A New Approach for Automating Organizational Capability Maturity Models using UML

Abdulhameed Alelaiwi, Saad Alruished

Department of Software Engineering, College of Computer & Information Sciences, King Saud University, P.O. Box 51178, Riyadh 11543, Kingdom of Saudi Arabia
aalelaiwi@ksu.edu.sa

Abstract: In modern society, organizations are forced to automate their processes and modify their behavior in order to achieve their planned goals. One efficient method used to achieve these goals is the software process method. Currently, the Capability Maturity Model (CMM) is used extensively to measure the maturity level of organizations. In this paper, we propose a new and efficient approach for automating Organizational Capability Maturity Model (OCMM) systems using the Unified Modeling Language (UML).This proposed approach covers all functional and non-functional requirements of an organization. Further, it helps organizations in different aspects of software houses to increase their maturity levels and make rapid and reliable assessments at low cost. The results of system development tests conducted indicate that such system could help assessing the maturity level for organizations.

[Abdulhameed Alelaiwi, Saad Alruished. **A New Approach for Automating Organizational Capability Maturity Models using UML.** *Life Sci J* 2014;11(9s):38-44]. (ISSN:1097-8135). http://www.lifesciencesite.com. 6

Keywords: Software development houses, UML, search services, organizational capability maturity model, software process assessment

1. Introduction

Software Process Methodologies (SPMs) such as the Capability Maturity Model (CMM) provide optimal solutions for improving the quality of organizational modularity. Software process improvement affects the impact factor of organizations and the evaluation of a software development process' maturity requires the measurement of specific attributes. CMM is used to measure the capability of companies or institutions with common interests. Within an organization, evaluation of a software process is conducted in stages,and measurement of the maturity of each process is conducted separately (Humphrey, 1988).

CMM can be used for software process assessment and capability evaluation. They are different from each other in terms of objective, motivation, and final results. For assessment of an organization's software process capability, capability evaluation techniques are used. This also identifies the performance of a company on the basis of work allotted. On the other hand, software process assessment plays an important role in improving the process capability of organizations. Thus, these types of assessments are used internally. Application of this model in all phases is sometimes a difficult task, especially when the process involves the use of different modules in different phases.

Proper execution of the various steps comprising a CMM is tedious and expensive. Consequently, in this paper, we propose a model for automating the organizational system, called the Organizational Capability Maturity Model (OCMM), in which different Unified Modeling Language (UML)models are applied within the development house on the basis of the criteria provided. The results of system development tests conducted indicate that our system could be one of the best automated systems for assessment of the maturity level for organizations. To control over-budgeting and scheduling, the OCMM evaluates organizations without any intervention from users outside the system.

The remainder of this paper is organized as follows. Related work is presented in Section 2.In Section 3, the CMM and its various levels are discussed. In Section 4,the proposed OCMM approach is outlined and discussed in detail. Finally, conclusions are presented in Section 5.

2. Related Work

Various researchers are currently working on CMM and its related models with the objective of providing the best solution for customers as well as for organizations by improving their maturity levels and efficiency. Implementation of measuring activities within organizations such as software organizations is also being carried out.

To understand the problems that can occur at the end of the measuring process, Gang and Jie created the specialized Software Process Capability Maturity Model (SP-CMM). The model is based on five consecutive levels: initial, tentatively, defined, compulsive, and optimized. Each level in the SP-CMM has its own criteria and responsibility. The model helps organizations with activities such as evaluation of their measurement process situation and control process improvement orientation. It can also provide guidance to an organization for getting to a higher maturity level (Mei and Ding, 2010).

April and Abran (2009)presented software maintenance information to aid in the identification of process and measurement practices of products in the Software Maintenance Maturity Model. For observation of maturity level 3, they presented eight advanced measurement practices. On assigning a target to each, they also monitor the progress towards the target after collecting data. Service level agreements are also defined, monitored, and tracked as stipulated in the maintenance contract. For maturity level 4, key process and product activities are created on the basis of quantitative aspects. To manage the achievement of the objectives, all these characterizations are designed and analysis and review of attributes for performance quality and application software performed. In addition, four advanced measurement practices are also defined for this level (April and Abran 2009).

Strutt et al. (2006) developed the Design safety Capability Maturity Model (DCMM), which enables identification of maturity level, scoring methods, and key processes—all important for safe operation of the organization. DCMM comprises twelve key safety management processes and maturity levels defined with associated organizational characteristics. They include initial, repeatable, defined, managed, and optimized levels. The offshore sector, which uses applications for environmental risk management, can gain advantages from DCMM as the main focus of environmental risk management is on installation and operation with environmental permits. Powerful reality checks on cooperate risk management statement are provided over plant level risk management capability (Strutt, Sharp et al. 2006).

For the management of service level processes, organizations adopt the IT Service Capability Maturity Model (IT Service CMM). Daneshgar et al. (2008) presented a conceptual modeling language (called the Awareness Net modeling language) and incorporated it into the IT Service CMM for identification of the collaboration requirements of different entities; thereby, helping IT organizations that are using the IT Service CMM to increase the quality and consistency of the services they provide. The Conceptual Model Quality Framework (CMQF) is used to evaluate and assess the suitability of modeling languages. Using a process of repetitive assessment, it enables improvement, correctness, acceptable quality, and effectiveness of conceptual modeling languages when applied to the IT Service CMM. Correct process models are produced by the conceptual modeling language for the related processes defined by the IT Service CMM. The framework also covers collaboration and knowledge sharing by the actors involved in the service process (Daneshgar, Ramarathinam et al. 2008).

3. Capability Maturity Model (CMM)

The Department of Defense (DOD) provides guidelines for Software Process Management (SMP) as an aid to evaluate the performance of contractors. The Software Engineering Institute was very active in the development of software maturity for software development organizations in the 1980s. They developed CMM for the evaluation of capabilities and assessments of software processes and facilitated judgment of the maturity of the software processes in organizations. CMM defines various levels, including the styles and quality of software production practices (Agarwal, Tayal et al. 2010). The levels defined in CMM are outlined in the ensuing sub-sections.

3.1 Level 1 (Initial)

A software process may be adhoc and disorganized. If an organization depends on an individual's effort or its processes are not defined and documented, then it is classified at level 1. Organizations at this level can benefit most by improving project management, quality assurance, and change control (Dion, 1992).

3.2 Level 2 (Repeatable)

This level covers basic management practices such as tracking costs, schedules, and establishment of functionalities, but not the procedure for doing it. To ensure project success, some work that has already been done can be repeated at this level. In this level, the characteristics of a process can be project commitments (that are realistic and based on experience with similar projects), costs, schedules, solution to problems, configurations for controlling mechanisms, and software project standard (Keane, 2011).

3.3 Level 3 (Defined)

In organizations, software processes include management and engineering procedures. This level of CMM covers both,with the aim of achieving ISO 9000 standard. These procedures are well defined, documented, and integrated with each other. For software development and maintenance, in-project documentation and the approved version of the organization's process are used (Chrissis, Konrad et al. 2003).

3.4 Level 4 (Managed)

This level focuses on software metrics such as product and process metrics. Product metrics measure the characteristics of the product under development, such as its size, reliability, time complexity, and understandability. Conversely, process metrics reflect the efficacy of the processes being used, such as average time for correction of defects, productivity, average defects per hour found during inspection, and average failure defects during testing. The results of process measurements are used for the evaluation of project performance rather than to improve the processes (Paulk, Weber et al. 1994).

3.5 Level 5 (Optimized)

This level defines the state of an organization committed to continuous process improvement. The process improvement covers budgeting and planning. Organizations commit to identifyingthe weaknesses and strength of a process to make it secure from defects. They also use best software engineering and management practices (Batten, 2008). Figure 1 shows the levelscomprising the CMM.

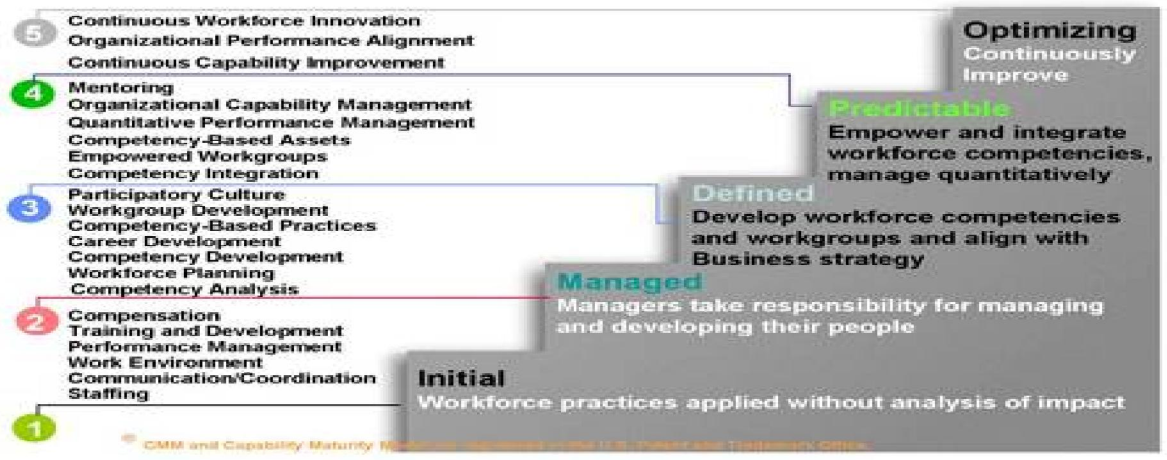
5
Continuous Workforce Innovation
Organizational Performance Alignment
Continuous Capability Improvement
4
Mentoring
Organizational Capability Management
Quantitative Performance Management
Competency-Based Assets
Empowered Workgroups
Competency Integration
3
Participatory Culture
Workgroup Development
Competency-Based Practices
Career Development
Competency Development
Workforce Planning
Competency Analysis
2
Compensation
Training and Development
Performance Management
Work Environment
Communication/Coordination
Staffing
1
Optimizing
Continuously improve
Predictable
Empower and integrate workforce competencies, manage quantitatively
Defined
Develop workforce competencies and align with Business strategy
Managed
Managers take responsibility for managing and developing their people
Initial
Workforce practices applied without analysis of impact

Figure 1: Software Engineering Institute Levels of Maturity (Mellon, 2012)

4. Proposed Approach

OCMM arose out of the fact that in many Software Development Houses (SDHs)there is a need for a model that can be applied in projectsto reduce incidents of the schedule and budget plans being exceeded. OCMM helps to solve the above problems by conducting evaluations without any external system intervention. Thesolutions available in the Software Engineering Competence Center (SECC) do not provide any service for evaluation level and search services for SDHs.

The major problems being faced by customers are the following:

- Difficulty determining the best SDH.
- Time wasteddetermining the appropriate SDH.
- SDHs not following the correct path because they are focusing only on the solution, at the expense of quality.

Bothorganizations and SDHs are affected by the above problems, which also resultin time and effort being wasted during assessments and the need to hire many employees because of the manual tasks involved.

We believe that our proposed OCMM approach is efficient and has the capability to provide accurate assessments for SDHs such that they can becomethe best in their field. In order to registerfor their evaluations, SDHs first request a registration level and the system respondsby checkingtheir eligibilityand sending an evaluator.

Our proposed approach providesthe following advantages:

- Automatic assessment process.
- Identification of the level of an SDH.
- Reduction in the cost, time, and effort required for the assessment process.
- Reduction in the amount of manpower required.
- Impartiality in the assessment process.

Two types of requirements are defined usingthe UML standard: functional and non-functional. In the following section, we explain these requirements in details.

4.1.1 Functional Requirements

The following are the functional requirements comprising the proposed model.

- OCMM evaluates and assists on the basis of updatable criteria.
- SDHs are required to register with the system by providing required information with usernames and passwords.
- Using their accounts, SDHs create requests for evaluations.

- Management receivessuch requests and provides a specific date to prepare documents and evidence for assessment.
- Following approval, management sends an order to an evaluator to begin evaluating the particular SDH.
- If the SDH is ready the evaluator enters the new evaluation into the system; otherwise,the evaluator gives them a new date.
- If the SDH has been evaluated before and is now planning to change status from one level to another,the database is also updated following the evaluation process.
- SDHs are provided with the requirements that need to be satisfied in order to get to the next level of the CMM.
- After a specified period (usually,three years), the evaluation is repeated to determine if the SDH has improved,is still at the same level, or has retrogressed.

4.1.2 Non-Functional Requirements

The following are the non-functional requirements that are necessary for the proposed system:

- **Reliability:** The system should be reliable, with a maximum failure rate of only once per month and a repair time not exceeding one hour. Further, the system's recovery time should not exceed ten minutes in 99% of cases of system failure.
- **Availability:**The system should be available on a 24/7 basis.
- **Performance:** The system must allow access by different users from different locations and the waiting time for any query should not exceed one second in 95% of cases.
- **Security:** The system must provide maximum data security.
- **Usability:** Training of new employees should not take more than ten days (three hours per day).
- **Portability:** The system must be executable on every operating system.

4.2 Use-Case Model for OCMM

Use-case modeling is a text based systems analysis and design tool. With the help of use-cases, developers can understand howa system is implemented(Gemino and Parker, 2009). In software and systems engineering, a use case is a detailed description of an action between actors and the system cases (Bittner and Spence, 2006). A use-case diagram of our proposed OCMM is depicted in Figure 2.

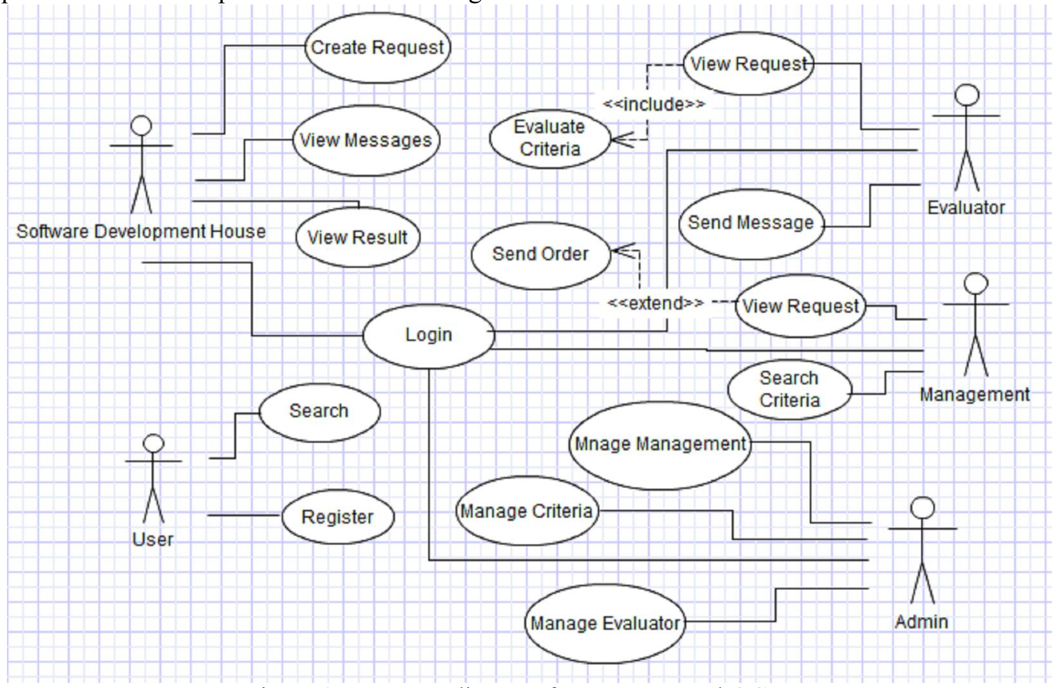

Figure 2: Use-case diagram for our proposed OCMM

In our use-case diagram,the actors are as follows: User (anyone who can access the OCMM details), SDH (customer who wishes to assess his/her organization to determine its CMM level), Evaluator (person or persons responsible for evaluation of SDHs), Supporter (responsible for maintaining network, security, queries, taking backups of the system, and ensuring efficient system operation), Admin (administrator of the system), and Management (entity that receives the request for evaluation from SDHs, approves it, and then sendsthe order to the Evaluator).

4.3 State Chart Diagram

In Unified Modeling Language (UML) development, state chart diagrams arevery important for software design. These diagrams explain all the possible states and transitions among the various states of an object. With the combination of a class diagram and a collaboration diagram, a highly structured state chart diagram can be generated (Xiaolong and Kerong, 2011). A state chart diagram of the proposed OCMM approach is depicted in Figure 3.

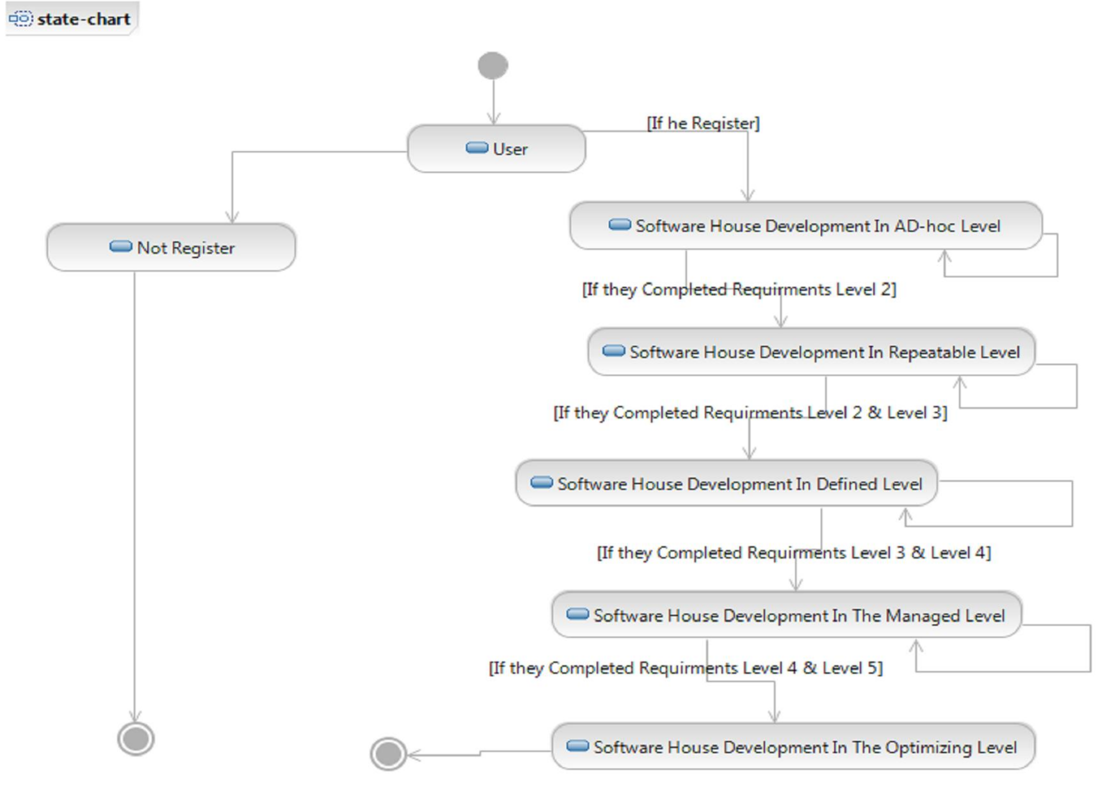
state-chart
User
[If he Register]
Not Register
Software House Development In AD-hoc Level
[If they Completed Requirments Level 2]
Software House Development In Repeatable Level
[If they Completed Requirments Level 2 & Level 3]
Software House Development In Defined Level
[If they Completed Requirments Level 3 & Level 4]
Software House Development In The Managed Level
[If they Completed Requirments Level 4 & Level 5]
Software House Development In The Optimizing Level

Figure 3: State Chart Diagram for OCMM

4.4 Design Classes

A class is a group of objects that have similar properties, common behaviors, a common relationship with other objects, and common meanings. It is used for general modeling of the application and modeling translation into programming code (Ma, Zhang et al. 2011). A class diagram represents the structural part of the system.It plays an important role in the system development lifecycle as it provides a clear view of the system to the developers and implementers when they are converting class diagrams into code. Designers of the system can also benefit from the class diagrams as they are responsible for proposing design features of the objects (Lindquist, 2010). A class design diagram of the proposed OCMM is depicted in Figure 4.

4.5 Model View Controller (MVC)

MVC is a computer interface methodology that represents information separately without user interaction (Reenskaug and Coplien, 2009). It consists of application data, business rules, and controller conciliate input, that convert into commands for view or model (GuangChun, Lu et al. 2003). View or model can be any type of output, such as a diagram or chart. It consists of three parts: Controller, Model, and View. Controller makes and combines models, views, and all other components. Model represents the structure of the application. View displays the model in a format that is understandable for the end user (Buschmann, Henney et al. 2007). An MVC diagram of the proposed OCMM is depicted in Figure 5.

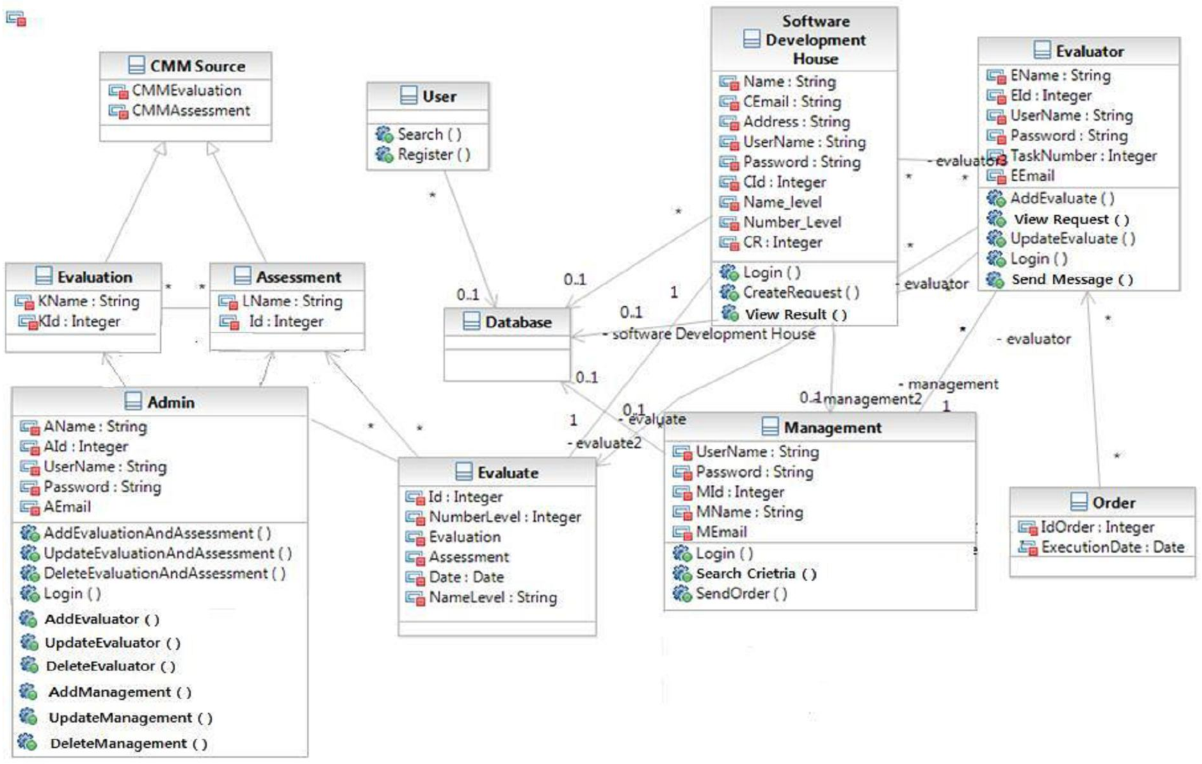

Figure 4: Class Design diagram for OCMM

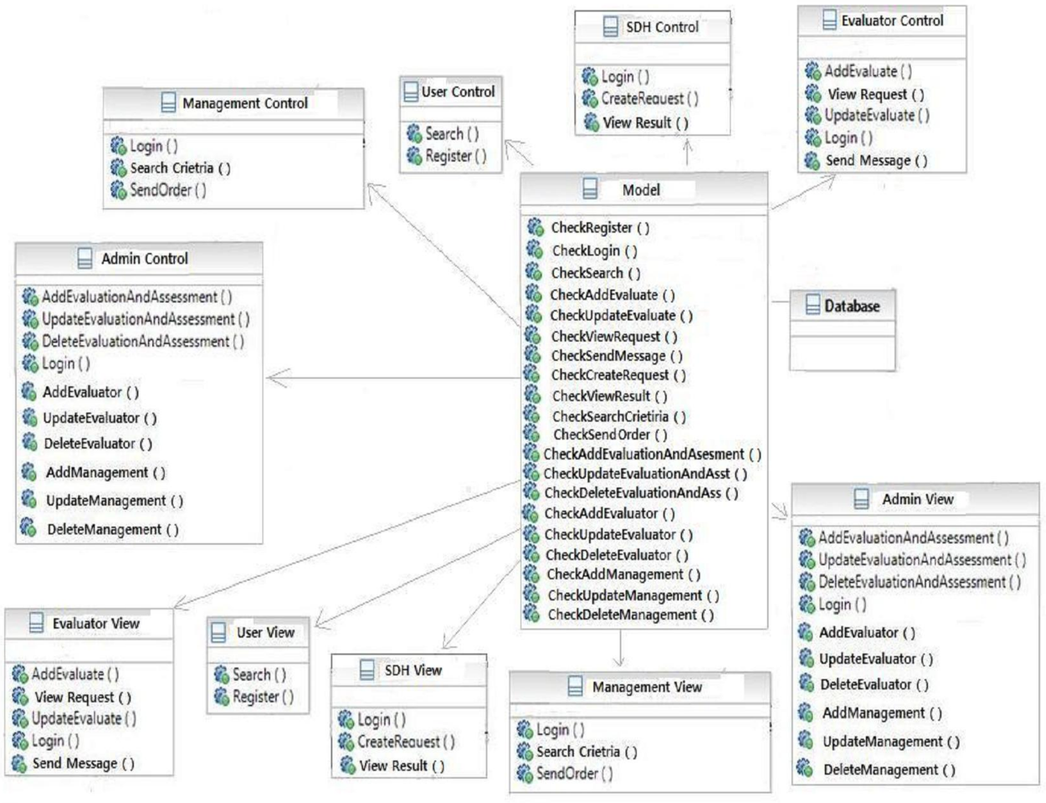

Figure 5: MVC diagram for OCMM

5. Conclusion

Software assessment processes for organizations are currently expensive and time consuming. Consequently, in this paper, we proposed and presented an automated Organizational Capability Maturity Model (OCMM) system using UML models. Different levels of CMM were discussed and presented while primarily focusing on its importance to software houses. The proposed system can help organizations to make rapid assessments at low cost.The OCMM will help organizations to achieve excellent focus in their field.

Acknowledgements:

The authors extend their appreciation to the Deanship of Scientific Research at King Saud University, Riyadh, Saudi Arabia for funding this work through the research group project No RGP-VPP-318.

Corresponding Author:
Dr. Abdulhameed Alelaiwi
Department of Software Engineering
College of Computer & Information
Sciences, King Saud University E-mail:
aalelaiwi@ksu.edu.sa

References

1. April A,AbranA. A software maintenance maturity model (S3M): Measurement practices at maturity levels 3 and 4. Electronic Notes in Theoretical Computer Science 2009;**233**(0): 73-87.
2. Agarwal B.B.,Tayal S.P. et al. Software engineering and testing, Jones and Bartlett Publishers. 2010.
3. Batten L. CMMI 100 success secrets: 100 most asked questions: The missing CMMI-DEV, CMMI-ACQ project management and process guide, Emereo Publishing. May 2008.
4. Bittner K., Spence I. Managing iterative software development projects. Addison-Wesley Professional. 2006.
5. Daneshgar F., Ramarathinam K. et al. Representation of knowledge in information technology service capability maturity model (IT Service CMM). Second International Conference on Research Challenges in Information Science, 2008. RCIS 2008.
6. Dion R. Elements of a process-improvement program. IEEE Software 1992; **9**(4): 83-85.
7. Buschmann F.,Henney K. et al. Pattern oriented software architecture volume 5: On patterns and pattern languages, Wiley. 2007.
8. Gemino A., Parker D. Use case diagrams in support of use case modeling: Deriving understanding from the picture. Database Management 2009; **20**(1): 1-24.
9. GuangChun L., Lu W.et al. A novel web application frame developed by MVC. SIGSOFT Softw. Eng. Notes 2003;**28**(2): 7.
10. Humphrey W. S. Characterizing the software process: A maturity framework. Software, IEEE 1988;**5**(2): 73-79.
11. Keane. Outsourcing & the capability maturity model (CMM). White Paper. 2011.
12. Lindquist M. Enterprise software architecture design: System representation through UML diagrams. 2010.
13. Chrissis M. B.,Konrad M. et al. CMMI: Guidelines for process integration and product improvement. Addison-Wesley Professional. March 2003.
14. Paulk M. C., Weber C. V. et al. The capability maturity model: Guidelines for improving the software process. Addison-Wesley Professional. 1994.
15. Ma Z. M., Zhang F. et al. Fuzzy information modeling in UML class diagram and relational database models. Applied Soft Computing 2011;**11**(6): 4236-4245.
16. Mei Y., Ding J. Software measurement process capability maturity model.Second International Conference on Computer Modeling and Simulation, 2010. ICCMS '10.
17. Mellon C. People CMM evolutionary improvement path. 2012.
18. Strutt J. E. Sharp J. V. et al. Capability maturity models for offshore organisational management. Environment International 2006;**32**(8): 1094-1105.
19. Reenskaug T., Coplien J. O. The DCI architecture: A new vision of object-oriented programming. 2009.
20. Xiaolong W.,Kerong B. A transition method from UML class diagram and collaboration diagram to statechart diagram. Journal of Wuhan University of Technology (Information & Management Engineering) 2011;**33**(6): 940-944.

10/10/2013