International Journal of Computer (IJC)
ISSN 2307-4523 (Print & Online)

http://ijcjournal.org/

Crowdsourced Machine Learning Based Recommender for Software Design Patterns

Sunbul Sajid Khowaja[a]*, Dr Qasim Ali[b], Erum Hamid[c], Rajesh Kumar[d], Gul Bano[e], Jatendar Dharani[f], Isma Farah[g], Zainab Umair[h]

[a,b,e,f,g,h]*Software Engineering Department, Mehran University of Engineering and Technology, Jamshoro 76020, Pakistan*

[c]*Beijing University of Posts and Telecommunication, Beijing, China*

[d]*Hamdard University, Karachi, Pakistan*

[a]*Email: sunbul.khuwaja@gmail.com,* [b]*Email: qasim.arain@faculty.muet.edu.pk*

[c]*Email: zubedierum@hotmail.com,* [d]*Email: rajesh.kumar@hamdard.edu.pk*

[e]*Email: asmagul_gh@yahoo.com,* [f]*Email: jatendurdharani@hotmail.com*

[g]*Email: isma.farah@faculty.muet.edu.pk,* [h]*Email: zainabumair11sw38@gmail.com*

Abstract

Software technology has become an essential part of human lives today. The role of software Engineers in making this technology as success is very fundamental. In software Engineering, the toughest stage is to design software as there is no particular rule or formula to covert requirements into design representation. A designer designs software using skills, critical thinking ability and previous experience only. To make this process easy, the design patterns came into existence which are the solutions that can be used repetitively to solve design problems. There have been several pieces of research presented regarding design Patterns but it is hard to find research regarding how the patterns are perceived and used in industries today and what nature of application uses which specific patterns. This paper uses a crowdsourced approach to acquire the finest practices that are being used in industries today including which quality attributes are affected most by the implementation of these patterns and which patterns are suitable for what type of applications. It also uses a machine learning supervised algorithm (Matchbox Recommender) to predict suitable design pattern for different nature of applications.

Keywords: Crowdsourcing; Design patterns; Machine Learning; Software Quality; Matchbox recommender; T-test.

---

* Corresponding author.

1. Introduction

The 21st century is known to be the golden era for computer technology as it has covered almost every single part of the necessities today. To continue this comfort for its users, software developers have left no stone unturned to improve this technology. To achieve fruitful results, it is necessary to make sure that the software achieves its intended purpose and quality within feasible time and budget [10]. SQA (Software quality assurance) is a technique that ensures the achievement of desired quality used with SDLC (software development lifecycle) steps [18]. Among SDLC steps the design step is perceived as more complicated because there is no algorithmic rule to transform theoretical requirements into diagrammatic design. The only approach can be used is to use some techniques, previous experience, and Design Patterns. Design Patterns are repeatedly occurring solutions to design problems. Or it can be said that design patterns convert problem domain into the solution domain [17].

The purpose of this research was to investigate documented design problems and how their implementation influences software products' quality. The technique used to do it is known as the Crowdsourced Design method. Crowd-wisdom is a type of crowdsourcing practice. Crowdsourcing is a method to involve various participants into a common chore to get a collective result and crowd wisdom is crowdsourcing by using a questionnaire survey approach [12]. This research has been mainly focused on the following main objectives:

- To identify which design patterns are best suitable for what type of applications (i-e web app, android, hybrid, etc.) to achieve the desired quality application,
- To analyze that which design patterns (i-e structural, Behavioral or creational) effect which quality attributes the most like maintainability, reliability or more,
- To what extent (in percent) the design patterns categories affect quality attributes of the end product. To recognize that which design patterns are more frequently being used in industries today than others?

The previous work has many limitations some of them are discussed below: The comparison of Design Patterns exists but "which are the most feasible ones or used commonly in industries" is still unclear. Also, Most of the new designers get confused of what patterns should be chosen which would provide better quality than others. In the initial stage of development, the project's final appearance is usually vague so this research would solve this issue by recommending patterns for particular end-product's preferred quality attributes. After this recommendation the designers or developers will have clear idea that what kind of quality or types will be seen in systems final version. In this work, we examine the facts that can affect the quality of software and we find out commonly used design patterns in the software industry. Also, we propose some particular patterns that can be used while designing a particular type of application. We have converted our research questions into survey questions. After that, we have used a Recommender (which uses Supervised Machine learning algorithm) to predict the best suitable patterns required to design a particular type of application. Machine Learning makes the machine to learn the environment itself without defining it separately [22]. The outcomes show the listed contributions that can help to improve the selection of suitable design patterns for the required product and helps new designers in understanding suitable patterns.

2. Related Work

As design pattern is a vital part of software development, various researches focus on analyzing the ways of choosing the suitable design pattern from a pool of patterns. In 2016, an automated system based on a Fuzzy c-means unsupervised learning technique was proposed to select and recommend the most suitable design pattern to in-experienced designers [1]. It works by determining the resemblance of multiple objects whereas our research uses a supervised machine learning approach and works on a real dataset. In 2018 M Noman Riaz performed a comparative study about the influence of design patterns on software quality. He has stated that there are mainly four attributes (fault proneness, change proneness, and evolution, maintainability, and performance) that are more focused in the literature while any consensus is not present on their effect. He has concluded in his paper that there is a negative impression of the design pattern on three attributes (evolution, proneness, and maintainability). Also, the results of performance attribute and change proneness were varied from one another so the researcher declares it as challenging to make the judgment for these attributes [4]. These were limited quality attributes while our research works on all the quality attributes that were defined by ISO 9126. Aslı Sarı, Ayşe Tosun and Gülfem Işıklar Alptekin in [5] performed a literature review on crowdsourcing in software engineering. They highlighted the crowdsourced design as a special type of crowdsourcing and reviewed certain crowdsourced design platforms like Topcoder and 99designs. They also stated that yahoo Answers is one of the popular platforms of crowd wisdom. We have used google form and survey monkey for our crowdsourced based data collection. In [7] the authors Relate pattern with quality of code. They interpreted that the existence of design patterns pulls down the figure of code smell occurrences. Moreover, they indicated that State Strategy, Factory-method, and Adaptor-command are less probable to be relative to the smelly code. We have identified patterns effect on individual quality attributes rather than evaluating the overall quality of the product. In [19, 20] the system to recommend suitable design patterns was introduced. It covered all GOF patterns. It took a textual description of a problem as input and guided a designer to select an appropriate pattern. Whereas our research takes the real data as input, processes using machine learning and then recommends appropriate design patterns. In [22] the investigators explained the use of Machine learning algorithms in recommender systems. They identified the trends which are being used in machine learning for recommender systems. The activities which have been performed in our research were referred from this paper. In [25] the Research predicts re-tweets of an original tweet by using a matchbox recommender algorithm and our research predicts suitable design patterns to be used for different application types by using the same algorithm but different factors.

3. Methodology

The research was comprised of the following steps as shown in figure 1.

3.1. Questionnaire Design

The Research assumptions which contributed towards the design of a questionnaire were made at the start that was:

R-1: Some design patterns are more commonly used in design decisions to produce a high-quality product while others are very rare.

R-2: Among all quality attributes, the maintainability attribute is affected the most from design pattern decisions.

R-3: The choice of best design pattern for the product reduces higher the chances of getting a poor quality.

R-4: Some design patterns are only feasible for certain types of applications and might be infeasible for other types of applications. The application types we consider for R4 are Web app, Mobile app, and Hybrid app.

The survey questionnaires were designed by interviewing with professionals, getting guidance from related work, following tutorials and based on previous field understanding. All the questions of surveys were sensibly designed to fulfill the goal of our research. The online forms were created and fetched to be solved by designers of software industries. The survey questionnaire links are: "https://www.surveymonkey.com/r/FNR6JL7" and "https://forms.gle/cPmVDDisX8TVhQ6w9".

3.2. Responses Taken

The surveys were conducted, then all results were evaluated and conclusions were made in accordance. We clarified the respondents about the aim of our research. The survey links were posted through E-mail, Facebook and WhatsApp. After the completion of the survey, the answers were validated by using a t-test.

3.3. Data Pre-Processing

Consequently, the data was pre-processed to be validated and then to be uploaded on the cloud. Pre-processing is an important step before machine learning is applied. It discards out-of-range values, missing values, and impossible data combinations.

The pre-processing is further divided into the following steps:

3.4. Data Validation

- *Data cleaning: A practice of identifying and modifying (or eliminating) corrupt or inaccurate data from a dataset.*
- *Data editing: A practice concerning the review and correction of collected survey records.*
- *Data reduction: A makeover of statistical or categorical figures derived into a rectified, well-arranged, and shortened form.*
- *Data wrangling: A practice of converting and charting data from raw format to a more appropriate format to make it valuable for further process.*

3.5. Applying Machine Learning

MS azure [24] was used as the ML platform. It is a Microsoft supported cloud service which provides many services (like SAAS, PAAS, and IAAS). It supports many apps and programming. In Azure, The Matchbox recommender algorithm was applied to data. This recommender uses the Matchbox algorithm to train the Bayesian recommender [9].

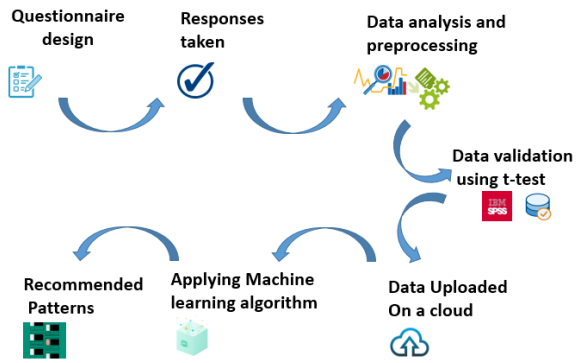

Figure 1: System Diagram

4. Results and Discussion

An online survey was conducted by leading software companies and Universities in Hyderabad, Jamshoro, and Karachi. In the survey 600 (samples) responses were selected for analysis. The analysis and validation were done using the SPSS tool. Regarding Gender, Age, roles, Experience, Location are shown in the following figures and tables.

Table 1: Gender wise responses ratio

Gender	Responses
Male	324
Female	276

Table 1 shows the Number of Male and Female respondents in our survey.

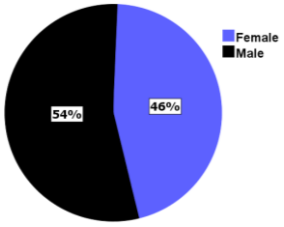

Figure 2: Gender Ratio Graph

Figure: 2 shows the percentage statistic of male and female survey respondents. 54% were male respondents and 46%were female respondents.

Table 2: Age wise responses ratio

Age Group	Respondents
21-30	450
31-40	114
41-50	36

Table: 2 shows the age-wise statistics of our respondents. It indicates that the 21-30 age group people were our major contributors.

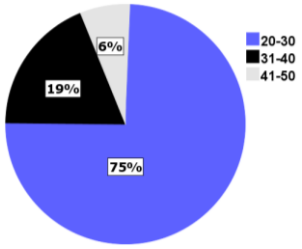

Figure 3: Age Ratio Graph

Figure 3 shows the percent of respondents WRT age groups.

Table 3: Role of wise responses ratio

Roles	Respondents
Android Developer	54
Applications Developer	12
Graphic Designer	66
Internee	114
IT officer or employee	12
Junior Software Engineer	174
Senior Software Engineer	84
Student	84

Table 3 shows the designations of our respondents. Our major contributors were junior software engineers (174) and Internees (114). The other contributors were Android and application developers, graphic designers, IT employees, senior software engineers from Hyderabad and Karachi (Pakistan). The Student respondents were from Mehran UET and the University Of Sindh, Jamshoro.

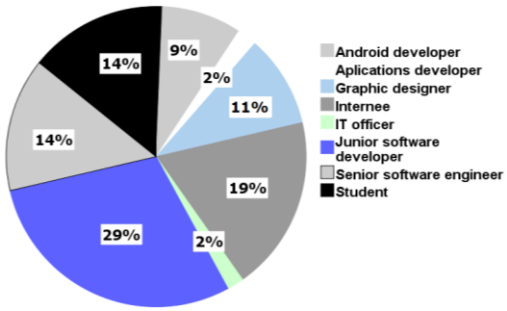

Figure 4: Roles ratio Graph

Figure 4 shows the percent ratio of all the respondents concerning their designation or roles.

Table 4: Experience wise responses ratio

Experience	Respondents
1 year	78
2-5 years	264
6-8years	36
9-10years	24
<1year	198

The table4 shows experience wise distribution of our respondents. Most of the respondents had 2 to 5 years of work experience. The second-highest share was less than one year which included internees and students. The least part was senior people which is 24 respondents only.

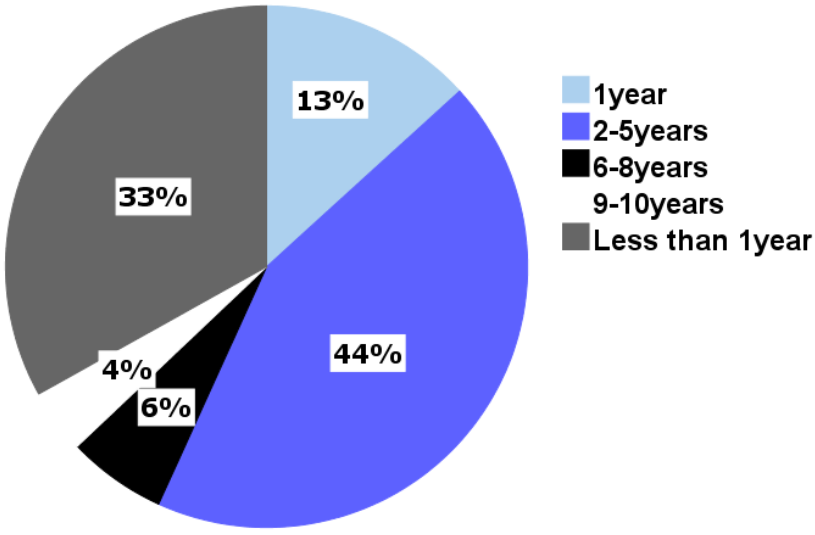

Figure 5: Experience ratio Graph

Figure 5 shows the percent distribution of our respondents in year wise experience categories.

Table 5: Location wise responses ratio

Location	Respondents
Hyderabad	204
Karachi	366
Jamshoro	198

The table 5 shows the locations of our number of respondents. The main contribution was from Karachi as it has many software industries. The second-highest number of respondents were from Hyderabad software employees and internees. The least contribution was from students and the internees of Jamshoro.

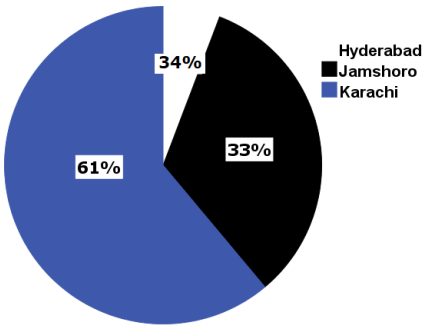

Figure 6: Location Ratio Graph

Figure 6 shows the location-wise share of respondents in percentage.

4.1. Question wise responses and validations

We asked our respondents which patterns they have ever used (or are suitable) in Mobile app (Q1), web app(Q2) and hybrid app (Q3) designing. Also, we asked questions about which quality factors were supported by what kind of design patterns (Q4 to Q8).To validate the survey answers we used the T-test method in the SPSS tool. For the T-test, we took confidence interval values like 95% and builder pattern values for comparison to other pattern values. As a rule, all the significant values of output which were less than 0.95 were counted as valid and the values greater than this were counted as invalid. Also, we have observed that the zero significant values were not exactly zero in real but were too small numbers so those were shown as zero.

The responses and validations are shown in the following figures.

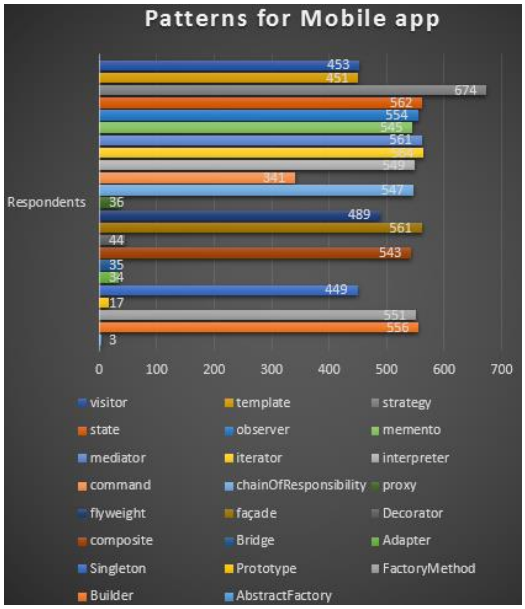

Figure 7: Mobile app suitability responses for Patterns

Table 6: Validation values For Mobile app responses

Pattern	Significance value	Pattern	Significance Value
1.Abstract factory	.837	12.Strategy	.794
2.Bridge	.837	13.Template	.808
3.Composite	.770	14.Visitor	.694
4.Chain of Responsibility	.770	15.Prototype	.808
5.Command	.000	16.Singleton	.837
6.Interpretor	.837	17.Factory	.770
7.Iterator	.677	18.Adapter	.749
8.Mediator	.770	19.Facade	.794
9.Memento	.000	20.Flyweight	.794
10.Observer	.702	21.Proxy	.770
11.State	.837	22.Decorator	.770

The figure 7. Shows the patterns which are or are not suitable to be used for mobile apps. The Table 6 shows the significant values of validation of patterns. Since all values are lesser than 0.95, all responses are valid for this question.

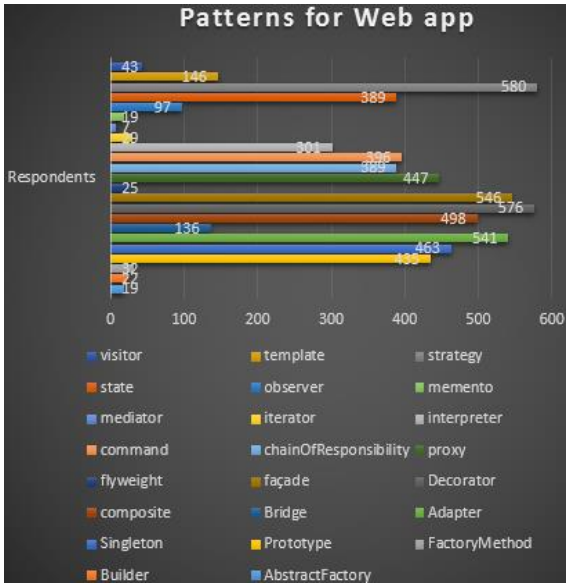

Figure 8: Web app suitability responses for Patterns

Table 7: Validation values For Web app responses

Pattern	Significance value	Pattern	Significance Value
1.Abstract factory	.000	12.Strategy	.000
2.Bridge	.000	13.Template	.000
3.Composite	.684	14.Visitor	.000
4.Chain of Responsibility	.000	15.Prototype	Invalid
5.Command	.000	16.Singleton	.000
6.Interpretor	.000	17.Factory	.000
7.Iterator	.000	18.Adapter	.000
8.Mediator	.000	19.Facade	.000
9.Memento	.000	20.Flyweight	Invalid
10.Observer	.000	21.Proxy	.000
11.State	.741	22.Decorator	.000

The figure 8 shows the patterns which are or are not suitable to be used for web apps. Table7 shows the significant values of validation of patterns. Responses for Prototype and flyweight patterns are invalid here. All other values are lesser than 0.95 so all those are valid.

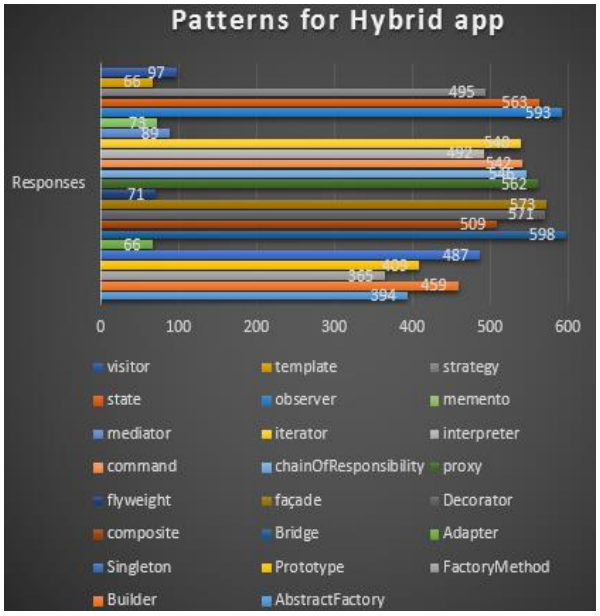

Figure 9: Hybrid app suitability responses for Patterns

Table 8: Validation values For Hybrid app responses

Pattern	Significance value	Pattern	Significance Value
1.Abstract factory	.000	12.Strategy	.770
2.Bridge	.720	13.Template	.000
3.Composite	.720	14.Visitor	.000
4.Chain of Responsibility	.677	15.Prototype	.001
5.Command	.640	16.Singleton	.000
6.Interpretor	.720	17.Factory	.002
7.Iterator	.677	18.Adapter	.000
8.Mediator	.000	19.Facade	.216
9.Memento	.000	20.Flyweight	.000
10.Observer	.640	21.Proxy	.000
11.State	.677	22.Decorator	.720

The figure 9 shows the patterns which are or are not suitable to be used for Hybrid apps. The Table 8 shows the significant values of validation of patterns for hybrid apps. Since all values are lesser than 0.95, all responses are valid for this question3.

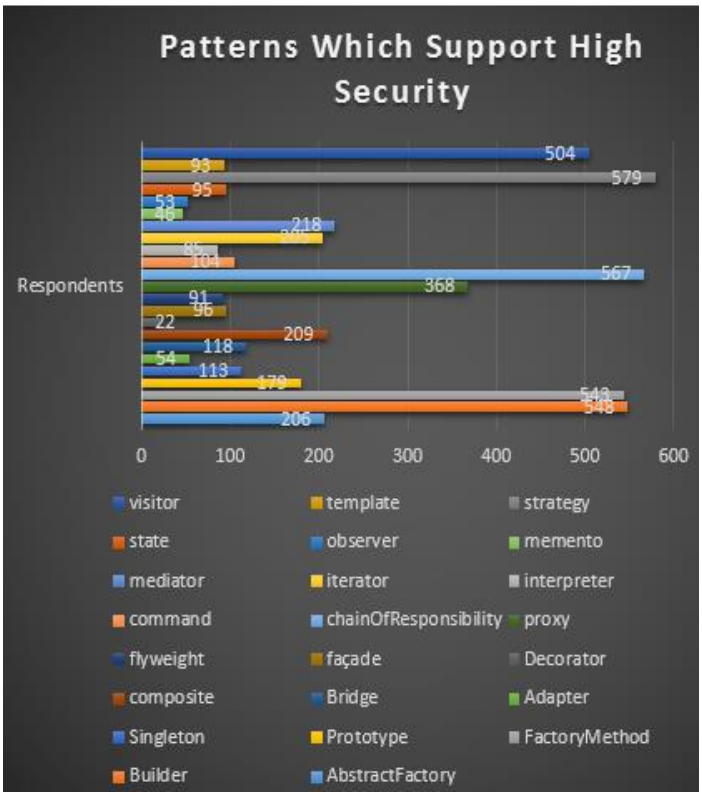

Figure 10: High-Security suitability responses for Patterns

Table 9: Validation values For High-Security responses

Pattern	Significance value	Pattern	Significance Value
1.Abstract factory	Invalid	12.Strategy	.003
2.Bridge	.000	13.Template	.000
3.Composite	.000	14.Visitor	.000
4.Chain of Responsibility	.000	15.Prototype	.000
5.Command	.000	16.Singleton	.000
6.Interpretor	Invalid	17.Factory	.000
7.Iterator	.000	18.Adapter	.720
8.Mediator	.001	19.Facade	.700
9.Memento	.007	20.Flyweight	.000
10.Observer	Invalid	21.Proxy	.000
11.State	.004	22.Decorator	.000

The figure 10 shows the patterns which High-security feature for applications. Table 9 shows the significant

values of validation of patterns. Interpreter and observer are invalid here and since all other values are lesser than 0.95, all other responses are valid for question4.

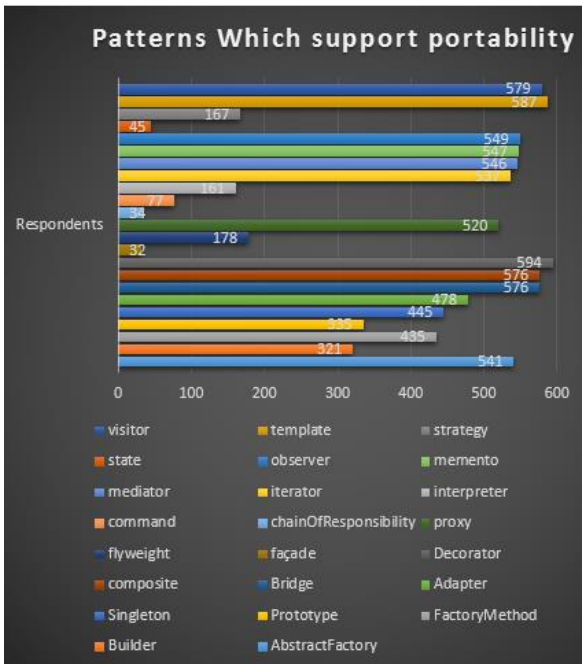

Figure 11: Portability suitability responses for Patterns

Table 10: Validation values For Portability suitability responses

Pattern	Significance value	Pattern	Significance Value
1.Abstract factory	.009	12.Strategy	.000
2.Bridge	.707	13.Template	.204
3.Composite	.000	14.Visitor	.466
4.Chain of Responsibility	.000	15.Prototype	.005
5.Command	.001	16.Singleton	.000
6.Interpretor	.809	17.Factory	.000
7.Iterator	.760	18.Adapter	.000
8.Mediator	.707	19.Facade	Invalid
9.Memento	.000	20.Flyweight	.000
10.Observer	.800	21.Proxy	.800
11.State	Invalid	22.Decorator	.000

The figure 11 Shows the patterns which support the portability feature. The table 10 shows the significant values of validation of patterns for portability. Since all values are lesser than 0.95, all responses are valid for question5.

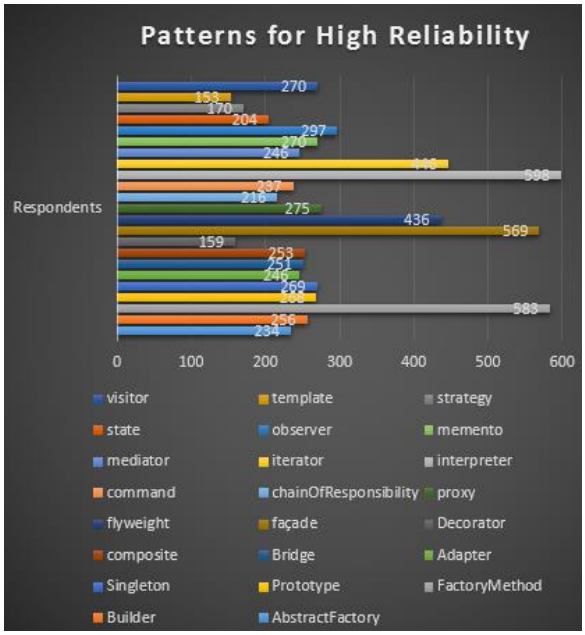

Figure 12: High-Reliability suitability responses for Patterns

Table 11: Validation values For High Reliability responses

Pattern	Significance value	Pattern	Significance Value
1.Abstract factory	.000	12.Strategy	.000
2.Bridge	.876	13.Template	.000
3.Composite	.700	14.Visitor	.000
4.Chain of Responsibility	.000	15.Prototype	.000
5.Command	.000	16.Singleton	.000
6.Interpretor	.000	17.Factory	.707
7.Iterator	.000	18.Adapter	.000
8.Mediator	.000	19.Facade	.877
9.Memento	.009	20.Flyweight	.000
10.Observer	.001	21.Proxy	.000
11.State	.000	22.Decorator	.000

The figure 12 shows the patterns which provide a high-reliability feature. The table 11 shows the significant

values of validation of patterns that provide high reliability. Since all values are lesser than 0.95, all responses are valid for question7.

From the survey, we asked about patterns that support high maintainability (Q6) and high efficiency (Q8). The answers were 100% for both so we haven't validated since it is obvious that all answers are valid (100%).

4.2. Recommender System

The recommender system was created on MS azure which recommends some best suitable patterns to users concerning application types and quality attributes. It uses all the above survey responses dataset and Splits (into 80, 20). Total three datasets were created. First contained names of patterns (patternID), factors (factorID) and their ratings (survey responses) into binary form. The second dataset had the information about pattern category (structural, behavioral and creational). The third dataset had factorID names (i-e Security, Portability, Maintainability, reliability, efficiency) and detail. The data was trained and experimented using the Train Matchbox Recommender algorithm as shown in figure 13.

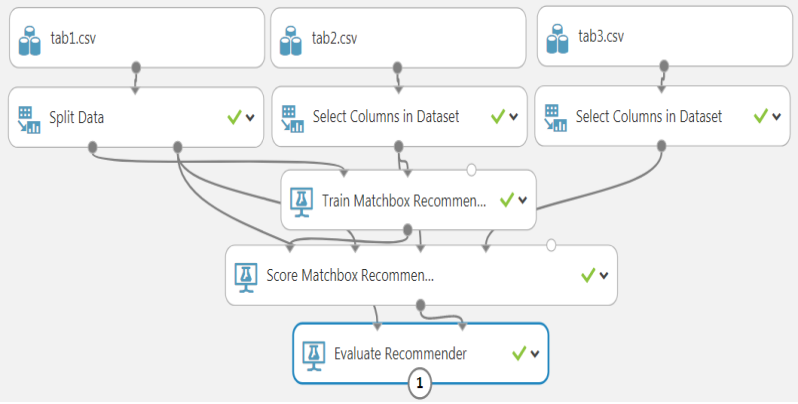

Figure 13: Experiment diagram of Design Patterns recommender

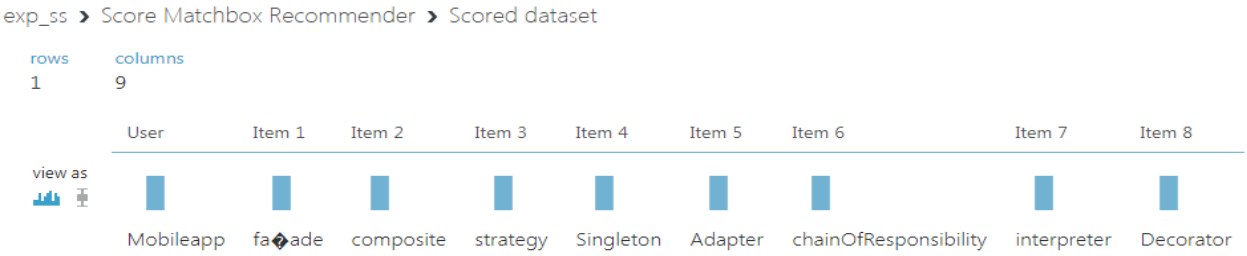

Figure 14: Recommender output for Mobile app suitability Patterns

The figure 14 shows patterns recommended from the experiment for mobile applications. These are the most suitable patterns to be considered for mobile app designing.

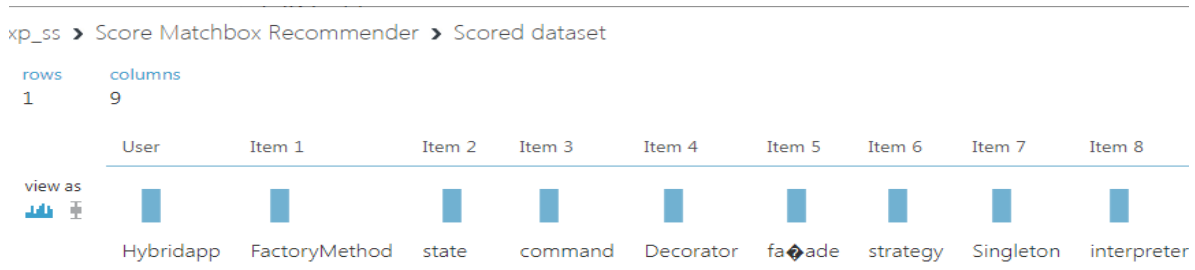

Figure 15: Recommender output for Hybrid app suitability Patterns

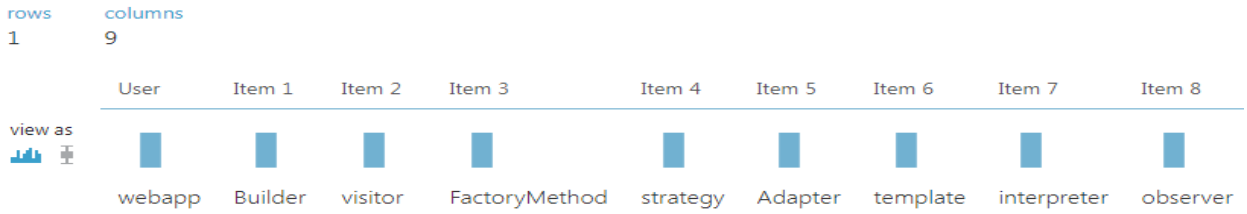

Figure 16: Recommender output for Web app suitability Patterns

Figure 17: Recommender output for Quality attributes' suitability Patterns

The figure 15 shows the eight patterns highly recommended for hybrid application designing. The figure 16 shows the eight patterns highly recommended for mobile application designing. The figure 17 shows the patterns recommended if a user prefers high maintainability, high reliability or high efficiency for their system. Or if a user prefers to keep a portability feature then corresponding patterns are more suitable. Whereas the survey responses for security preferred patterns were lesser therefore it wasn't recommended by our ML recommender system.

5. Recommendation of Patterns

Table 12 and Table 13 Show the lists of recommended patterns for particular application types and quality attributes respectively.

Table 12: App Types and Recommended patterns

Application Type	Recommended Patterns
Web App	Builder, Visitor, Factory Method, Strategy, Adapter, Template, Interpreter, Observer
Mobile App	Façade, Composite, Strategy, Singleton, Adapter, Chain of Responsibility, Interpreter, Decorator
Hybrid App	Factory Method, State, Command, Decorator, Façade, Strategy, Singleton, Interpreter

Table 13: Attributes and Recommended Patterns

Quality Attribute	Recommended Patterns
Maintainability, Efficiency, Portability	Chain Of Responsibility, Strategy, Factory Method, Builder, Visitor, Singleton, composite, Interpreter
Reliability	Strategy, Chain Of responsibility, Interpreter, composite, Command, Decorator, Façade, State

6. Conclusion

The crowdsourced tactic specified in this paper was directed to discover optimum practices of patterns in software Domain and their effect on quality. The assumptions were validated or invalidated based on responses collected from the crowd of software engineers and ML experiments. This research determines that 8 out of 23 patterns are highly recommended in industries for each (Web, Mobile or Hybrid) type of applications. It is established from the survey that maintainability is the most affected quality attribute among all the attributes as its answers were 100 percent from the survey. Additionally, the preference of Patterns for preferred quality attributes (like Maintainability, reliability, portability, and efficiency) was also identified from the ML recommender system. This finding of exact patterns guides the designers to concentrate on these patterns more than others while focusing on software product's desired nature and preferred quality to be gained.

References

[1]. S. Hussain, J. Keung, A. A. Khan, and K. E. Bennin, "A Methodology to Automate the Selection of Design Patterns," 2016 IEEE 40th Annual Computer Software and Applications Conference

(COMPSAC), 2016.

[2]. B. B. Mayvan, A. Rasoolzadegan, and Z. G. Yazdi, "The state of the art on design patterns: A systematic mapping of the literature," Journal of Systems and Software, vol. 125, pp. 93–118, 2017

[3]. F. Khomh and Y.-G. Gueheneuc, "Design patterns impact on software quality: Where are the theories?," 2018 IEEE 25th International Conference on Software Analysis, Evolution and Reengineering (SANER), 2018

[4]. M. N. Riaz, "Impact of software design patterns on the quality of software: A comparative study," 2018 International Conference on Computing, Mathematics and Engineering Technologies (iCoMET), 2018.

[5]. A. Sarı, A. Tosun, and G. I. Alptekin, "A systematic literature review on crowdsourcing in software engineering," Journal of Systems and Software, vol. 153, pp. 200–219, 2019..

[6]. B. Lin, "Crowdsourced software development and maintenance," Proceedings of the 40th International Conference on Software Engineering Companion Proceeedings - ICSE 18, 2018.

[7]. B. Walter and T. Alkhaeir, "The relationship between design patterns and code smells: An exploratory study," Information and Software Technology, vol. 74, pp. 127–142, 2016.

[8]. F. Khomh and Y.-G. Gueheneuce, "Do Design Patterns Impact Software Quality Positively?," 2008 12th European Conference on Software Maintenance and Reengineering, 2008

[9]. Gautam, Shikha, and Brijendra Singh. "Assessing the Theoretical Impact of Design Patterns on Software Quality." Software Quality Professional 21.1 (2018).

[10]. Pressman, Roger S. Software engineering: a practitioner's approach. Palgrave macmillan, 2005.

[11]. D. Yu, Z. Zhou, and Y. Wang, "Crowdsourcing Software Task Assignment Method for Collaborative Development," IEEE Access, vol. 7, pp. 35743–35754, 2019.

[12]. A. Sarı, A. Tosun, and G. I. Alptekin, "A systematic literature review on crowdsourcing in software engineering," Journal of Systems and Software, vol. 153, pp. 200–219, 2019

[13]. R. Qiao, S. Yan, and B. Shen, "A Reinforcement Learning Solution to Cold-Start Problem in Software Crowdsourcing Recommendations," 2018 IEEE International Conference on Progress in Informatics and Computing (PIC), 2018.

[14]. A. J. Paramita and M. Z. C. Candra, "CODECOD: Crowdsourcing Platform for Code Smell Detection," 2018 5th International Conference on Data and Software Engineering (ICoDSE), 2018

[15]. X. Zhang, B. Gong, H. Ni, Z. Liang, and J. Su, "Identifying Participants Characteristics Influencing Participant Estimation in Knowledge-Intensive Crowdsourcing," 2019 8th International Conference on Industrial Technology and Management (ICITM), 2019.

[16]. J. Wang, S. Wang, J. Chen, T. Menzies, Q. Cui, M. Xie, and Q. Wang, "Characterizing Crowds to Better Optimize Worker Recommendation in Crowdsourced Testing," IEEE Transactions on Software Engineering, pp. 1–1, 2019.

[17]. E. Gamma, R. Helm, and R. Johnson, Design patterns elements of reusable object oriented software. Reading: Addison Wesley, 1998.

[18]. Claude Y. Laporte and Alain April. "Software Quality Assurance (1st ed.)". Wiley-IEEE Computer Society Pr, 2015.

[19]. S. M. H. Hasheminejad and S. Jalili, "Design patterns selection: An automatic two-phase method,"

Journal of Systems and Software, vol. 85, no. 2, pp. 408–424, 2012.

[20]. S. Hussain, J. Keung, and A. A. Khan, "Software design patterns classification and selection using text categorization approach," Applied Soft Computing, vol. 58, pp. 225–244, 2017

[21]. R. Aliady and S. Alyahya, "Crowdsourced Software Design Platforms: Critical Assessment," Journal of Computer Science, vol. 14, no. 4, pp. 546–561, Jan. 2018.

[22]. Portugal, P. Alencar, and D. Cowan, "The use of machine learning algorithms in recommender systems: A systematic review," Expert Systems with Applications, vol. 97, pp. 205–227, 2018.

[23]. S. Belouafa, F. Habti, S. Benhar, B. Belafkih, S. Tayane, S. Hamdouch, A. Bennamara, and A. Abourriche, "Statistical tools and approaches to validate analytical methods: methodology and practical examples★," International Journal of Metrology and Quality Engineering, vol. 8, p. 9, 2017

[24]. T. Redkar and T. Guidici, Windows Azure Platform. Berkeley, CA: Apress, 2011.

[25]. S. Unankard, "Prediction of Re-tweeting Activities in Social Networks Based on Event Popularity and User Connectivity," Machine Learning and Data Mining in Pattern Recognition Lecture Notes in Computer Science, pp. 357–368, 2018.

[26]. D. H. Stern, R. Herbrich, and T. Graepel, "Matchbox," Proceedings of the 18th international conference on World wide web - WWW 09, 2009.

[27]. Using SPSS to Perform Statistical Analyses," An Introductory Guide to SPSS® for Windows®, pp. 61–90.

[28]. Krzywinski, Martin, and Naomi Altman. "Points of significance: Significance, P values and t-tests." (2013): 1041.

[29]. N. Pathan, Q. Ali, S. Iftikhar, G. Batool, and I. Memon, "Personality Type Recommendation System using Crowdsourcing," 2019 2nd International Conference on Computing, Mathematics and Engineering Technologies (iCoMET), 2019.

[30]. S. Taj, Q. Arain, I. Memon, and A. Zubedi, "To apply Data Mining for Classification of Crowd sourced Software Requirements," Proceedings of the 2019 8th International Conference on Software and Information Engineering - ICSIE 19, 2019.

[31]. Z. U. Kamangar, U. A. Kamangar, Q. Ali, I. Farah, S. Nizamani, and T. H. Ali, "To enhance Effectiveness of Crowdsource Software Testing by applying Personality Types," Proceedings of the 2019 8th International Conference on Software and Information Engineering - ICSIE 19, 2019.

[32]. S. S. Memon, A. S. Shah, I. H. Memon, Q. A. Arain, G. M. Morio, and W. A. Channa, "To Explore the Project Management towards Academic Discipline and Practical Approaches," ENGINEERING SCIENCE AND TECHNOLOGY INTERNATIONAL RESEARCH JOURNAL, VOL.3, NO.3, Sep. 2019.

[33]. I. Memon, H. Fazal, R. A. Shaikh, Q. A. Arain, and T. K. Khatri, "Big data, Cloud and 5G networks create smart and intelligent world: A survey," University of Sindh Journal of Information and Communication Technology , pp. 185–192, 2019.

[34]. K. N. Soomro, rabeea jaffri, I. Farah, and S. A. undefined, "Predictive analysis for employee churn in software industry using exploratory data analysis," 1st international conference on computational sciences and technologies, Mehran University of Engineering and technology, 2019.