Effectively Teaching Coding Standards in Programming

Dr Xiaosong Li
Computing and Information Technology
Unitec New Zealand
Carrington Rd, Auckland, New Zealand
(649) 8154321 xt 6019
xli@unitec.ac.nz

Christine Prasad
Computing and Information Technology
Unitec New Zealand
Carrington Rd, Auckland, New Zealand
(649) 8154321 xt 6015
cprasad@unitec.ac.nz

ABSTRACT
In this paper, we report on a study that was carried out to investigate students' opinions on learning and accepting coding standards in programming courses. We used a questionnaire survey to gather data. We also used the information observed from our teaching practices. An analysis of the data indicated that most students' believe coding standards are important in programming courses but tend not to comply with them, thus implying possible flaws in the teaching strategies used. We also present current strategies we use for teaching coding standards, and evaluate them for effectiveness. In doing so, we propose strategies that are likely to be effective in teaching coding standards as they would be used in industry, and present suggestions for further studies that can be carried out to implement these strategies.

Categories and Subject Descriptors
K.3 [**Computers & Education**]: Computer & Information Science Education – *Computer Science Education.*

General Terms
Documentation, Human Factors, Standardization.

Keywords
Coding standards, teaching programming.

1. INTRODUCTION
Coding standards are a set of industry-recognized best practices that provide a variety of guidelines for developing software code. There is evidence to suggest that compliance to coding standards in software development can enhance team communication, reduce program errors and improve code quality [3, 6, 8 p46, 11, 20]. Complying with given coding standards is thus, a vital professional skill required by the software industry; one that ought to be actively developed within IT education. Programming and software development courses are undoubtedly most suitable for developing this skill.

SIGITE'05, October 20–22, 2005, Newark, New Jersey, USA.

Available literature suggests that although many software companies have standards of coding, individual developers tend not to adhere to them, due to a variety of misperceptions about coding standards [6, 11]. As a result, coding standards usually need to be implemented through a formal process in industry [11, 18]. Our three years observation, as programming instructors, is that students are reluctant to apply given coding standards. This observation could either be a result of defective teaching strategies used by instructors or simply the students' inability or lack of interest to learn this skill.

We initially set out to answer the following questions:

How should we teach coding standards?

Is there any common way to implement coding standards in all the programming courses?

How should we assess the learning of coding standards?

We felt that before we could identify how coding standards should be taught, we need to evaluate how we are currently teaching it, and what students' perceptions on coding standards and the way in which it is currently taught are.

The study reported in this paper is the first of a series of studies we are carrying out to look at the teaching of coding standards. In this study, we investigate student perceptions on what coding standards are, and how they are being taught, and get students opinions on how it should be taught.

In the next section we describe the instrument we used to gather this preliminary data. We then present the results and an analysis of the gathered data, discuss current teaching strategies and present suggestions of strategies that we believe can be used to effectively teach and assess coding standards. We conclude this paper by provided a preview of further studies that will be carried out on this topic.

2. METHOD
This is an exploratory study, which means we are trying to generate hypotheses rather than test hypotheses. We used a questionnaire to get an understanding of students' needs and possible flaws in the teaching strategies we are currently using in teaching coding standards. The term "questionnaire" is used loosely here, as the questionnaire was not intended to provide quantitative data for the purpose of carrying out rigorous statistical analysis, but to provide qualitative data to provide the background investigative information. In the earliest stages of research, "what" to study is sometimes not clear. In this case,

qualitative inquiry can be used to uncover problems and generate hypotheses [15]. A written survey can be used to collect opinions [15]. In this study, a mixture of close-ended and open-ended questions is mainly used to collect students' opinions. Each of the questions consisted of a number of choices and an "other" category. The choices were used to avoid ambiguity and the "other" category was used to allow the respondents provide different ideas and respond in detail. Many qualitative studies use more than one type of data and collection technique to enrich and add perspective to the pool of information on their subject of inquiry (called triangulation) [15]. In this study, we also considered the information obtained from our observation and our conversations with the students.

The questionnaire consisted of 21 questions divided into three categories: background information, opinion on programming in general, and opinion on coding standards and how to teach them. As part of the background information, we also asked for students to provide information on their native language background. This data, we felt, was important because (a) we have students of various language backgrounds and, (b) a student's native language could have a possible effect on their level of understanding of coding standards.

We distributed the questionnaire to students in three different programming courses. All of these courses are elective courses self-selected by students. We received 57 responses as shown in table 1, out of these responses 16 were native English speakers and 41 were non-native English speakers.

Table 1. Distribution of responses according to course.

Course	Level	Responses
Advanced Multimedia	3rd yr	9
Web Application Development	3rd yr	22
Programming Principles	1st yr	26
Total		57

3. RESULTS AND ANALYSIS

3.1 Background Information

We sought background information from students on the previous programming courses they had done, and whether or not they currently or previously had been working in a programming related job. We felt this information was necessary to see if there was a correlation between programming exposure to an awareness of coding standards.

Table 2 shows the number of programming courses done by the 3rd year level students, including the current course they were enrolled in. It is not possible for students to enroll in 3rd year level programming courses without having passed the necessary programming prerequisite courses. The two students who claimed that the current course was the only one they had done were probably students who had transferred from other institutions.

Table 2. Previous and current programming courses done by students

Number of programming courses (inclusive of current)	1	2	3	4	5	6
Percentage of 3rd year students (n=31)	6	29	32	23	6	3

Out of the 1st year level students, only one student had done another programming course.

Out of all students, only one student claimed to be currently working in a programming related job while two students claimed to have previously worked in a programming related job. Also, 24 out of the 31 (77%) 3rd year level students and 18 out the 26 (69%) 1st year level student said they wished to become a software developer, presumably after graduating. All 3rd year level students who did not wish to become a software developer were enrolled in the Advanced Multimedia course.

3.2 Opinion of Computer Programming

We felt that it was important to ascertain students' opinion on computer programming as this could be a factor leading to their awareness and understanding of coding standards. Table 3 gives a breakdown of students' opinion of computer programming.

Table 3. Opinion of students on programming.

Opinion	Percentage of students at 3rd year level (n = 31)	Percentage of students at 1st year level (n = 26)
Enjoyable	58	50
Difficult	16	61
Boring	9	0
Ordinary	3	3
No comments	3	0
Challenging	58	45
Frustrating	3	0

Many 3rd year level students found their current programming course to be enjoyable or challenging, while many 1st year level students found it to be difficult.

Students were also asked which topics in particular they faced difficulties with, if any, and table 4 summarizes these responses. For each topic, a count of the number of native and non-native English speakers was taken.

The most difficult topic amongst 3rd year level students was surprisingly "writing programs", while amongst the 1st year level students was "understanding coding standards". From the 3rd year level students, more native English speakers felt that writing programs was difficult, while all the other topics were regarded as difficult by non-native English speakers. Of particular interest was the fact that none of the native English speakers found it difficult to follow coding standards while 25% of non-native English speakers found the topic difficult. This is consistent with our observation from our teaching practices.

From the 1st year level students, a higher number of native English speakers found the topics of "program design" and "understanding program concepts" difficult. Similar to the 3rd year level students, more of these students found the topics of "understanding coding standards" and "following coding standards" difficult. One of the reasons for this could be that most programming languages and the documentation on coding standards are in English. Further research is required to investigate this.

Table 4. Distribution of topics of difficulty

Topic of difficulty	Number of students at 3rd year level (n = 31)			Number of students at 1st year level (n = 26)		
		% of responses by native English speakers (n = 7)	% of responses by non-native English speakers (n = 24)		% of responses by native English speakers (n = 9)	% of responses by non-native English speakers (n = 17)
Program design	**9 (29%)**	29	29	**8 (31%)**	56	18
Writing programs	**12 (39%)**	57	33	**8 (31%)**	22	35
Understanding program concepts	**7 (23%)**	0	29	**6 (23%)**	33	18
Understanding program specification	**3 (10%)**	0	13	**2 (8%)**	0	12
Understanding coding standards	**6 (19%)**	14	21	**11 (42%)**	22	53
Following coding standards	**6 (19%)**	0	25	**5 (19%)**	11	24
Other: Apply OOP	**1 (3%)**	0	4	**0 (0%)**	0	0
Other: Terminology	**1 (3%)**	14	0	**0 (0%)**	0	0
Other: Debugging	**0 (0%)**	0	0	**1 (4%)**	11	0

3.3 Perception of Coding Standards

This section enquired about students' awareness and perception on the teaching and learning of coding standards. There was very little difference between the 1st year and 3rd year level students here. 71% of the 3rd year level students and 73% of 1st year level students had heard of the terms "coding standards" before completing the questionnaire.

Of the 3rd year level students, 97% thought that coding standards was an important topic while 3% (1 student) did not care. Of the 1st year level students, 85% thought that coding standards was an important topic while 12% did not care.

Table 5 shows the distribution of responses when students were asked which documentation standards and techniques they were familiar with.

Table 5. Distribution of response on known coding standards/documentation techniques

Known technique	Percentage of students at 3rd year level (n = 31)	Percentage of students at 1st year level (n = 26)
Naming Convention	52	54
Hungarian notation	45	65
Comments	74	65
Documentation	55	62
Other	0	0
Don't know	3	0

Interestingly, only 65% of 1st year level students were familiar with the Hungarian notation for naming variables. This notation was introduced in this course and fully enforced throughout lectures and assessment.

When asked their opinion on coding standards, the majority of student at both levels felt that it was helpful in reducing errors. However, as shown in table 6, more 1st year level students thought that enforcing a coding standard was extra work than 3rd year level students, while less 1st year students felt that it would be helpful for team communication. The latter result is perhaps a direct effect of the absence of group assessment in this course.

Some of the other responses to this question included:

"Better for support"

"Increase code readability"

"Work well when used with common sense, overly strict"

Table 6. Distribution of opinions on using coding standards

Opinion of coding standards	Percentage of students at 3rd year level (n = 31)	Percentage of students at 1st year level (n = 26)
Extra work	16	23
Makes code look better	58	46
Helpful in reducing errors	71	69
Helpful for team communication	52	38

Table 7. Distribution of opinion on time to be spent on coding standard in programming course

Time to be spent on coding standards	Percentage of students at 3rd year level (n = 31)	Percentage of students at 1st year level (n = 26)
1%	3	8
2%	29	4
5%	23	27
10%	16	38
More than 15%	29	19

Of the 3rd year level students, 87% expected to learn coding standards in a programming course as opposed to 77% of 1st year level students. However, the same proportion of students at both levels thought that coding standards ought to be assessed.

Students were also asked how much time they prefer to spend on coding standards within their course. Table 7 shows the distribution of responses for this question.

Surprisingly, an equal proportion of 3rd year level students felt that not much time (2%) and a lot of time (>15%) should be spent on coding standards. Most of the 1st year level students felt that 5-10% would be sufficient.

Students were also asked how much coding standards they have learnt so far, the distribution of which is shown in Table 8. Both groups of students felt that they had learnt little to average.

Table 8. Distribution of opinion of how much coding standards has been learnt so far.

How much coding standards have you learnt so far?	Percentage of students at 3rd year level (n = 31)	Percentage of students at 1st year level (n = 26)
Nothing	10	4
Little	42	42
Average	39	38
Plenty	6	12
Cant remember	3	0

Table 9. Distribution of preference of learning coding standards

Topic of difficulty	Number of students at 3rd year level (n = 31)			Number of students at 1st year level (n = 26)		
		% of responses by native English speakers (n = 7)	% of responses by non-native English speakers (n = 24)		% of responses by native English speakers (n = 9)	% of responses by non-native English speakers (n = 17)
Documents	**13 (42%)**	43	42	**4 (15%)**	22	12
Example	**20 (65%)**	86	58	**20 (77%)**	100	65
Lectures	**10 (32%)**	29	33	**12 (46%)**	33	53
Practice	**11 (35%)**	43	33	**11 (42%)**	67	29

Finally students were asked for their preference of how they would like to learn coding standards. The distribution of responses is summarized in Table 9.

As evident from Table 9, both groups of students mostly prefer to learn coding standards by example, and least like to learn from documents, although 1st year level students showed a much lower preference to learning from documents and a higher preference to learning from example and practice in comparison to 3rd year level students.

Native English speakers prefer examples and practice much more than non-native English speakers while non-native English speakers prefer lectures more than native English speakers. This is also consistent with our observation from our teaching practices. However, we felt that this is more like a culture issue than a language issue. Further research needs to be carried out to determine this.

4. CURRENT PRACTICES

Many programming languages, for example Java, have their own common set of coding standards. Most programming courses provide coding standards and require students to follow [1, 2, 3, 5, 12, 13, 16, 17, 19, 21]. Usually, programming courses use a small set of common coding standards. The purpose is to introduce students to a representative set of coding standards that are typical to professional programming practices and to help students develop the habit of good coding style, which is necessary to successfully complete large programs. In some case, one programming course has its own coding standards, while in other cases several programming courses use the same set of coding standards.

In reflecting on our teaching practices on coding standards, we have basically implemented a similar strategy as the above. In one of the 3rd year level courses we teach the VB language. In the past, we basically gave students a document which briefly outlined expectations in broad generalities on coding standards and referred the students to Microsoft VB Coding Convention for more specific guidelines. We asked the students to follow these in their assignments and assessed them accordingly. What was found in the assessment is that students rarely made any effort on enforcing the standards. When the students were asked why they didn't follow the coding standards, the answers usually were that they had very short schedule for the assignment and didn't have time to worry about coding standards. In other words, enforcing coding standards was not a high priority for students.

Clark's article [3, 10] indicates that although Microsoft's guidelines are an excellent guide, they are not exactly a quick reference guide and there is no summary of standards. Instead, Clark proposed a small set of the same coding standard. This year, we adopted a modified version of Clark's coding standards, which is small and explicit. As a result, many students attempted the coding standards in their assignments to a much greater extent than in the past, but still not to the extent to develop a habit of following it. This shows that we are heading in the right direction although still deficient.

For the same course, we asked the students to follow the coding standards while doing their practical exercises in the practical

sessions. We also enforce them in our sample code. However, we found that the students still did not pay great attention to them. The coding standards were only assessed in the assignment, but not in the final exam. This might suggest to the students that coding standards are not a compulsory part for their learning.

For the 1st year level course, a rough guideline is given on laying out code, self-documenting code and using Hungarian notation for naming functions and variables. That is, a formal guideline isn't given. These guidelines are enforced in lectures and practical, and assessed in practical tests and assignments. Many students tend to follow it for assessments, where marks are allocated for proper naming, code layout and documentation, but do not enforce it for practical exercises. For example, it is easier to use a variable called x or y, than iNum or bValid. This suggests that students are not forming a habit of good coding style but are only doing it to gain some marks. A negative factor in this course is that the course textbook does not enforce these standards, although this may always be a problem with different textbooks enforcing different standards.

However, as indicated in the survey results (Table 9), students feel that learning coding standards from documents is their least popular preference. This means that the lack of use of coding standards in textbook should not matter as long they are enforced in sample code and lectures.

Another noteworthy observation is that some integrated development environments (IDE's) automatically create variable names. These names violate the name conventions defined in our coding standards. For example, in MS Visual Studio.NET, a save button is created automatically as Button1, but it should be called btnSave to comply with our coding standards. Although it is possible to change the variable names, this take times and somewhat defeats the purpose of using an IDE.

Overall, our observation is that students are reluctant to apply given coding standards. This is consistent with the results we found from the questionnaire, for example, in table 4, many student found the topics of understanding and following coding standards difficult. Also, from table 8, students feel that they have learnt little to average about coding standards. At 1st year level, it is perhaps a lack of insight into program maintenance and the lack of experience of writing large programs that hinders students in following coding standards, but at 3rd year level, students should have developed sufficient experience to learn the advantages of following standards. Table 6 indicates that students, particularly at 3rd year level, are aware of the advantages, yet they don't seem to apply these standards.

5. PROPOSED TEACHING STRATEGIES

It appears that we need a balanced approach for teaching students coding standards at different levels.

This approach should only include a small set of common coding standards as the purpose of it is to introduce students to a representative set of coding standards that are typical in professional programming and to help students develop skills on coding standards. However, the complexity of given coding standards should be different for courses at different levels. So it is not adequate to give the same set of coding standards for one language (for example, VB) at all the levels of a program. Given the variation of responses between 1st year and 3rd year level students, we should give students at lower levels more guidance on coding standards than at higher levels.

There are some reports on how to implement coding standards in industry [3, 11, 12]. Usually this is a process whereby the developer checks and modifies code by following formal style standards, the standards are enforced during code reviews and design reviews, and finally the progress data is collected and analyzed. This may well work in teaching. For example, we can introduce code review process as parts of the assessment for an assignment, in which the students can do peer assessment on their codes. This is also consistent with students' preference of learning by example and practice (table 9). We should still allocate a small amount of time during lectures in each course to do so.

As indicated by Qiu [14], providing detailed individualized feedback to student code remains critical. Code review provides an opportunity for the students to get peer feedback. Other than that, teachers should provide feedback by face to face assessment. We can also do some coding standards practical exercises, in which we can give the students some small programming tasks and then review their code by paying particular attention on their compliance with the coding standards. We will review these with the students and give them immediate feedback. After the review, the students will be provided with a couple of the solutions for the same tasks which comply with the given coding standards.

Another possible way to enforce coding standards is to introduce some automatic tools to facilitate the compliance of the coding standards. Within the last few years, new tools have emerged that are intelligent enough to exercise a new piece of software - to bend it, break it, and subject it to realistic and comprehensive test environments - and are flexible enough to extend the same sort of testing to any number of software products [9].

We should try to provide better environment and opportunities to facilitate the communication between native English speaker students and non-native English speaker students regarding coding standards. So they can learn from others strong points to offset their own weakness. For example, we can encourage students to form mixed culture groups in the code review and the coding standards exercises.

In defining a general process, a coding standards document should be given initially in each course, then appropriate examples should be given in the lectures and sample code and these should be reinforced in the practical exercises.

Also, we feel that in order to the compliance of coding standards the priority it deserves, they should be assessed consistently throughout the course, even in exams.

6. SUMMARY AND FUTURE RESEARCH

To summarize this paper, more effective teaching strategies are needed for coding standards. Different teaching practices should apply to the different levels of the courses. A formal process should be adopted for both introducing coding standards and implementing coding standards. Also, different strategies may be required for native and non-native English speakers but this is open to further research.

It is obvious that the more effort we put on coding standards, the better results we'll get. However, it is difficult to determine that how much effort on coding standards is appropriate for a programming course given that we have a lot of more challenging material to teach. This makes effective teaching strategies more important.

There are some limitations of the research methods used in this study. One of them is that the opinions collected by means of questionnaires are less natural than the opinions collected by means of interview [15]. Another is the prestige bias, which is the tendency for respondents to answer in a way that make them feel better. The teaching strategies proposed need to be tested and evaluated. Further research is needed to provide more concrete steps for the proposed teaching strategies. Research is needed on introducing tools to implement coding standards. Further research is needed to study the latest IDEs to decide what really is needed for modern coding standards, as the IDEs enforce consistency which may make some of the traditional coding standards obsolete. The next step of this research involves using a questionnaire survey to get perceptions on how to teach coding standards from different faculty across different institutions.

7. REFERENCES

[1] Bezroukov, N. *Programming Style*. 2003. http://www.softpanorama.org/SE/programming_style.shtml . Accessed 1 July 2005.

[2] Boston University, *Java Coding Standard for CS 111 & CS 112*, 2005. Accessed March 4, 2005. http://www.cs.bu.edu/fac/snyder/cs112/CourseMaterials/JavaStyleNotes.html

[3] Central Washington University, Java Programming Style Guide CS 110. Accessed July 4, 2005. http://www.cwu.edu/~gellenbe/javastyle/

[4] Clarke, D. *Resources - ASP.Net Coding Standards*, 2005. Accessed March 4, 2005. http://www.visualize.uk.com/resources/asp-net-standards.asp

[5] Forward, B. *ITSE 1418 -- COBOL Coding Standards*, 2001. Accessed March 4, 2005. http://www.texarkanacollege.edu/~bforward/mcobstnd.htm

[6] Fryman, J. *Coding Standards: Good Idea or Subtle Evil?* 1999. Accessed March 4, 2005. http://freshmeat.net/articles/view/139/

[7] Gellenbeck, E. *Java Programming Style Guide CS110.* Accessed 1 July 2005. http://www.cwu.edu/~gellenbe/javastyle/

[8] Howles, T. Fostering the growth of a software quality culture. *The SIGCSE Bulletin, Volume 35 Number 2*, 2003 June.

[9] Kolawa, A. and Coffee, K. (2005) *Why Is Error Prevention Important?*. Accessed March 4, 2005. URL: http://www.stickyminds.com/sitewide.asp?ObjectId=3489&Function=DETAILBROWSE&ObjectType=ART

[10] Microsoft, *.NET Framework General Reference Design Guidelines for Class Library Developers*, 2005. Accessed March 4, 2005. http://msdn.microsoft.com/library/default.asp?url=/library/en-us/cpgenref/html/cpconNETFrameworkDesignGuidelines.asp

[11] Parasoft, *Understanding the Workflow in a Coding Standards Implementation*, 2005, Accessed March 4, 2005. http://66.102.7.104/search?q=cache:TvLOqsjDqxYJ:www.parasoft.com/jsp/products/article_reg.jsp%3FarticleId%3D1158+is+coding+standard+important+for+software+industry&hl=en

[12] Pfeiffer, J. *Coding Standards in 400- and 500-level Classes*. http://www.cs.nmsu.edu/~pfeiffer/classes/general/s05/coding.html. Accessed 1 July 2005.

[13] Purdue University*, CS 177 Java Programming Standards, 2005*. Accessed March 4, 2005. http://web.ics.purdue.edu/~cs180/Spring2005cs177/java_programming_standards.html

[14] Qiu, L. *Intelligent Educational System for Teaching Programming*, Accessed July 4, 2005. http://www.cs.oswego.edu/~lqiu/critiquer/publications/acm2004.pdf

[15] Rich, M. and Ginsburg, K. The Reason and Rhyme of Qualitative Research: Why, When, and How to Use Qualitative Methods in the Study of Adolescent Health. *Journal of Adolescent Health, 25*, 1999, 371-378.

[16] Struble, C. *CS 1044 General Programming Standards*. 2000. http://courses.cs.vt.edu/~cs1044/fall00/cstruble/Standards.html. Accessed 1 July 2005.

[17] Sutter, H. *C++ Coding Standards: Rules and Guidelines for Writing Programs.* Addison-Wesley. 2004

[18] TIOBE Software BV. *TIOBE Coding Standard Methodology, 2003*. Accessed March 4, 2005. http://www.tiobe.com/standards/tekst.htm

[19] Wightman, R. *C Language Coding Standard for CS1003 and CS1013*. 2002. http://www.cs.unb.ca/courses/cs1013/ProgrammingStandard.pdf. Accessed 1 July 2005

[20] Wiki. *Formal Standards. September 18*, 2004. Accessed March 4, 2005. http://c2.com/cgi/wiki?FormalStandards

[21] Wilson, J. N. *CIS 4930 Section 0998X*, Spring, 1996. Accessed March 4, 2005. http://www.cis.ufl.edu/~jnw/OOCourse/

[22] Zaidman, M. Teaching defensive programming in Java. *Journal of Computing Sciences in Colleges, 19, 3* (Jan. 2004), 33 – 43.