Saved development time and money will be wasted if performance evaluation of the system is varied. The following are the areas where the end users should concentrate:

- Integration of API's
- Data synchronization
- Disruptions
- Security
- Prototypes
- Customizations

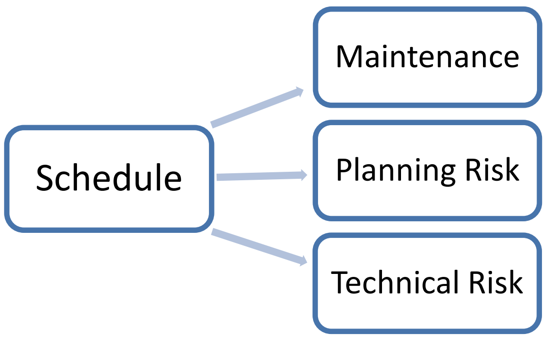


Fig 1(c): Risks involved in schedule

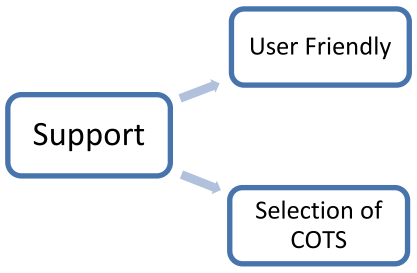
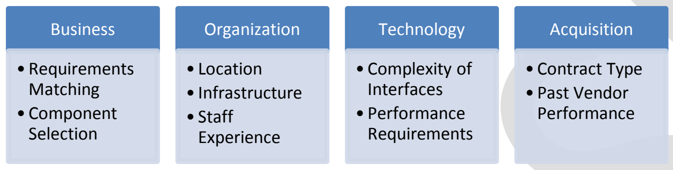


Fig 1(d): Risks involved in support

Performance experts are necessary to address potential risks impacting the performance of the system.The maturity model developed concentrates on both above mentioned perspectives. By following the steps in the maturity model we can achieve a qualitative insight. The checklist questions involved concentrates on business purposes, organization factors, technology, acquisition and implementation. The tabulation below will list out the sub factors of each mentioned factors.



Business	Organization	Technology	Acquisition
• Requirements Matching • Component Selection	• Location • Infrastructure • Staff Experience	• Complexity of Interfaces • Performance Requirements	• Contract Type • Past Vendor Performance

Table 1: Risk Profile

Business purpose drives the organization to consider COTS solution and fit of the requirements with COTS packages. Organization determines the appropriateness of a specific COTS solution based on the sub factors mentioned in table1. Implementation of COTS drives the delivery of COTS solution within the organization and not limited to cost, schedule, testing and managing organizational change. The questions in the checklist include three options each where risk assessment is classified as high, medium or low.

II. RELATED WORK

Existing research work includes verification and validation of various COTS products. Metrics and models were proposed to determine the cost and quality of the components. To evaluate the quality of COTS many models were introduced but risk driven factors were not analyzed. The proposed quality model is the combination of all the existing metrics and models evaluating criticality and functionality of COTS. Validation of new thesis on off-the shelf component based development[12] was proposed by Jingyue Li, ReiderConradi and their team. In this study they have included seven structured interviews and nearly six theses were concluded. The survey was done in three countries Norway, Italy and Germany from 133 component based projects. The theses supported the following factors:

- Open source Software components were mainly used without modification in practice.
- Custom code provides additional functionality.
- COTS users managed to get required changes from the vendors

Similarly there were certain mismatches too:

- Standard mismatches are more than architectural mismatches.
- Components were selected based on architectural compliance instead of functional completeness.

The thesis was based on following factors:

- Open source software is often used as closed source.
- Integration problems result from lack of compliance with standards.
- Custom code mainly provides additional functions.
- Architecture is more important than requirement for product selection.
- Integration influences the vendors on product evolution whenever possible.

PradeepTowar and Nasib proposed a new X component based model. Verification and validation are done here to check the software against its specifications. Component based software engineering puts high demands to ensure the functionality and quality of software with verification and validation. The main characteristic of this model is to enhance reusability in which software is developed by building reusable and testable components.

Error propagation analysis for COTS systems by Jeffrey Voas[11], gives us an clear idea about where and how risk is arised in COTS. The model proposed concentrates on those factors too. The semantic dependencies between software components are so complex. The technique introduced forcefully corrupts the information that flows between components and observer impact due to corruption. Through that we can isolate the components that cannot tolerate the failure of other components. Research by Barey Boehm deals with requirements that handle IKIWISI(I'll know it when I see it").The research provides four guiding principles that will help the developers to fit our situation.They ensurevalue driven requirements[3]:

- Shared vision requirements
- Change driven requirements
- Risk driven requirements

Risk driven requirements is when anything is less than specifying interface requirements between the software and specialized hardware device or between the software in two or large integrated systems risks. If the interface are ambiguous or undefined interface mismatches causing serious operational problem or massive rework and delays during integration.

Chongwon Lee, Byunjeong Lee and ChisuWu[2], provided a guideline to determine quality checklists priorities based on evaluation records of software products. The research suggests the major way of quality evaluation and certification requires dynamic behavior testing. External characteristics of COTS software is considered thoroughly.

Risk concern and quality will limit the application of COTS to non-critical applications. Metrics and models for cost and quality of component based software were evaluated by SahraSedigh-Ali, ArifGhafoor and Raymond A.Paul[7]. The metric helped to eliminate various sources of risk.

Risk reduction takes many forms like component wrappers or middleware, replacing components relaxing system requirements, legal disclaimers for certain-prone software features.R.Moraes, J.Durales, R.Barbosa, E.Martins and H.Madeira[5], conducted a risk assessment and comparison using software fault injection. This research focused on practical approach to assess the risk of the COTS component. It is helpful in assessing the risks in individual modules.The proposed approach uses injection of realistic software faults to analyze the impact of possible component failures and complexity metrics is used to estimate the probability of residual defects in software components. This method can only measure the impact of faults so we have to analyze the estimation of the probability of the fault in the target component.The proposed approach measures experimentally the risk of using a given component C in a system S by the following equation

$$\boldsymbol{Risk_c = prob(f_c) * cost(f_c)}$$

Where,

- prob(f_c) represents the likelihood of the existence of residual software faults in components C
- cost(f_c) represents the impact of the activation of faults in component C measured by software fault injection.

Ned Chapin proposed a entropy metric for systems with COTS software. This metric helps is validation of components like COTS software, Component based software, reused software and object oriented software. He proposed a L-Metric in order to measure the complexity of the interaction of the above components.The research focuses on four factors:

i. The amount of COTS software incorporated into the system
ii. The choice of maintainer
iii. Extent of customization of COTS components such as by wrappers and in-components changes.
iv. The effects of technology changes

The metric helps in assessing at the stage of software modification, the changes in the system complexity affecting maintainability for systems with component software such as COTS.

Arun Sharma, Rajesh Kumar and P.S.Grover[1],proposed empirical evaluation and validation of interface complexity metrics for software components. Due to the black box nature of COTS complexity of software components is more crucial for component based systems. The interface is the only source of information to known about the components therefore interface complexity metric derived will be helpful in evaluating the risk factors due to interaction. A correlation analysis between proposed metric and several other metrics like performance, customizability and readability is done to validate the metric.

Lakshmi Narasimhan and B.Hendradjaya[6],suggested some theoretical considerations for a suite of metrics for the integration of software components. In this research static and dynamic aspects of component assembly are addressed. Static metrics measure complexity and criticality of components assembly. Complexity is measured using component packing density and component interaction density metrics. Dynamic metrics are calculated during runtime of the complete application. It helps in evaluating the degree of utilization of various components.Dependability certification of software components was analysed by Jeffrey Voas and Jeffery Payne in order to attain a uniform approach in rating the quality of software components. The paper proposed a **Test quality rating (TQR)** to obtain components dependability score[9]. They suggest that it can be done by an individual organization and the score can be displayed on any marketing materials or contracts which license that component.

III. PROPOSED MODEL

Risk maturity model developed will surely help to evaluate various risks in COTS and their quality. The model was designed based on industrial perspective which includes interaction risk, criticality and functionality updates. Model consists of 5 stages of analysis which includes:

- Requirement risk analysis
- Integration risks
- Criticality
- User friendly factors
- Security

The purchased COTS product has to undergo all the 5 stages of the model. Only if the COTS product satisfies the requirements of all the stages in the model the component can be integrated into the system. Following steps are followed:

- Requirement risks is focused, here it is checked whether **the requirements**of the end users are matched with purchased COTS and best among the COTS component vendors are chosen from the repository based on the requirements and integrated into the system. If this is fulfilled the component attains one star maturity level.
- Next level of the risk model is risks involved during **integration,** here incoming and outgoing interaction complexity is measured using a metrics and risk percentage is obtained. If risk percentage is less next level can be evaluated else mitigation strategies are applied to reduce the percentage of risks considerably risk percentage is reduced and then next level of the

model can be applied to the COTS product. Here second level risk maturity is obtained.

- Third level of the model is **analysis of criticality**; here number of links, bridges, line of code and inheritance values is calculated. If all the values mentioned above do not exceed the threshold value third level of maturity is obtained.
- Fourth Level is risk factors affecting the **user friendly nature of COTS.** Here attractiveness features are analyzed. This is done using component manuals, demos, help system and marketing information. Thereby COTS software attains fourth level of maturity.
- Fifth factor is risks involved in **security,** this differs based on various application.
  Here the COTS vendors have to answer certain questions and risk factors are analyzed. Thereby all five stages of risk maturity model will be achieved.

After undergoing all the above steps the vendors have to answer nearly 40 questions in the checklist and here risks are classified on the basis of high, medium and low. Therefore many risks are eliminated if the above steps are followed by COTS.

IV. FIRST PHASE: REQUIREMENT RISK

Risks involved in requirement phase are:

- Matching with user needs
- Best COTS selection among various vendors

A. Matching with User Needs:

The developed COTS should map with end user requirements therefore the COTS developers should clearly obtain the requirements before developing the software. The developed software should not be outdated by the time of integration which means it has to be adaptive.

MITIGATION STEPS:

- Analyze the environment where it is integrated.
- Obtain the requirement factors
- Convert the requirement into actors
- Analyze the dependability of actors
- Map it with the existing quality models

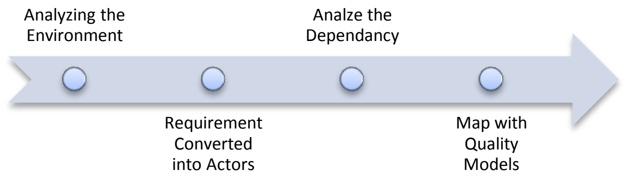


Figure2: steps involved in mapping with quality models

Example:

Let us consider a mail server system as a COTS component here. After it is purchased by the third party vendors requirements are checked. By following the above steps, first environment are analyzed here it is found that it is a web environment and there are infinite numbers of users.Secondly, the requirements for mail server system are mail server administrator, mail server user, firewall and mail client. They are converted into actors.Third stage is to analyze the dependency of the actors. In order to attain the dependency we are in need to find the sub-factors of the actors. The subfactors of mail server system are:

- Mail server administrator(MSA)
- Mail client(MC)
- Mail server user(MSU)
- Firewall
- Mail client(MC)

The sub-factors of the above actors are represented below to find out the dependency of each.

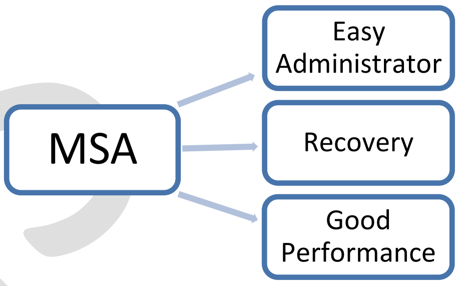


Figure3: Mail server system and their sub factors

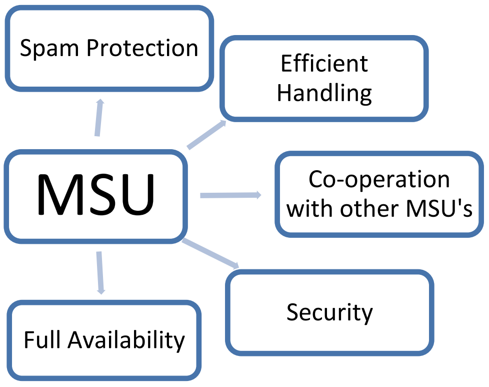


Figure4: Sub factors of Mail server User

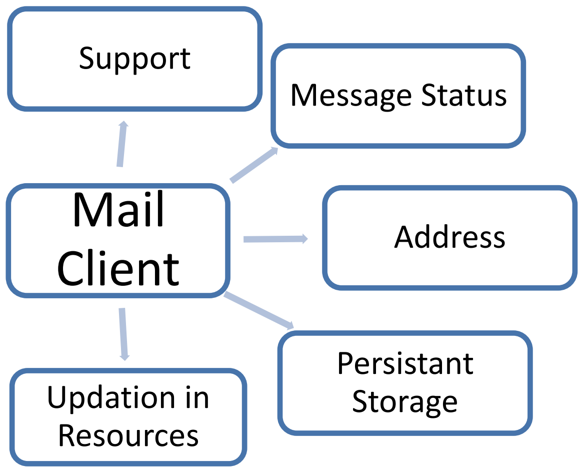


Figure5: Sub factors of mail Client

Therefore actors and their dependencies are obtained with these factors we have to map them with a quality model available. The suitable quality model satisfying all the factors of mail server system is ISO/IEC MODEL.

PARAMETERS	***ISO/IEC model***
No. of required users	Resource utilization
Simultaneous connections allowed	Fault tolerance
Percentage of connection by mail connect type	Security and interoperability
Average amount of information managed	Resource utilization

Table2:Mapping of parameters to quality model.

The mapping involves can be classified as two goals market driven and goal driven. Market driven is to update knowledge of types of tools currently available in the market and the services they provide. Goal driven approach is the ability to identify the actors in the system and their goals.

Cots Selection:

There are lot of vendors providing the COTS components all are not of good quality therefore best among the vendors has to be selected. The industry willing to purchase COTS have to maintain various COTS vendors and the features they provide in a repository. Simulator is available to obtain it. This method overcomes cost and time overrun while integrating and deploying the COTS components. The users can map their needs based on their functionalities.

V. SECOND PHASE: INTEGRATION RISK

While integrating the COTS software into the system there is a chance of system performance to be affected. This is due to increase in incoming and outgoing interactions, changes in the system before integration and the problems involved in the glue code. Many COTS are shipped with application programming interface for smooth integration of components. Prototypes must be created to identify the performance impact before accepting and supporting the suitability of the technologies. Connection of COTS for remote collaboration consumes lots of resources due to allocation of networks and memory resources for them.

Mitigation:

The changes in the system should be updated when the components are integrated this helps in analyzing the adjustability of the COTS software with the changing environment. It has to be adaptable to all kind of situations.The risk percentage can be calculated based on incoming and outgoing interactions. LatikaKharb and Rajendra Singh proposed a complexity metrics for component oriented software systems[4].

$$AI = \frac{I_i + I_o}{I_{max}}$$

Where,

- **AI -** Actual number of interactions
- **$I_{(i)}$**– Incoming interaction
- **$I_{(o)}$** - Outgoing interaction

TIP = AI / C

- **TIP-** Total number of interactions
- **C** - Total number of components

Change Notification:

The change notification plays a major role is assessing the interaction risks of COTS components. The model is proposed by means of a flow diagram, the model identifies the changes and notifies the components based software.

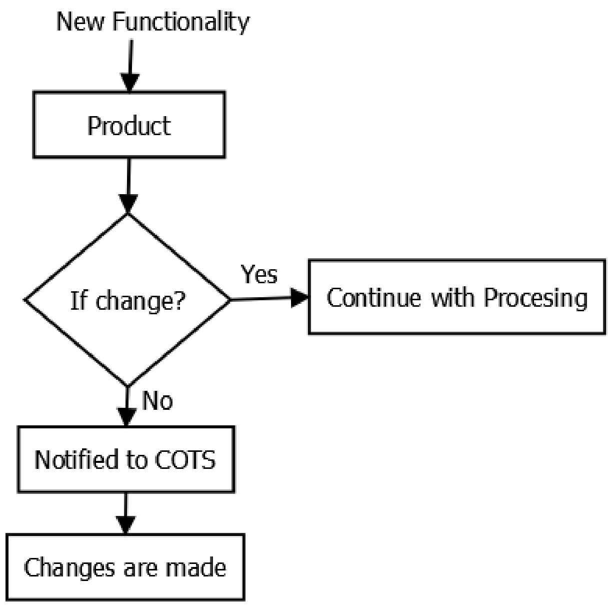


Figure6: Change Notification

VI. THIRD PHASE: FUNCTIONALITY METRICS

The functionality comes under the suitability factor. Functionality has both extra and required functionality. Required functionality is nothing but the requirements suggested by the end users to the vendors. Extra functionality is additional functions provided by the vendors. If extra functionalities are increased interactions of COTS also increases therefore it has to be moderate in number. Unwanted functionalities increase the complexity.

$$Risk\ \%\ of\ Functionality = \frac{EFC - RFS}{TFES} * 100$$

Where,

- EFC – Extra functionality provided by the component
- RFC- Required functionality provided by the component
- TFES- Total number of functionalities required by component based system.

Criticality is extreme risk factor which will lead to complete damage to the system which is using the COTS component. It arises when interactions, links, bridges and component sharing the attributes exceeds the threshold value.

Risk percentage metrics for criticality is the ratio of sum of links, bridges and sharing attributes exceeding the threshold value to the total number of pairs involved in the component.

$$Critic\ \% = \frac{ETV_b + ETV_{sa} + ETV_i}{N} * 100$$

Where,

Critic - Criticality

ETV_b – Number of bridges exceeding the threshold value

ETV_{sa} – Number of sharing attributes exceeding the threshold

ETV_i– Number of links exceeding the threshold value

N – Total number of pairs in the component

The COTS must be ready to handle both planned and unplanned outages. The procedures to diagnose failures and attempt to adapt to conditions must be reviewed properly. Methods to reduce criticality are:

1. Moduleshas to be reconnected with failed COTS components.
2. Router Algorithms has to be established to identify and notify the failures to other peer components.
3. Alerting mechanisms should be introduced to inform the entities.

VII. FOURTH PHASE: USABILITY

The component developed must be user friendly its one among the required factor to evaluate the adaptability and flexibility of COTS components. In COTS the usability is based on learnability and attractiveness to the user on various domains. The attractiveness is based on manuals; demos help desk and marketing information. Usability goal intends to follow human factors approach.

The component provider must demonstrate efficiently and manuals must provide easy understanding to the component based software for the end users. The maintenance is also important; after the component is sold out if any problem arises the help desk must support the end users. Help desk is also one among the usability factors. Since usability plays much importance in Component based software risks involved in it must also be evaluated.

Vendors providefigures, tables and design diagrams to increase the learnability of COTS components but while using the component based software end users find it difficult to handle the component, therefore the user must check whether the developers provide all the features mentioned in demos and manuals. The metrics for usability includes demo coverage value and manual size.

VIII. FIFTH PHASE: SECURITY RISKS

The final and most important phase of risk maturity model is security. Without achieving the security needs the COTS component cannot be matured. Data communication has to be secure due to lot of interactions involved in COTS. In this phase resources is intense due to encryption and decryption algorithms used.

Performance cost of handshake is more than the cost due to encryption and decryption of data transferred.The following questions has to be answered by the vendors to evaluate the risk factors:

1. Does the system switch to non-secure communication when data security over the network is not necessary?
2. Are connections kept alive always during communication between components is going on?
3. How many request can be passed during live connection before the connections is dropped?
4. Will performance of encryption and decryption be affected during the communication process?
5. Does a hardware accelerator support the environment of COTS?

The questions may vary based on various domains; the above questions are common to all applications.

Checklist:

COTS components usually require interface to an existing system. Interface may be simple or difficult to implement so resources are needed to resolve those problems. Before analyzing the risks we have to understand the COTS product functionality. Checklist with three options each are listed out to obtain a risk framework. We have to find out whether the organization can accept the gap in requirements without degrading the performance.

The industry should have the time, financial and personnel resources necessary to support the component based software product. Performance and scalability has to be validated and the users should select the matured product. The diagram below will surely provide a risk maturity framework which helps in validating a COTS component. Followed by it checklist will be listed out.

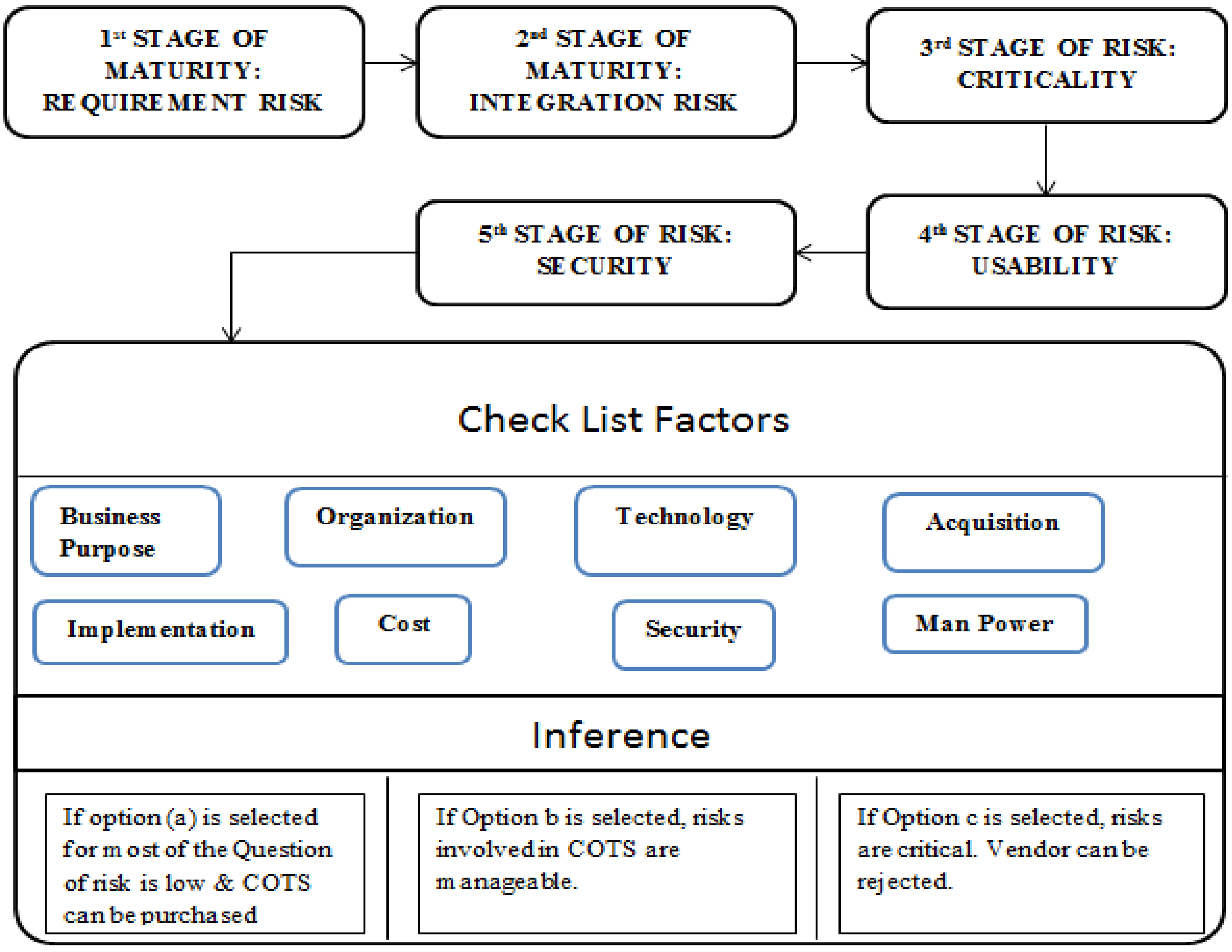

1st STAGE OF MATURITY: REQUIREMENT RISK
2nd STAGE OF MATURITY: INTEGRATION RISK
3rd STAGE OF RISK: CRITICALITY
4th STAGE OF RISK: USABILITY
5th STAGE OF RISK: SECURITY
Check List Factors
Business Purpose
Organization
Technology
Acquisition
Implementation
Cost
Security
Man Power
Inference
If option (a) is selected for most of the Question of risk is low & COTS can be purchased
If Option b is selected, risks involved in COTS are manageable.
If Option c is selected, risks are critical. Vendor can be rejected.

Figure7.Risk Maturity Model

IX. RISK MATURITY MODEL

Fig7 depicts Risk Maturity model to evaluate the risks in COTS components. The model is applicable to various domain of Component based systems. The model eliminates all sorts of critical risk and selects the best among COTS vendors. Followed by it the organization has to check the features with the checklist, if most of the questions are answered as (a) the risks involved is nil and the component can be purchased and installed, if (b) is opted risks involved is manageable, if (c) is selected risks involved is too critical. The COTS need not be purchased. The questions are based on performance, cost, support, schedule, functionality, requirement factors, criticality, security and usability. Organizations environment can also be predicted at this stage.

1. How is organizations business requirement documented?
   a. Current status updated
   b. Updation not done recently
   c. Minimal documentation
2. How organization ability can be described to adapt to COTS components?
   a. Well adaptable
   b. Somewhat able
   c. Not possible
3. Where requirements analyzed to determine the fit of the identified requirements with the COTS products?
   a. Yes
   b. Not sure
   c. No
4. How many COTS product can be accommodated by the organization?
   a. Many
   b. Some
   c. Few
5. How need for the organization is characterized to respond to mandatory and quick changes?
   a. Demands for changes are limited
   b. Demands are moderate
   c. Demands are frequent
6. Who is responsible if organization gets affected by COTS?
   a. Customers purchasing COTS
   b. Middle management
   c. Executive management
7. How many sites of the single organization will be affected if changes are made due to COTS?
   a. One
   b. Several
   c. Many
8. What is the geographic dispersion of the organization where implementation of COTS is done?
   a. Local office
   b. Regional location
   c. National dispersion
9. What is the nature of operational control affected by COTS products?
   a. Centralized
   b. Centralized as well as decentralized
   c. Decentralized
10. Does the organization has the ability to support new configurations and processes?
    a. Can support
    b. Needs improvement
    c. Cannot support
11. Are the staffs skilled who are involved in COTS application projects?
    a. Efficient and skilled at all locations
    b. Minimal staffs but skilled
    c. Insufficient staffs and less skilled
12. Is the project team experienced in implementing the COTS products?
    a. Well experienced
    b. Moderate experience
    c. No experience
13. If DBMS is included with COTS packages what is the experience of the team members?
    a. Extensive experience
    b. Moderate experience
    c. Experience is nil
14. What is the level of customization needed for COTS product purchased?
    a. No customization
    b. Less number of customization
    c. More customization is necessary
15. How does COTS fit with organizations existing and planned architecture?
    a. Best fit
    b. May fit
    c. Not fit
16. How many interfaces remain without changing after COTS is implemented?
    a. Few
    b. Some
    c. Many
17. What is the complexity of interfaces between COTS and the existing system?
    a. Not complex
    b. Less complex
    c. Very complex
18. What kind of testing is conducted to COTS product in the organization?
    a. Extensive testing
    b. Less testing
    c. Testing not done
19. Do the security features have to be modified to meet the needs of COTS implementation environment?
    a. No modification
    b. Some modification is needed
    c. More modification has to be done
20. Does the database design and structure of COTS support the plan of the organization?
    a. Supports all requirements
    b. Only some requirements can be supported
    c. Requirements are not supported
21. What is the run time performance of COTS in organization environment?
    a. Efficient
    b. Moderately efficient
    c. Not efficient

22. How flexible is the COTS to accept changes in the functionality?
    a. Very flexible
    b. Moderately flexible
    c. Not flexible
23. On which basis COTS application packages can be purchased?
    a. Performance basis
    b. Fixed price
    c. Cost reimbursable
24. Does COTS product purchased mature enough to handle the disruption caused?
    a. Matured to handle all situations
    b. Somewhat mature
    c. Immature
25. What is user's satisfaction level in using COTS?
    a. Very well satisfied
    b. Limited satisfaction
    c. Not satisfied
26. What is the vendor experience in handling COTS product similar to your organization?
    a. Extensive experience
    b. Some experience
    c. No experience
27. Has vendor experienced in performing integration of COTS to some other organization?
    a. Excellent past experience
    b. Good experience
    c. Poor experience
28. Are the COTS product user satisfied with the experience of vendor staff?
    a. Very satisfied
    b. Somewhat satisfied
    c. Not satisfied
29. How far the organization involves in future plans of vendors providing COTS product?
    a. Planned enhancements and date of release has been updated
    b. Discussions are conducted
    c. No updates regarding the future enhancements
30. Are maintenance and customization included in the cost proposal based on the changes?
    a. Every changes are negotiated in the cost
    b. Only few changes are negotiated
    c. Uncertain about the changes
31. Has the organization learnt lessons from other organization who has purchased the same COTS of yours?
    a. Lessons learnt are incorporated into the implementation plan
    b. Past projects has been discussed but not implemented
    c. No information has been gathered regarding the past projects
32. What are the performance measures to analyze the effectiveness of COTS product?
    a. Cost and time spent on every activity
    b. Performance measure are discussed but not finalized
    c. No discussion about performance measures
33. What are the testing approaches planned for COTS application products?
    a. Designed in specific for COTS
    b. Traditional testing approaches are combined with COTS specific testing
    c. Only traditional testing methods are followed
34. Who are responsible for implementation schedule?
    a. Developed by implementation team
    b. Developed by individuals not responsible for implementation
    c. No implementation was developed
35. What kinds of process will the organization implement new requirements after initial implementation of COTS products?
    a. Well defined process to evaluate and implement new requirements
    b. Discussions are made but not yet ready to adapt to new requirements
    c. No process exists for evaluating new requirements
36. What is the organization ability to support new releases of COTS?
    a. Sufficient
    b. Moderate
    c. No resources
37. How far organization is ready to handle the situation if vendor goes out of business?
    a. Ready to handle the worst situation
    b. Possibility discussed not finalized
    c. No contingency plan developed.

Therefore if the model and the metrics proposed here is applied to all COTS application packages risks can be eliminated.

CONCLUSION AND FUTURE WORK

Thus risk profile is created with the help of above model. It's a step by step procedure to verify various risks which may affect the entire system using COTS. Past research helps only in providing mitigation steps but they cannot be applicable to all domains. The risk maturity model proposed here can be applicable to almost all domain using COTS application packages. The future work is to enhance the present work by applying the metrics and methods presented here to applications like airline reservation system and mail server system. Both system are purchased as COTS from third party vendors. The above mentioned packages have lot of interactions and due to which complexity may arise. The model will surely focus on the risk involved due to interactions and additional functionalities.

REFERENCES:

[1] Arun Sharma, Rajesh Kumar and P.S.Grover, "Empirical Validation of interface complexity metrics for software components",International Journal of Software Engineering Vol18,No.7(2008) 919-931.

[2] Chongwon Lee, Byungjeong Lee and Chisu Wu, " Providing guidelines of determining Quality Checklists Priorities Based on

Evaluation Records of Software products", 15th Asia Pacific Software Engineering Conference(2008).

[3] Barry Boehm and JesalBhuta, "Balancing Opportunities and risks in Componenst based Software Development" by IEEE computer Society 2008.

[4] LatikaKharb and RajenderSingh,"Complexity "Metris for Component Oriented Software Systems", ACM SIGSOFT Software Engineering Notes Volume 33 Number 2, March 2008.

[5] R. Moraes, J.Duraes, R.Barbosa, E.Martins, H.Madeira , "Experimental Risk Assessment and Comparison Using Software Fault Injection", 37th Annual IEEE/IFIP International Conference on Dependable Systems and Networks(DSN'07)

[6] V.LakshmiNarasimhan and B.Hendradjaya , "Some theoretical Considerations for a suite of metrics for the integration of software components", Elseiver Science Direct(2007).

[7] SahraSedigh-Ali and ArifGahoor& Raymond A.Paul, " Metrics and models for cost and quality of Component based software", IEEE Proceedings of the Sixth IEEE International Symposium on Object Oriented Real Time Distributed Computing(ISORC'03).

[8] Juan P.Carvallo, Xavier Franch and CArmeQuer, "Requirements Engineering for COTS based Software Systems", SAC'08 ACM Publications.

[9] Ned Chapin, "Entropy-Metric for syatems with COTS Software", Proceedings of the 8th IEEE symposium on Software Metrics(METRICS'02).

[10] V.PrasennaVenkatesan and M.Krishnamoorthy," A Metrics Suite for Measuring Software Components"(2006).

[11] Jeffery Voas and Jeffery Payne, "Dependability Certification of Software components ", The Journal of Systems and Software Elseiver 2000.

[12] Jingyue Li, ReiderConradi, Odd PetterN.Slyngstad, Christian Bunse, Umair Khan, "Validation of New Theses on Off-the shelf Component Based Development"(2010).