

Additional Costs And Risks In Software Acquisition Projects

Maliha Haddad (Email: mhaddad@gwu.edu), George Washington University
Anita La Salle (Email: lasalle@american.edu), American University

Abstract

Organizations usually contract their software projects to avoid the risks associated with developing the software internally and to control their costs. However, a study of two-dozen contracted project indicates that such organizations face unique risks and hidden costs that are particular to software acquisitions. This paper describes research done to estimate the effort expended by organizations in overseeing and participating in contracted software projects and the implications for predicting costs and identifying risks of such projects. It presents a framework for collecting and measuring costs incurred before, during and after the contract award. For many of the organizations that participated in the survey, realizing the actual costs and risks of a project was an eye opener – hidden costs and risks are significant and they are typically not managed. The research results emphasize the need for institutionalizing processes for the collection of data about contracting costs within an organization so that databases of metrics about completed projects can be built and later used to forecast costs for future projects to improve decision-making processes. The research results also emphasize the need to acknowledge the risks involved so they can be mitigated to improve the relationship with the contractor and the chances for project success. The authors are engaged in research directed toward assisting such organizations in identifying the risks and costs and improving the acquisition process.

1.0 Introduction

An increasing number of organizations are contracting their software development projects to outside contractors predominantly to avoid the risk of failure due to continuously evolving technologies and lack of expertise in-house. This is particularly true for government agencies where technical salaries have not kept pace with salaries offered by business resulting in an inability to develop internal expertise. Studies show that between one fourth and one third of U.S. software projects involve outside contractors. However, such an environment introduces additional cost factors and risks attributed to the interaction between the contractor and the customer. The primary additional costs include the contracting organization personnel involved in planning, acquiring, and managing the project, and user involvement and participation.

If we look at software cost estimation processes, methods and tools in use, most estimate the cost of the technical and some management and support resources needed for a software project. What is missing in these tools and techniques are estimates of the contracting organization resources that are incurred before, during and after the contract award, in particular the user involvement dealing with the contractor during development processes. Such costs are considered hidden because they are incurred but not accounted for. This paper addresses the need for improvements in existing software cost estimation processes by making the hidden costs of a contracting organization visible. The results of this research are significant to contracting organizations as they show that the hidden costs are incurred, are significant, but not managed. Failure to plan and schedule critical resources, such as user involvement, result in introducing risks to the project as these resources might not be available when needed. Incorporating the hidden costs into software economic analyses could improve the decisions making process.

Readers with comments or questions are encouraged to contact the authors via email

The paper also describes some of the sources of risk to the customer and introduces three primary areas of risks: internally within the customer's organization, externally within the contractor's organization, and in the interfaces between the customer and the contractor.

2.0 Background: Software Acquisition Process Improvement Efforts

Particularly significant to the study described in this paper is the set of CMM recommendations, developed by the Software Engineering Institute (SEI) at Carnegie Mellon University, and widely used in the USA. The CMM was developed in the late 80s as a result of the SEI being tasked by the Department of Defense (DoD) to develop a means to evaluate the software development capability of its bidding contractors. The processes recommended by the SEI in their Capability Maturity Model for Software development (CMM, CMM-I) [21,25] are now widely accepted as productive ways for organizations to achieve higher quality software than in the past. A more recent focus of the SEI is on the processes involved in software acquisition. The experience of the Software Engineering Institute in developing the Capability Maturity Model for Software (SW-CMM) was directly applicable to developing the Software Acquisition Capability Maturity Model (SA-CMM). The SW-CMM describes the developer's (contractor's) role while the SA-CMM describes the buyer's (acquirer's) role in the software acquisition process. The same maturity paradigm used in the SW-CMM was used in building the SA-CMM. The SA-CMM addresses the functions that support the acquisition of software. The SA-CMM is designed to be generic enough for use by any government or industry organization, regardless of size, acquiring software. Organizations and individuals involved with outsourcing soon realized that customer involvement meant involvement in all stages of the SDLC if a reasonable product was the expectation. And, that involvement requires not only contract oversight, but also user, tester, inspector, and other kinds of participation. Few estimation methodologies take into account the cost to the contracting organization of this involvement or what form the involvement or cost estimation should take.

As with the CMM, the SEI sought to help contracting organizations by defining processes that could improve software acquisition by specifying the following maturity levels for the SA-CMM that characterize acquisition management:

- Initial – The software acquisition process is characterized as ad hoc, and occasionally even chaotic. Few, if any, management controls are put in place.
- Repeatable – Basic software acquisition project management processes are established to plan all aspects of the acquisition, manage software requirements, track project team and contractor team performance, manage the project's cost and schedule baselines, evaluate the products and services, and successfully transition the software to the support organization. The project team is basically reacting to circumstances as they arise.
- Defined – The acquisition organization's software acquisition process is documented and standardized. All projects use an approved, tailored version of the organization's standard software acquisition process for acquiring their software products and services. Project and contract management activities are proactive, attempting to anticipate and deal with acquisition circumstances before they arise. *Risk management is integrated into all aspects of the project*, and the organization provides training required by personnel involved in the acquisition.
- Quantitative – Detailed measures of the software acquisition processes, products, and services are collected. The software processes, products, and services are quantitatively and qualitatively understood and controlled.
- Optimizing – Continuous process improvement is empowered by quantitative feedback from the process and from piloting innovative ideas and technologies. Ultimately, an organization recognizes that continual improvement (and continual change) is necessary to survive.

In the survey that supports the authors' research, it was evident that most organizations were not even aware of the SEI's SA-CMM or SW-CMM. Few had formal, institutionalized acquisition planning and tracking processes for software; contractor selection processes (if formalized) tended to follow traditional selection processes for products and services; software requirements documents lacked precision and specificity; few projects had formalized acquisition project management, contract management, or configuration management plans for tracking

project activities and artifacts, overseeing internal and contractor personnel, evaluating progress and performance against requirements, or tracking costs; customer inspections of defined milestones were not part of the acquisition processes; risk identification and management processes were missing; and, data collection and archiving, if done at all, was the consequence of individual efforts rather than institutionalized requirements for software acquisition. Few projects involved post-mortems of the project's lifecycle to improve processes in the future.

3.0 Scope of Survey

In surveying companies about their contracted software project costs, a questionnaire was prepared and distributed. In some cases, the target-organization returned the completed survey. In others, face-to-face interviews were carried out using the questionnaire as an interview instrument. User's cost estimation processes were identified and examined, and later the users and management activities costs were measured for the surveyed projects. The respondents' organizations represented a broad range of companies and government agencies. The contracted projects costs studied ranged from \$30K to \$50M. Cost estimation methods varied widely ranging from "No estimation method" to the use of "Lines of Code." The most common project estimation technique was "Experience." Regardless of the estimating technique used, 61 % of the projects had cost over-runs and 50% had schedule over-runs. The survey was designed to examine the contracting organizations cost estimation processes, to determine the magnitude of the hidden cost and the impact of not managing the hidden costs including the risks to the organizations. The focus of the study was to determine:

- The cost incurred by the procuring organization for resources needed before and during the system development lifecycle to acquire, manage, coordinate, control and support the software project.
- The effort expended by management, users, support and other personnel with the required skills and expertise, and hardware and software tools required to support such efforts.
- The cost of management of the contract, the contractor, the users and the quality of the product for the duration of the project.
- The involvement of the users, user management and the user functional experts who have to participate in most activities in the development life cycle including requirement definition, product reviews, document reviews, and testing activities throughout the lifecycle.
- The cost of quality assurance activities by quality assurance personnel.
- The cost of software tools, hardware tools, travel expenses, user training, acceptance testing, management of deliverables, and management of the user and other support and miscellaneous expense items in support of the various activities.

Results related to the Contracting Organization Cost Estimation Processes and Practices

The results of the survey showed that:

- The majority (88%) of procuring organizations in the sample did not estimate their resources on completed projects.
- The majority (65%) did not have formal processes to estimate, plan and schedule such resources.
- The management resources, which are critical for project oversight, are more likely than other resources to be planned by organizations, but still are not planned by 69% of organizations.
- The user resources, whose involvement in projects is critical, are not planned by 92% of organizations.
- The cost of such resources is not included in any economic analysis or feasibility study of the project by 57% of the organizations surveyed.
- The collection of historical data on resources involved in completed projects is not done by 62% of organizations.
-

The lack of formal processes for planning resources introduces risks to the project as some of the critical resources are not committed formally and might not be available when needed, in particular those of management and the users. Jones [17].

The conclusion is that estimating the hidden cost of a contracting for over fifty percent of the organizations is an ad-hoc process, usually an indication of a low maturity level.

The survey tried to identify the source of funding of resources such as management, users, and others (other than contractors) working on software development projects and whether the cost of such resources are included in the final project cost.

The results indicated that the funding of resources, such as a project manager or a user, does not come from a project’s budget. Since such costs are incurred but not tracked or included in the final project cost, they are considered hidden. This also means that the contracted price of a software development project does not reflect the actual cost. In addition there is likely to be a lack of commitment of resources whose functions may be critical for project success, Jones [18], and therefore risks are introduced.

5.0 Results of the Measurement of the Additional /Hidden Cost

The findings from this part of the study exceeded the expectations of what the value of the hidden costs might be. Initially, it was estimated to be about fifteen percent of the project development cost. The study showed that the hidden costs are quite substantial, the mean value being 190 % of the total development cost of the system, almost twice as much as the contract cost. The value of the hidden cost is significant enough to motivate organizations to have formal procedures to account for it and include it in project plans. The value of the hidden cost must also be included in estimation models used for economic analysis of software projects and must be taken into account in the decision-making processes when assessing feasibility of a project.

The attribution of percentages of hidden costs to phase distribution in Table1 shows that analysis and implementation constitute the largest percentages followed by the design and testing phases.

TABLE 1: Phase Distribution of the Hidden Cost

Phase	Percent of Hidden Cost
Acquisition	7.1
Analysis	18.9
Design	14.2
Programming	5.6
Testing	18.4
Implementation	18.9
Training	8.5
Other	8.4

Studies show that analysis is considered one of the most risk prone activities of a system’s SDLC, an activity in which the user is heavily involved. The role of the user is also important during the testing of the software product for acceptance and when the product is deployed.

The results of the distribution of the hidden cost by labor categories in Table 2 show that the users and project management resources are the most significant.

TABLE 2: Distribution of the Hidden Cost by Labor Category

Labor Category	Percent of the Hidden Cost for the Labor Category
Project Management	34
User	48
Quality Assurance	6
Consultants	6
Others	6

This result is important in the sense that it draws attention to the necessity to plan and schedule the procuring organization’s resources. The distribution of the hidden costs by labor category and by phase draws attention to the various needed resources and when they should be planned and scheduled during the various phases of the software project along with their proportions. The results of the study concerning the user and management resources needed on a

software project complements and supports results of other studies. For example, a recurring theme in risk studies, such as the one by Statz and Tennisson [26], is that user participation, customer and user interaction, and user resource allocation, are sources of risks that need to be assessed and managed. Also, risk studies by Jones [18] point to the importance of project management activities. Project management is considered by Jones [17] to be one of the key factors for the success of systems. The general conclusions that we can make based on these results of the study

is that the hidden costs are substantial, and the needed resources must be planned, scheduled and better managed during the points in the lifecycle when their participation is critical.

In addition, the study showed a very strong relationship between the project size and the hidden cost expressed in man-months. For the 26 projects studied, the influence of project size (removing outliers) was found to be linear and can be expressed as:

$$M = 2.2 * KLOC + 52$$

Where M is hidden cost in person-months and KLOC are thousands of lines of code. It should be noted that different projects recorded data about project size primarily as function points or lines of code. The relationship was made uniform for KLOC using widely accepted ratios that are dependent upon the development language/system used to produce the project.

For many of the organizations that participated in the survey, realizing the actual cost of a project was an eye opener -- hidden costs are incurred, the costs are significant, and are typically not managed. Organizations that understand inherent costs of contracting software are better positioned to estimate costs of future projects and also improve decision-making processes associated with software contract oversight

5.0 Risk Identification and Management in Software Acquisition

In addition to identifying the hidden costs, the customer's assessment and management of risks associated with projects is equally important. At a minimum, failure to plan and schedule critical resources such as users, project managers, domain experts, management software, and others may pose a risk to a contracting organization and to the project itself. Some risks have significant side effects that impact on other projects within the customer's organization. Some risks are magnified by sub-contracting relationships. This phase of the research explores the risks associated with sub-contracting with the expectation that recommendations can be devised that will help minimize those risks or at least help organizations design mitigation plans.

What are some of the sources of risk to the customer? Risks are introduced in three primary areas: internally within the customer's organization, externally within the contractor's organization, and in the interfaces between the customer and the contractor. Some conditions that might be defined as potential risks from any of the three sources are not really risks in that there is a certainty of their occurrence but they are typically not identified by the customer or the contractor and they translate directly into cost or schedule overruns, in degradation of the customer-contractor relationship, or reduction in software quality.

The following is a list of risks in the three primary areas. Mitigation plans for these risks are left unspecified since they also vary between projects and organizations:

5.1 Customer Internal Risk Sources

- Inaccurate Estimations: Customers, even those who use estimation tools, can be characterized as "gutless" estimators of time, cost, and effort needed to produce software. Published software metrics uniformly support that real effort normally exceeds
- Personnel: Unique knowledge, training and skills are needed to successfully launch, monitor, and bring to successful completion a software project that is contracted. Few organizations have enough personnel who are trained and capable in the processes involved in software acquisition.
- Users: The greatest risk in this category is the availability and knowledge of future system users at the customer's site. Frequently overlooked is the consumption of user time on a project that interferes with the user's normal work.
- User Requirements: Customers frequently fail to understand, precisely articulate, or fully specify their software requirements.

- **Contract Specificity:** Related to specification of user requirements, but broader in scope, is the specification of not only software requirements but also processes to be used between the customer and the contractor and the details of the interfaces between the two.
- **Creeping Requirements:** Also related to specification of user requirements and contract specificity is the problem of requirements that change or expand. This characteristic of software projects has been exacerbated by web applications where changes to interfaces usually imply underlying changes in functionality.
- **Unanticipated Coordination Efforts:** Management of a contracted software project, if it is to be successful, requires a great deal of interfacing between customer and contractor during the entire life-cycle of the project. Few contracting organizations anticipate the rigor demanded in coordination and oversight.

5.2 Contractor Internal Risk Sources

Like the availability to the customer of the SEI's SA-CMM, software contractors have the SEI's CMM/CMMI [6, 20, 24] recommendations to adhere to and enable their organization to position itself to successfully complete high quality software projects. We include the contractor subsection here for completeness but intentionally ignore it, concentrating instead on the interfaces between the contractor and the customer that pose risks.

5.3 Risk Sources in the Customer/Contractor Interfaces

The interfaces between the customer and the software contractor are the source of many risks that are typically not acknowledged and therefore not managed or mitigated. In the ideal world, customer and contractor would share processes and tools, their activities would be synchronized, and oversight would be welcomed. However, the number of failed contracted projects is testimony to the problems that arise in the relationship between the customer and contractor. Like the list in the previous section, the following list is intentionally not prioritized since, depending on the project and the organizations involved, the severity varies among projects and organizations. In addition, contracts where software components are sub-contracted and must ultimately be integrated pose additional risks and risk complications. The following is an abbreviated list of potential risks between customer and contractor.

- **Mutually Accepted Ambiguous Contract:** In this case, both the customer and the contractor are willing to sign a contract that is not precise in the specification of the work to be done.
- **Ill-defined Interfaces:** The parties do not specify the personnel interfaces for work coordination, nor do they specify the formal processes for customer-contractor interaction.
- **Multiple Contacts:** This is related to the previous item and involves interaction of multiple personnel on both sides. Audit trails are lost, multiple conflicting approvals may be given, and procedures, if any, are side stepped.
- **Antagonistic Interfaces:** Over time and during stress, customer-contractor interfaces can suffer because of failed performance (perceived or real) on either side.
- **Deficient Inspections:** Work inspections are central to the success of a contracted project. When inspections (assuming they are scheduled) are ignored, postponed, poorly structured, or ineffectual, trust suffers.
- **Loosely Defined Checkpoints:** When a mutually agreed upon project management plan is absent or deficient, agreement on acceptable checkpoints may be a source of contention.
- **Testing Criteria, Processes and Data:** Frequently overlooked by the customer is the necessity to provide application data in a format, and in a timely manner, so that can be used in the contractor's testing processes.
- **Shared Repositories:** The existence of shared repositories of life-cycle artifacts bodes well for a project. However, ownership and controls can be a source of problems between the customer and contractor.
- **Configuration Management:** Related to the previous item, configuration management controls may be the source of problems about shared documents and even deployed systems.
- **Risk Management Program:** Negotiating agreement on a risk management plan is difficult. The contractor may be resentful of intrusive processes and the customer may be suspicious of opaque contractor processes.


- Quality Assurance Program: When the customer has more stringent QA processes than the contractor (or vice versa), quality inspections are likely to be antagonistic.
- Potential for Re-use and Maintainability: Customers have fewer opportunities for controlling re-use potential unless processes are in place to insure re-use. Control over design approaches for ease of maintenance also presents a risk.
- Missed Schedules: With a well-developed project management plan with high task granularity, the contractor's schedules may be fully defined and oversight straightforward. However, frequently overlooked is the responsibility of the customer to provide contract backup in any number of ways. The customer's tasks, too, need to be supported by a thorough (synchronized) project management plan.
- Cost of Tools: System development, project planning, project management, oversight, documentation, and all of the other phases of the systems development life cycle require automated tools for building repositories of project artifacts.
- Incompatible Deployment Infrastructures: Insuring that the development architecture is compatible with the deployment architecture is an often overlooked aspect of a contracting arrangement, a source of added costs, and a source of contention.
- Security: Breach potential within the system architecture poses short and long term risks and data used for testing or benchmarking presents potential security vulnerability.
- Transient Contract and Customer Personnel: Employees change jobs or are moved within organizations.
- Unanticipated Direct Costs: Travel, on-site/off-site space requirements, audit requirements, user testing costs, meetings, use of external "overseers", materials and equipment.

6.0 Benefits In Managing The Hidden Cost And Identifying Risks

During the data collection phase of the authors' research, many top managers of organizations who were participating in the survey demonstrated and aroused interest in the study and the anticipated findings. As they completed the survey, they became conscious of the magnitude of the hidden cost that they typically overlooked. Their interest can also be attributed to the following benefits that they can achieve in estimating hidden costs:

- Including the hidden cost in strategic decisions made by executive management can improve the accuracy of estimates and the decision making process before undertaking a software project. A decision is usually based on an economic analysis, which requires an estimate of the money, resources, and time required to complete the project. By including the hidden cost in cost benefit analysis, breakeven analysis or make-buy decisions, managers can improve the accuracy of their estimates and confidence in their subsequent decisions.
- Using the estimates of the hidden cost in project management to plan, monitor and control the development of a project can improve the management process. Good planning and effective control require an estimate of the activities required to complete a project, and the resources required for each activity for monitoring progress.
- The costs estimates can also be used for the allocations of adequate funds for projects over time. The relationship between cost estimation and controlling projects are detailed further in DeMarco [8].
- Using the estimates of the hidden cost in work breakdown structure to allocate resources and establish their commitments.
- Planning for hidden costs can result in improved relations and communications between the procuring organization and the contractor by allocating the needed user and management resources in a timely manner. For members of a project team to work together more efficiently on a project, it is necessary that each member understand his/her role in the project and the overall activities of the project.
- Including the critical resources in the risk prone activities can result in improved risk management.
- Improve the chances for successful software projects: Estimating and providing the necessary resources for a software project may help in avoiding the negative impacts identified in the findings and may improve the chances for developing better quality systems; systems within budget, on schedule, that meet the user requirements.

7.0 Conclusions

A formal, institutionalized, software acquisition project management process model will require that the organization plan all aspects of the acquisition, manage software requirements, track project team and contractor team performance, manage the project's cost and schedule baselines, evaluate the products and services, and successfully transition the software to its support organization. Especially critical to the success of a contracted project is the understanding of how to identify and manage risks. The cost and risk impacts of not improving software acquisition processes are substantial. 

References

1. Albrecht, A.J and J.E. Gaffney Jr. (1983). Software Function, Source Lines of Code, and Development Effort Prediction. *IEEE Transactions on Software Engineering*, SE-9, no 6, November, pp639-648.
2. Bailey, J. W, and V. R A. Basili (1981). Meta-Model for Software Development Resource Expenditure, *Proceedings, Fifth International Conference on Software Engineering*, IEEE/ACM, NBS, March, pp106-116.
3. Baumert, J.H. and M. S. McWhinney (1992). *Software Measures and the Capability Maturity Model*. Software Engineering Institute, CMU/SEI-92-TR-25, DTIC Number ADA257238.
4. Carleton, Anita D. and Robert E. Park (1992). *Software Measurement for DoD Systems: Recommendations for Initial Core Measures*. Software Engineering Institute, CMU/SEI-92-TR-19, DTIC Number ADA258305.
5. Collier, Bonnie, Tom DeMarco and Peter Fearey (1996). *A Defined Process for Project Postmortem Review*. *IEEE Software*, Vol. 13, No. 4, July 1996, pp. 65-72.
6. Cooper, J., Fisher, M., and Sherer, S. W., Editors, (1999). *Software Acquisition Capability Maturity Model (SA-CMM) Ver.1.02*, Technical Report CMU/SEI-99-TR-002 ESC-TR-99-002
7. Cowderoy, A.J.C and J. O. Jenkins, (1989). *New Trends in Cost-Estimation in Measurement for Software Control and Assurance*. Elsevier Applied Science, New York.
8. DeMarco, Tom. (1982). *Controlling Software Projects: Management, Measurement, and Estimation*, Yourdon Press, New York, NY.
9. Eisner, Howard (1997). *Essentials of Project and Systems Engineering Management*, John Wiley and Sons, Inc, New York, NY.
10. Fenton, Norman (1994). Software Measurement: A Necessary Scientific Basis. *IEEE Transactions on Software Engineering*, Vol. 20, No. 3, March, pp. 199-206.
11. Haddad, Maliha (1999). *The Hidden Cost of Software for the Contracting Organization. Dissertation*, George Washington University. Washington, DC.
12. Haddad, M. La Salle, A. and Ribiere, V. (2000). "Using the SA-CMM as a Tool for Estimating the User and Management Costs for Software Acquisition Projects," *Proceedings of Americas Conference on Information Systems (AMCIS 2000)*, Long Beach, CA.
13. Haddad, M. and La Salle, A. (2000). "Measuring Hidden User-Costs in Software Acquisition Projects," *Proceedings of Intl. Council on Sys. Engineering Conf. (INCOSE)*, Reston, VA.
14. Haddad, M., La Salle, A. and Harrald, J. (1999), "Software Subcontracting and the Cost to the Mature Organization," *Proceedings of AIS'99*, Milwaukee, WI.
15. Hihn, H. and Habib-Agahi, H. (1991). Cost Estimation of Software Intensive Projects: A Survey of Current Practices. *Proceedings of the 13th International Conference on Software Engineering*, Austin, TX, 13-17 May, pp. 276-287.
16. Jones, Capers (1997). *Applied Software Measurement: Assuring Productivity And Quality*. McGraw-Hill, New York, NY.
17. Jones, Capers (1994). *Assessment and Control of Software Risks*. Yourdan Press, Prentice-Hall, Inc. Englewood Cliffs, NJ.
18. Jones, Capers (1996). *Patterns of Software Systems Failure and Success*. International Thomson Computer Press.
19. Kitchenham, B., Pfleeger, S. and Fenton, N. (1995). Towards a Framework for Software Measurement Validation. *IEEE Transactions on Software Engineering*, Vol. 21, No. 12, December 1995, pp. 929-944.

20. Nielsen, J and Miller, A. (1996). Selecting Software Subcontractors, *IEEE Software*, Vol. 13, No. 4, July, pp. 104-109.
21. Paulk, M.C., Curtis, M. B. Chrissis, and C.V. Weber (1993). *The Capability Maturity Model for Software*, Version 1.1, Software Engineering Institute, CMU/SEI-93-TR-24, February.
22. Paulk, M.C., Weber, C. V., Curtis, B. and Chrissis, M.B., (1995), *The Capability Maturity Model: Guideline for Improving the Software Process*, Addison-Wesley Publishing Company.
23. Pfleeger, Shari L. (1993). Lessons Learned in Building a Corporate Metrics Program. *IEEE Software*, May, pp. 67-74.
24. Putnam, L.H. (1978). A General Empirical Solution to the Macro Sizing and Estimating problem. *IEEE Transactions Software Engineering*, vol. 4, no. 4, pp. 345-361.
25. CMMI Product Team (2002). Capability Maturity Model Integration (CMMI) Version 1.1, Carnegie-Mellon Software Engineering Institute, March.
26. Statz, Joyce, and Susan Tennison (1995). Getting Started with Software Risk Management. *American Programmer*, Vol.8, No.3, pp. 23-30 March.

Notes

Notes