The Effect of Layout on the Comprehension of UML Class Diagrams: A Controlled Experiment

Bonita Sharif and Jonathan I. Maletic
Department of Computer Science
Kent State University
Kent Ohio 44242
bsimoes@cs.kent.edu, jmaletic@cs.kent.edu

Abstract

The results of a controlled experiment assessing the effects of different layout strategies on the comprehension of UML class diagrams of two software systems is presented. Six different categories of software comprehension tasks, with varying degrees of difficulty, are used to assess the layouts. Each task consists of several questions aimed at measuring the comprehensibility of a layout. The study involved 45 participants of varied experience in software design and programming ability. A report on the quantitative analysis of accuracy, speed, confidence level and preference of solving the tasks is given. Results indicate that clustered layouts demonstrate significant improvement in subject accuracy and speed in solving the problems in a majority of tasks.

1. Introduction

The UML class diagram is one of the ways a maintainer visualizes the design of software. Empirical studies [3] have shown UML diagrams to be useful during maintenance tasks. Here, a distinction is made between a class diagram and a class model. A class diagram consists of a set of classes and relationships from the class model with layout positioning applied to them. Class stereotypes of control, boundary and entity [2] are useful in understanding the roles and duties of classes in a system. This research aims at empirically validating layout techniques for class diagrams utilizing class stereotypes. That is, we are interested in how different layouts impact comprehension. From our work we have learned that layout is not just about edge crossings and general aesthetics, it is a much more difficult problem with respect to how we process and comprehend the diagram.

This paper continues our previous work [1, 14] of assessing the usefulness of class stereotypes (control, boundary and entity) in class diagram layout. Along with general aesthetics [5, 7, 9, 11, 12], the layouts are primarily organized by class stereotypes of control, boundary, and entity. Our initial pilot study [1] compared three class diagram layouts (orthogonal, multi-cluster, 3-cluster) and showed that two of the clustered layouts (multi-cluster and three-cluster) were more helpful in class diagram comprehension. Yusuf et al. continued this work by conducting an eye-tracking study [19], to understand how people explore information in stereotyped class diagrams using the stereotyped layouts with different tasks. The use of layout, color, and class stereotypes were all assessed to determine their effectiveness in program comprehension. Recently in [14], we replicated the eye-tracking study via online questionnaires and got comparable results. The results of the replicated study were presented in a short paper that statistically showed that the multi-cluster layout does indeed outperform the orthogonal and three-cluster layouts in two types of tasks: UML syntax related tasks and design related tasks.

In this paper, we report the results of a third experiment in our investigation of the effectiveness of stereotyped class diagram layouts. This study includes a larger number of participants, uses two systems to validate our hypotheses and also includes a larger number of class diagrams. In addition, the main goal of this study is to investigate the effect clustered stereotyped layouts have on six types of task categories ranging from easy to challenging. In other words, a finer grained analysis is conducted. Our previous studies did not focus on task categories. They mainly dealt with very basic overview tasks such as the identifying the role of a class in a diagram. The two main research questions this paper tries to address can be stated as follows:

- RQ1: Which software comprehension task categories benefit most from stereotyped class diagram layouts?
- RQ2: Do stereotyped layouts in each task category help design experts and design novices in the same way?

The stereotyped layout techniques are described next. The experimental design is discussed in Section 3. Section 4 analyzes the results and reports the findings of the experiment. Section 5 addresses the threats to validity followed by related work and conclusions.

2. Stereotyped Layouts

Three class stereotypes of control, boundary and entity are visually represented via textual annotations and color. This study uses two types of layouts: orthogonal and multi-cluster. The selection is based on our previous work [14] that suggests the multi-cluster layout outperforms the three-cluster layout. We direct the reader to [1] and [19] for examples of the two layouts used in this study. We briefly describe them below.

The orthogonal layout is based on general aesthetic criteria [5, 7, 9, 11, 12] such as minimizing edge crossings, minimizing edge bends, minimizing edge length, maximizing symmetry, and using 90 degree bends. It does not use information about the class stereotype in layout positioning.

The multi-cluster layout uses information about the class stereotype to position classes into multiple clusters in the diagram. Each cluster represents a tightly connected component. This layout depends on the types of relationships that exist between the classes. A cluster could represent a specific feature in the system. For example, even though in a generalization hierarchy children are shown immediately below the parent class, in the multi-cluster layout we might position the child closer to another class it is associated with or dependent on thus highlighting a particular feature in the system.

The orthogonal and multi-cluster layout both display stereotype information via textual annotations and color to control any biases even though the orthogonal layout does not use the stereotype information.

3. Experimental Design

This section presents details on the logistics of the experiment. The overall design, hypotheses, subject systems used, subjects, tasks, data collection and the basic running of the experiment is presented.

3.1. Experimental Goal and Hypotheses

The experiment seeks to *analyze* two class diagram layouts primarily based on class stereotypes *for the purpose of* evaluating their usefulness in six categories of software comprehension tasks *with respect to* effectiveness (accuracy) and efficiency (speed) *from the point of view of* the researcher *in the context of* students at two universities. The detailed null hypotheses are given below. The alternative hypotheses are 1-tailed predicting the multi-cluster layout performs better.

H_{0a}: There is no significant difference in comprehension *accuracy* between orthogonal layouts and multi-cluster layouts for six categories of tasks. $\mu(\text{Accuracy}_{\text{ortho}}) = \mu(\text{Accuracy}_{\text{multi}})$

H_{0s}: There is no significant difference in comprehension *speed* between orthogonal layouts and multi-cluster layouts for six categories of tasks. $\mu(\text{Speed}_{\text{ortho}}) = \mu(\text{Speed}_{\text{multi}})$

H_{0e}: Design experience does not significantly interact with the layout type (orthogonal or multi-cluster) to have an effect on task comprehension or speed.

The overview of the experiment is shown in Table 1. The main factor being analyzed is the layout of class diagrams with two treatments: orthogonal and multi-cluster. While analyzing the results we also looked at secondary factors such as the interaction effects subjects' design experience (design experts vs. design novices) has on accuracy and speed.

Table 1. Experiment overview

Goal	Study the comprehension effect of two types of class diagram layouts in six software task categories.
Main Factor	Class diagram layouts with two treatments: orthogonal layout, multi-cluster layout
Dependent variables	Accuracy, speed, confidence level
Secondary factors	Design experience (experts and novices) Subject systems (qt and wxWidgets)

Table 2. A 2*2 factorial design with two systems and two layouts.

	Group A (N=27)	Group B (N=18)
First Run	qt with multi-cluster layout on Day 1	qt with orthogonal layout on Day 1
Second Run	wxWidgets with orthogonal layout on Day 2	wxWidgets with multi-cluster layout on Day 2
	N_{experts}= 14 N_{novices}= 13	N_{experts}= 9 N_{novices}= 9

The experiment is split into two parts: comprehension and preference. The comprehension part consists of 22 questions in six different task categories (See Section 3.3) for task category description. The preference part consists of rating two class diagram layouts for aesthetics and comprehensibility with respect to two questions from the comprehension part of the experiment. That is, the preference rating is tied to a question from a task category. We believe this gives a better rating versus asking a user to rate a diagram with respect to no task.

The comprehension part of the experiment was conducted as a within-subjects 2*2 factorial design where all subjects in each group were tested on both types of layouts on two different systems. See Table 2 for experiment setup. The layout type and system were switched for Group B. The same question was not asked for the same layout in the same system to avoid learning effects; instead a similar question was asked in a different system. The preference part of the experiment was the same for each group i.e., both groups rated the orthogonal and multi-cluster layout for each system with respect to two questions. The experiment was conducted in two runs, each on different days. The two runs were done on different (not necessarily consecutive) days to reduce sequencing effects.

3.2. Subject Systems

Two comparable GUI toolkits were used to test our hypotheses: Qt[1] and wxWidgets[2]. The number of unique classes used in Qt and wxWidgets are 122 and 109 respectively. Both Qt and wxWidgets have a good set of documentation available online along with a basic hierarchy of their class model. However, neither system includes class diagrams as documentation with the exception of *doxygen* documentation. The *srctools* framework [17] was used to generate the class model. The diagrams in two different layouts were then manually engineered in a UML drawing editor by inspecting the code and online documentation for entity, boundary and control classes that work together towards a specific feature or functional requirement.

3.3. Task Categories

This study consists of six types of tasks: *Reading* (4), *Overview* (9), *Impact Analysis* (3), *Bug Fix* (1), *Feature Addition* (3), and *Refactoring* (2). See Table 3. There were 22 questions spread across these six task categories[3].

Reading task questions dealt with finding paths between classes, reading the class stereotype information from the diagram, or identifying relationships and classes involved. This was the least difficult of all the tasks. No critical thinking was required to perform these tasks.

Overview task questions consisted of 9 questions that involved understanding the system at a high level such as identifying the role of a class/method or identifying high level functionality in classes.

Impact Analysis task questions asked to identify classes that need to be changed if some functional requirement was added or modified. These tasks impacted both classes and methods. Even though impact analysis tasks are well supported by static analysis tools, we include this type of task to determine the effect layout might have in determining classes impacted by a design change at the UML class diagram level.

The *Bug Fix* question involved presenting the subject with a real bug description taken from the bug tracking repository (bug id 85876 for qt and 1168331 for wxWidgets). The objective was to identify classes that needed to be looked at to fix the bug.

Feature Addition task questions involved presenting a new feature addition scenario. This mainly involved sub-classing operations in order to add the new functionality.

Refactoring task questions asked to improve the design of the existing class hierarchy. Subjects were scored based on whether they chose the correct methods/classes for the pull up method/field or collapse hierarchy refactorings. They were not specifically asked for the names of the refactorings.

A total of eight class modules were constructed based on related functionality which resulted in 32 diagrams (8 diagrams * 2 layouts * 2 systems). Aesthetic criteria [5] are adhered to whenever possible. However, sometimes we prioritized on cluster formation in multi-cluster layouts over aesthetics. The multi-cluster layouts in each module (except the Widgets module which contained five clusters) contained four clusters for both qt and wxWidgets.

Each of the task categories were rated by an expert in terms of their difficulty level. The *Reading* and *Refactoring* tasks were considered to be easy. *Overview* tasks were moderately difficult. *Impact analysis* tasks were considered to be more difficult. The *Bug Fix* and *Feature Addition* tasks were classified as challenging. The classification was based on the very nature of the task itself and the questions involved. The *Bug Fix* and *Feature Addition* tasks needed the subject to understand and process a lot more of the diagram to answer the question correctly. Answers to all questions could be found by analyzing the classes, relationships, attributes, methods, and stereotypes in the diagram.

3.4. Data Collection

The study was conducted online at the subject's place of choice. Four online questionnaires were used: background questionnaire, qt questionnaire, wxWidgets questionnaire and, the debriefing questionnaire. The background questionnaire gathered information about the subjects including a self-assessment of their programming and design abilities. The second and third questionnaires formed the first and second run of the experiment for each system. Each question in each task category was scored on a scale from 0 to 1 depending on the number of correct answers. A sum of all the scores represents the accuracy. The speed i.e., time taken to complete each question was also recorded discreetly via a timing mechanism implemented in the software. The subjects were instructed to answer the questions quickly to the best of their ability. Besides the accuracy and speed, we also collected a confidence level (Likert scale 1=very confident to 5=not confident) of the subject's answer for each question. The debriefing questionnaire collected data after each run, about task clarity, realism, understandability, difficulty, and whether stereotypes helped in answering the questions.

3.5. Study Subjects

A total of 45 students (23 design experts, 22 design novices) participated from two universities. Two of these participants were from industry (experts in design and programming). The subjects were randomly assigned to one of two groups. The subjects had varied

[1] http://www.qtsoftware.com, v. 4.3.3, ~ 729 KLOC
[2] http://www.wxwidgets.org, v. 2.8.7, ~ 609 KLOC
[3] See http://www.sdml.info/downloads/taskcat09.pdf for the questions.

programming and design expertise. Both graduate and undergraduate students participated. The subjects were classified into experts or novices based on their grades in the course they were enrolled in. The subjects were informed that the purpose of the study was to understand how people interpret class diagrams (not their UML proficiency). They were also instructed to answer the questions from the viewpoint of a maintainer trying to understand the system. Most of the subjects were not familiar with the design of either system.

Table 3. Modules per task category used in qt and wxWidgets (wx). Each module is represented by a class diagram in two layouts: orthogonal (ortho) and multi-cluster (multi). The number of edge crossings and edge bends is also given for each layout. The number of classes and relationships is the same across both layouts. The difficulty level is shown near each task category.

Modules	System	Number of classes	Number of Relationships	Number of Crossings		Number of Bends		*Software Task Categories*						Preference Questions
								Reading (Easy)	Overview (Moderate)	Impact Analysis (Difficult)	Bug Fix (Challenging)	Feature Addition (Challenging)	Refactoring (Easy)	
				Layout Type		Layout Type								
				multi	ortho	multi	ortho	Questions per module						
Widgets	qt	23	25	0	0	27	25		Q10	Q15	Q17	Q19		Q19(qt)
	wx	23	26	0	0	14	36							
Database	qt	18	20	1	0	30	38	Q1	Q13					
	wx	18	20	1	1	15	40							
Main Window	qt	17	18	0	0	16	20		Q6	Q14				Q6(wx)
	wx	17	17	0	0	11	26							
Graphics	qt	19	20	0	1	23	26	Q2	Q9			Q20		Q9(qt)
	wx	19	20	0	0	14	33							
IO	qt	18	17	0	0	14	26	Q3				Q18	Q21	
	wx	18	19	0	0	15	17							
Events	qt	18	22	0	0	19	16	Q4	Q12					
	wx	18	19	0	0	12	19							
Model View	qt	22	23	0	0	16	23		Q5	Q16				Q16(wx)
Document View	wx	22	27	2	2	28	33		Q8					
Layout	qt	24	27	1	0	20	28		Q7				Q22	
	wx	24	24	0	0	13	22		Q11					

3.6. Running the Experiment

A couple of days before the experiment, the subjects were asked to go through a class diagram tutorial. A short description of class stereotypes and their graphically representation was given. They were also informed of the colors used to differentiate between different class stereotypes. The tutorial was optional; however, all subjects with an exception of a few participated in the tutorial and stated that it helped them refresh their knowledge of class diagrams. After the tutorial, the study was completed in two runs.

During each run, subjects were first presented with a set of instructions and a one screen description (on a 17” monitor with the resolution at 1280*1024) of the system (qt or wxWidgets). We recommended that the subjects take the study on a 17” monitor with the appropriate resolution to view most of the diagram on one screen with minimal scrolling. Most of the questions were multiple-choice. After each question, they were prompted to enter a confidence rating for the question they just answered. There was no time limit set. The only requirement was they finish each run in one sitting. The following information was presented for each question: the question statement, answer choices and a class diagram for qt (or wxWidgets) in one of two possible layouts. The subjects were asked to choose the answer for the question with respect to the class diagram. After completing all the tasks, subjects were presented with a debriefing questionnaire.

4. Experimental Results and Analyses

This section presents the results of the comprehension and preference parts of the experiment. Descriptive statistics are given in Table 4 along with box plots in Figure 1 for accuracy and speed. We report on the effect of layout on accuracy and speed in all task categories as well as in each of the six task categories. Effect size is also reported using Cohen’s d for accuracy and speed to facilitate easy comparison to other studies. Since this is a within-subjects study, we use the paired Wilcoxon non-parametric test to determine significance of the results.

4.1. Effect of Layout on Accuracy

The alternative hypothesis to H_{0a} is that the multi-cluster layout performs better (has a higher accuracy) than the orthogonal layout. We can reject the null hypothesis H_{0a} considering all tasks in both Groups A and B (p-value=0.000036 and 0.00005 respectively). The effect size is medium (0.51 and 0.55) for both groups. However, after performing a finer grained significance test on each task category, we cannot reject H_{0a} for the feature addition tasks (Group A p-value=0.07, Group B p-value=0.09) and the refactoring tasks (Group

A p-value=0.233, Group B p-value=0.874) in either group. The effect sizes for these two task categories (feature addition and refactoring) are small (Cohen's d ~ 0.2) compared to the rest of the tasks that were found to improve significantly (medium effect) with the multi-cluster layout. The only thing we can conclude from this, is that there was no significant difference (with a small effect) in layout for these two task categories.

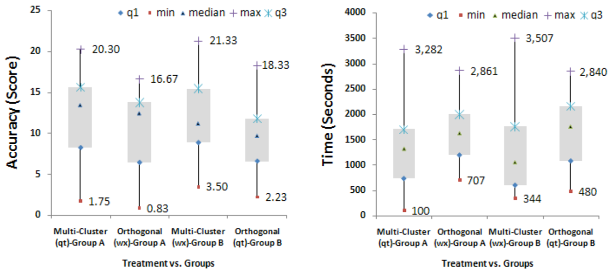

Figure 1. Accuracy and time box plots across groups

We believe this outcome was caused by the specifics of the questions involved in these task categories. The questions did not take advantage of information in clusters causing both the layouts to have no significant difference. Since refactoring tasks focused on a small very specific portion of the hierarchy (such as a pull up field/ collapse hierarchy refactoring), layout did not have much effect. One of the three feature addition tasks (Q19) was much more difficult than the other two. This might have also affected the result. A question by question analysis between layouts is left as a future exercise.

Over all 45 subjects, the Pearson correlation indicates a significant positive correlation between accuracy and programming and design skills reported by subjects in the background questionnaire (p-value<0.0024 r_s=0.41). We report this correlation to validate our subject categorization of experts and novices.

4.2. Effect of Layout on Speed

The alternative hypothesis to H_{0s} is that the multi-cluster layout takes less time to complete a task compared to the orthogonal layout. This can be considered to be the effort required to complete a task. We can reject the null hypothesis H_{0s} for all tasks in both Groups A and B (p-value=0.001 and 0.006 respectively). See Figure 1 for distributions. The effect size is low (0.1) in Group A and close to medium (0.46) in Group B.

A finer grained analysis per task category shows no significance in speed between the two layouts for the refactoring tasks in Group A (p-value=0.06). The same is the case for Group B (p-value=0.187). This effect could be attributed to the same earlier fact that refactoring tasks were considered to be easy tasks focusing on a small portion of a hierarchy. It is also important to note that the effect size is very small (~0.1) in both groups indicating a very small effect that both the layouts are not different. In addition to the refactoring tasks, we cannot reject H_{0s} for the reading and bug fix tasks for Group B, albeit with a very small effect size (0.03 for reading and 0.12 for the bug fix task).

We also notice a high effect in accuracy (1.46) for the bug fix task in Group B, but a low effect in terms of time. From this we observe that even though the time taken was not significantly different, the accuracy for the multi-cluster layout was significantly improved. The same observation is detected in the reading task category in Group B. In Group A, the bug fix task had a large effect (1.04) in speed: both accuracy and speed were significantly improved for the multi-cluster layout.

The only task category that we could not reject H_{0a} and H_{0s} was for the refactoring tasks for both groups (with a small effect). The refactoring tasks were considered to be easy and did not take advantage of layout as much as the other tasks, since they focused on a very specific area of the diagram and did not involve searching for related classes or relationships. For the difficult and challenging task categories, speed of task completion was significantly lower for the multi-cluster layouts with a medium effect on average. A medium effect is considered to be practically significant.

4.3. Secondary Factor Interactions

The third null hypothesis H_{0e} seeks to determine if the secondary factors such as design experience or the systems used, interacts with the layout while performing the tasks. We perform a secondary factor analysis on the whole data set (both groups N=45). Here, we satisfy the conditions of the ANOVA significance test (normality was measured using Shapiro-Wilk). We conduct a 2-way ANOVA between layout and design experience and between layout and system.

Results indicate that design experience does not significantly interact with the layout type (orthogonal or multi-cluster) to have an effect on task comprehension accuracy or speed. We cannot reject H_{0e}. The same is true for the interaction between layout and system used. This validates the fact that the systems chosen were comparable in nature. However, ANOVA did report a direct significant effect due to the experience factor alone for both accuracy (p-value=0.02) and speed (p-value<0.0001), indicating a significant difference in the way experts and novices comprehend diagrams. During this analysis we noticed that novices tend to benefit more from the multi-cluster layout and took much less time to complete the tasks. See interaction plot in Figure 2. A 3-way ANOVA between layout, design experience and system used found no additional interactions.

To summarize, we revisit the research questions we posed in the Introduction. With respect to RQ1, we find the difficult and challenging task categories to benefit most from the stereotyped multi-cluster layouts. With respect to RQ2, experts gave more correct answers in the multi-cluster layout than novices and there was a slightly bigger difference in average accuracy between the two layouts for experts compared to novices. However, in terms of time, novices had a bigger difference between the two layouts as shown in Figure 2. This states that the multi-cluster layout helped novices answer the question much quicker compared to experts. We also find that novices spent less time on average compared to experts. One possible explanation for this could be due to the fact that the experts approached the study in a more professional manner and took the study more seriously.

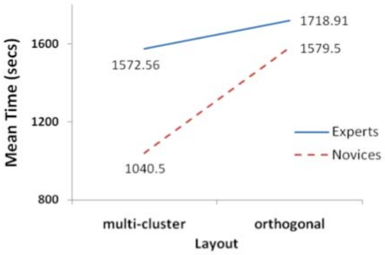

Figure 2. Interaction between layout and design experience with respect to time

Table 4. Descriptive statistics (Accuracy and Speed) for each task including p-values for the 1-tailed Wilcoxon test (alpha=0.05). Cohen's d denotes the effect size: 0.2(small), 0.5 (medium), >=0.8 (large). Alternative hypotheses for the 1-tailed tests indicate the following directionality: μ(Accuracy$_{ortho}$)<μ(Accuracy$_{multi}$) and μ(Speed$_{ortho}$) >μ(Speed$_{multi}$). * indicates significance.

Layout	System	Tasks	Mean Accuracy (Speed)	Standard Dev. Accuracy (Speed)	Cohen's d for Accuracy (Speed)	p(Wilcoxon) 1-tailed Accuracy (Speed)
Group A (N = 27)						
Orthogonal Layout	wx-Widgets	**All**	10.354 (1670.74)	4.507 (534.945)	0.51 (0.1)	0.000036 * (0.001 *)
		Reading	3.104 (174.148)	1.110 (68.201)	0.44 (0.5)	0.035 * (0.048 *)
		Overview	3.524 (614.778)	1.970 (218.547)	0.55 (0.35)	0.00091 * (0.019 *)
		Impact Analysis	1.25 (306.037)	0.943 (165.435)	0.28 (0.3)	0.0295 * (0.025 *)
		Bug Fix	0.333 (134.852)	0.339 (55.851)	0.427 (1.04)	0.01 * (0.0003 *)
		Feature Addition	1.296 (272.556)	0.948 (104.677)	0.252 (0.4)	0.07 (0.007 *)
		Refactoring	0.845 (168.370)	0.508 (128.093)	0.12 (0.1)	0.233 (0.06)
Multi-cluster Layout	Qt	**All**	12.759 (1344.667)	4.818 (754.180)		
		Reading	3.518 (139.741)	0.726 (59.791)		
		Overview	4.827 (526.333)	2.651 (274.742)		
		Impact Analysis	1.487 (244.296)	0.674 (173.291)		
		Bug Fix	0.481 (67.556)	0.353 (71.606)		
		Feature Addition	1.530 (212.556)	0.902 (157.966)		
		Refactoring	0.913 (154.185)	0.553 (119.803)		
Group B (N = 18)						
Orthogonal Layout	Qt	**All**	9.619 (1620.778)	4.070 (676.805)	0.55 (0.46)	0.00005 * (0.006 *)
		Reading	3.111 (132.500)	1.118 (71.837)	0.37 (0.03)	0.0058 * (0.4)
		Overview	3.332 (675.278)	2.159 (313.818)	0.46 (0.35)	0.0053 * (0.03 *)
		Impact Analysis	1.175 (262.000)	0.549 (184.309)	0.52 (0.50)	0.037 * (0.007 *)
		Bug Fix	0.027 (76.889)	0.117 (53.619)	1.46 (0.12)	0.00048 * (0.16)
		Feature Addition	0.888 (267.833)	0.626 (135.577)	0.30 (0.69)	0.09 (0.00001*)
		Refactoring	1.083 (206.278)	0.580 (145.070)	0.26 (0.1)	0.874 (0.187)
Multi-cluster Layout	wx-Widgets	**All**	12.05 (1264.111)	4.722 (840.271)		
		Reading	3.5 (129.833)	0.923 (84.025)		
		Overview	4.407 (542.778)	2.436 (418.339)		
		Impact Analysis	1.569 (170.222)	0.910 (141.417)		
		Bug Fix	0.527 (68.778)	0.468 (76.069)		
		Feature Addition	1.111 (171.222)	0.824 (141.976)		
		Refactoring	0.935 (181.278)	0.533 (156.820)		

4.4. Confidence Level

The confidence level ratings for each question were used to determine if a correlation exists between confidence reported by subjects and accuracy of tasks. The Spearman rank correlation between the confidence level for all questions compared to the accuracy of all the questions showed a positive correlation. (Group A wx: r_s=0.5,p-value=0.0036, Group A qt: r_s=0.55,p-value=0.0014, Group B wx: r_s=0.78,p-value<0.0001, Group B qt: r_s = 0.67,p-value =0.0011). This states that subjects were able to self-assess their responses to the questions in a majority of the questions. In general, a majority of the task categories had consistently higher confidence ratings for the multi-cluster layout compared to orthogonal layout. In cases where there was no difference in layout, the confidence level was also similar in both layouts.

4.5. Debriefing Questionnaire

The debriefing questionnaire asked the subjects to rate the difficulty level of qt and wxWidgets. The Pearson's correlation (N=45) between the difficulty level of each system shows a positive correlation (r_s=0.59, p-value<0.0001). This shows that the subjects found the difficulty level to be around the same. A Mann-Whitney 1-tailed test on all the responses (N=45) in the debriefing questionnaire after each run in each group found no significant differences between the two runs in terms of difficulty, clarity, time needed and task realism. In both runs, stereotypes were found helpful in answering the questions based on a Likert scale rating.

4.6. Preference Ratings

After completing the comprehension part of the experiment, subjects were asked to rate the multi-cluster and orthogonal layouts for two questions (See Table 3) based on two criteria: comprehension and aesthetics. We analyze these preferences using the unpaired Mann-Whitney test (since the ratings are on an ordinal scale) for each question (N=45). See Table 5 for the results.

The aesthetics were comparable i.e., no significant difference was found between the two layouts in the qt system. In terms of comprehensibility of the layouts, the multi-cluster layout was found better for the Overview task-Q9. The feature addition task's comprehension preference ratings did not show any significance. This question (Q19) in particular was much harder and this resulted in a low rating for both layouts. Subjects were not sure if they could answer this correctly and as a result tended to rate both layouts equally badly. This matches our analysis for this task category in Section 4.1. In wxWidgets, the multi-cluster layout was significantly better in both comprehension and aesthetic ratings. This was probably due to the fact that the orthogonal layout had a larger number of bends compared to the multi-cluster layout (See Table 3). In general, the preference ratings tend to mimic the comprehension results presented above.

Table 5. Mann-Whitney results (1-tailed) of preference ratings for 2 questions in each system

Question	Module	Task Category	p(Mann-W) Comp multi > ortho	p(Mann-W) Aesthetic multi > ortho
Q9 - qt	Graphics	Overview	0.01 *	0.20
Q19 - qt	Widgets	Feat. Add.	0.133	0.98
Q6 - wx	Main Window	Overview	0.0004 *	0.00001 *
Q16 - wx	DocumentView	Imp. Anal.	0.005 *	0.014 *

Many subjects commented that they considered the closeness of related classes and relationships to make their choice. Other comments included the ease of tracing relationships as important. Short lines were preferred over long ones. They also preferred a less 'cluttered' look. One subject pointed out that their rating was based on the ability to break the diagram into smaller parts easily. This is precisely what the multi-cluster layout does.

5. Threats to Validity

Since the diagrams were manually engineered, there is the possibility that the type of task might favor a certain layout such that relevant classes for the task are closer in the diagram. The questions in this study were designed in a way that minimizes such situations. The answers to certain tasks were sometimes found not in just one cluster but a combination of clusters. Also, care was taken to ensure aesthetic criteria in all diagrams. The multi-cluster layouts were not drawn to match the tasks, rather they represent certain features in the system.

Being a within-subjects study, learning effects were avoided by using two systems and conducting the study on different days. This experiment was part of a subject's grade in a course. They received credit for participating in the whole experiment not on actual performance. The students we used as subjects had varied experience in programming and design. The experts were comparable to mid-level to senior developers. The subject systems used were real applications. The questions in each task were derived from the repository information available for both systems. Overall, the subjects agreed that the questions were clear, realistic and the information in the class diagrams was understandable.

All conditions for the statistical tests were tested to ensure conclusion validity. We use the non-parametric Wilcoxon test to determine significance due to a small sample size and to perform paired analyses. ANOVA and the unpaired Mann-Whitney test (for preference ratings) were used where appropriate.

6. Related Work

The related work broadly falls into two categories: the proposal of new layout techniques for class diagrams and the empirical validation of class diagrams for comprehension. Eiglsperger et al. [6, 7] present a topology-shape-metrics automatic layout method for class diagrams based on graph aesthetic criteria. Eichelberger et al. [4, 5] investigated the effect of object oriented design, cognitive psychology and human computer interactions on UML aesthetics criteria for class diagrams. They suggest incorporating annotated complexity stereotypes, spatial distribution, scaling based on complexity and coloring into the set of aesthetic criteria for layout of class diagrams. Gutwenger et al. [9] also propose an algorithm for layout of UML class diagrams that balances certain criteria such as crossings and bends. Sun et al. [16] use laws of perceptual theories to propose graph layout criteria for class diagrams. von Gudenberg et al. [18] propose an evolutionary algorithm for class diagrams which can be quite slow.

In the empirical area, Purchase et al. [11, 12] identified that the most important aesthetic preferences for class diagrams were minimizing crossings, minimizing bends, horizontal labels, joined inheritance arcs and more orthogonal layout. Kuzniarz et al. [10][15] investigated the role and effect of telecommunication domain stereotypes in the comprehension of class and collaboration diagrams. Ricca et al. [13] conducted a set of three experiments to determine the usefulness of Conallen's stereotyped class diagrams vs. UML class diagrams. Conallen's stereotypes did not help graduate students but did significantly help undergraduates with little experience in design. The main difference between this study and the above studies is that they do not focus on layout. We see our study as complementary to these studies. An eye tracking study by Guéhéneuc et al. [8] studies how software engineers obtain design information from class diagrams during program comprehension.

7. Conclusions and Future Work

The paper empirically validates two layouts in six categories of software tasks using two open source systems. To our knowledge, this is the first attempt at conducting such an analysis on class diagram layouts. The multi-cluster layout achieves a higher level of accuracy and takes less time than the orthogonal layout for a majority of the task categories including difficult and challenging tasks. In addition, design novices complete the tasks much faster than experts for the multi-cluster layout compared to the orthogonal layout. A preference rating also detects that subjects preferred the multi-cluster layout over the orthogonal one when asked to rate the two diagrams with respect to a task. In the future, we plan on conducting further empirical studies with a larger sample in different software domains and different languages.

8. References

[1] Andriyevska, O., Dragan, N., Simoes, B., and Maletic, J. I., "Evaluating UML Class Diagram Layout based on Architectural Importance", in Proceedings of 3rd IEEE International Workshop on Visualizing Software for Understanding and Analysis (VISSOFT), Budapest, Hungary, September 25th 2005, pp. 14-19.

[2] Booch, G., Jacobson, I., and Rumbaugh, J., The Unified Software Development Process, Addison-Wesley, 1999.

[3] Dzidek, W. J., Arisholm, E., and Briand, L. C., "A Realistic Empirical Evaluation of the Costs and Benefits of UML in Software Maintenance", IEEE Transactions on Software Engineering, vol. 34, no. 3, 2008, pp. 407-432.

[4] Eichelberger, H., "Aesthetics of Class Diagrams", in Proceedings of 1st International Workshop on Visualizing Software for Understanding and Analysis (VISSOFT), Washington, DC, USA, 2002, pp. 23 - 31.

[5] Eichelberger, H., "Nice Class Diagrams Admit Good Design?" in Proceedings of ACM Symposium on Software Visualization (SoftVis), 2003, pp. 159-167.

[6] Eiglsperger, M., Gutwenger, C., Kaufmann, M., Kupke, J., Jünger, M., Leipert, S., Klein, K., Mutzel, P., and Siebenhaller, M., "Automatic Layout of UML Class Diagrams in Orthogonal Style", in Proc. of Information Visualization, 2004, pp. 189-208.

[7] Eiglsperger, M., Kaufmann, M., and Siebenhaller, M., "A Topology-Shape-Metrics Approach for the Automatic Layout of UML Class Diagram", in Proc. of SoftVis, San Diego, CA, USA, 2003, pp. 189-198.

[8] Guéhéneuc, Y.-G., "TAUPE: towards understanding program comprehension", in Proc. of conference of the Center for Advanced Studies on Collaborative research (CASCON), Canada, 2006.

[9] Gutwenger, C., Jünger, M., Klein, K., Kupke, J., Leipert, S., and Mutzel, P., "A New Approach for Visualizing UML Class Diagrams", in Proc. of ACM Symposium on Software Visualization (SoftVis), 2003, pp. 179-188.

[10] Kuzniarz, L., Staron, M., and Wohlin, C., "An Empirical Study on Using Stereotypes to Improve Understanding of UML Models", in Proc. of 12th Intl Workshop on Program Comprehension (IWPC) 2004, pp. 14-23.

[11] Purchase, C. H., Allder, J.-A., and Carrington, D., "Graph Layout Aesthetics in UML Diagrams: User Preferences", Journal of Graph Algorithms and Applications, vol. 6, no. 3, 2002, pp. 255-279.

[12] Purchase, C. H., McGill, M., Colpoys, L., and Carrington, D., "Graph Drawing Aesthetics and the Comprehension of UML Class Diagrams: An Empirical Study", in Proceedings of Australian Symposium on Information Visualisation, Sydney, Australia, December 2001, pp. 129-137.

[13] Ricca, F., Di Penta, M., Torchiano, M., Tonella, P., and Ceccato, M., "The Role of Experience and Ability in Comprehension Tasks supported by UML Stereotypes", in Proceedings of 29th Intl. Conf. on Software Engineering (ICSE), Minneapolis, May 19-27 2007, pp. 375-384.

[14] Sharif, B. and Maletic, J. I., "An Empirical Study on the Comprehension of Stereotyped UML Class Diagram Layouts", in Proceedings of 17th IEEE Intl. Conf. on Program Comprehension (ICPC), Vancouver, BC, Canada, May 17-19 2009, pp. 268-272.

[15] Staron, M., Kuzniarz, L., and Wohlin, C., "Empirical assessment of using stereotypes to improve comprehension of UML models: a set of experiments", Journal of Systems and Software, vol. 79, 2006, pp. 727-742.

[16] Sun, D. and Wong, K., "On Evaluating the Layout of UML Class Diagrams for Program Comprehension", in Proceedings of 13th IEEE Intl. Workshop on Program Comprehension, St. Louis, Missouri, USA, 2005, pp. 317-328.

[17] Sutton, A. and Maletic, J. I., "Recovering UML Class Models from C++: A Detailed Explanation", Information and Software Technology, vol. 49, no. 3, Jan 2007, pp. 212-229.

[18] von Gudenberg, J. W., Niederle, A., Ebner, M., and Eichelberger, H., "Evolutionary Layout of UML Class Diagrams", in Proc.of SoftViz, Brighton, UK,2006, pp.163-164.

[19] Yusuf, S., Kagdi, H., and Maletic, J. I., "Assessing the Comprehension of UML Class Diagrams via Eye Tracking", in Proc. of 15th IEEE Intl. Conf. on Program Comprehension (ICPC), Banff Canada, June 26-29 2007, pp. 113-122.