Journal of Scientific Research and Development 2 (4): 95-99, 2015
Available online at www.jsrad.org
ISSN 1115-7569

Increasing program's readability using INFC protocol

Rahim Rashidi *, Behnam Zebardast, Kamal Habibi, Diako Saadi

Department of Computer, Boukan Branch, Islamic Azad University, Boukan, Iran

Abstract: The ability to read and write is the criterion of a programmer and both are of general criteria of a good language. If the structure of algorithm's program and program's data are well defined , that program has the readability, but syntactic specifications of a language that makes programming easy are often in conflict with syntactic specifications which increase the readability. In this paper, a stack protocol called INFC has been offered to increase readability and communication between programmers. The use of this stack protocol causes readability of the program has been independent of specification of a language which increases the ability to write. Decrease of logical errors, maintenance, learning proper coding, and high quality and easy to realize the code are the benefits of the INFC stack protocol.

Key words: Readability feature; The ability of writing; Protocol INFC; Programming logic; Development team

1. Introduction

If the structure of the algorithm of program and readable its data are well defined, that program is and it is called self-citation program, it means that this program is understandable with no separate documentation, but somehow the syntactic specification of a language that makes easy the ability to write a program are often in conflict with the syntactic specification which increase the readability (Abran et al., 2001). The ability to write is the result of an ordered and organized structure, while long structures are useful for readability. In this paper, a stack protocol called INFC is presented to increase readability and communication among programmers (Lauese, 2002). Using this stack protocol is caused those syntactic properties of a language that increases the readability to be independent of those syntactic specifications that increases the capability to write. Therefore, the use of ordered and precise structures while coding increases the capability to write the program while using INFC protocol for the program code increases the readability.

In other words, you can simultaneously use the advantage of the program's readability and ability of writing. Verification or inspection of program's logic, code maintainability, easy understanding of the program, code quality and the communication between the programmers are linked with readability of program. From important advantages of this protocol can be pointed to the reduction of logic errors in program and the relationship between programmers. Because each program during its lifetime rarely and hardly maintained by the original author (Kovitz and Practical, 1998), so INFC stack protocol provides a mechanism so that programmers can easily communicate with each other and to minimize the number of program's logic errors.

2. INFC protocol stack

Given that the syntactic specifications of a programming language which simplify writing the programs are in conflict with the syntactic specifications which increase readability, there is no mechanism that can be applied the rules to the structure of programs to increase the readability and the capability to write. INFC stack protocol makes independent the readability of the syntactic specification of the writing ability. This stack protocol consists of four layers (from bottom to up):

- Integrated Development Environment
- Naming Conventions
- Formatting
- Commenting

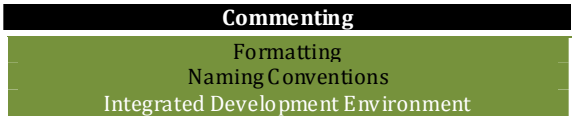

Fig. 1: shows INFC protocol along with its layers

3. The proposed protocol

First layer is software development environment that a programmer writes and edits code, compiles and runs it. The second layer is an application name's contracts and the programmer applied these contracts to the choice of ID's names. The third layer is the template package, and includes rules for denting, alignment, the length of lines and empty spaces. The fourth layer is of explanation or interpretation of the code where the programmer

* Corresponding Author.

describes program lines. This stack protocol will follow the following objectives:

It leads to code compatibility, and code readers can focus on the logic of the code**.**

Based on the rules defined in each layer, the programmer quickly can understand the logic of the code.

The accuracy or inspection of the logic of the program and repairing of the code becomes simpler and faster.

Ability to code maintenance increases.

The following fig.2 shows the available protocols in INFC:

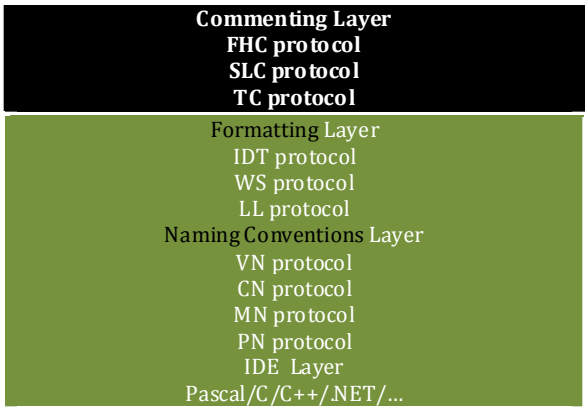

Fig. 2: The available protocols in INFC

4. The sum of INFC protocols:

The layer of integrated development environment. While reading or writing of a program's source of the code in the simple text editor such as Notepad may be difficult, doing this in the IDE due to its vast possibilities of using the system color code is very easy. In this layer, the string variables can be displayed in a program code with the desired color from the presented protocol stack in order to be easily differentiated from the code throughout the source code. Improved staining of code can also be observed in the other backgrounds. According to the proposed protocol, the first factor which separates software development environments and increases the readability is the programming environment or IDE. This word is the abbreviation of integrated development environment. IDE is an integrated development environment in which the entire coding process is applicable within it from programming language to producing an executable file. In addition the integrated development environment can be held various other facilities in the service of programmer including sorting code, code highlighting, debugging and project management of the program that will be greatly facilitated writing the program, and will increase its readability. Although there are other environments for writing the language of program-maker but INFC protocol suggests the integrated development environment for writing code because IDE is one of the best developmental environments for programmers.

5. The layer of naming contracts

A consistent naming and standard protocol suggested by the second layer INFC is an important factor in readability and maintenance of the program. Using Naming Conventions' layer in INFC protocols can choose the IDs in the case that increased readability of the program and the written code can be easy to understand for others (programmer or development team). In this protocol it should be considered the following for selecting the name:

Fully define the first name and last name. Avoid abbreviations. In other words, instead of using the names First Name and Last Name, its shorter copy i.e. f Name and l Name should be used. Avoid too long names (more than 15 characters). For example set The Length Field's ID should be shortening as set Length. Avoid the names that are very similar or differ in only one character. For example, do not use the names of student and students which are very similar.

Using meaningful names in this layer can be written a self-citation code; it means the program is understandable without any separate documentation. For example, the following code snippet shows the names of the variables of sales Tax and income Tax is suitable for tax1 and tax2 because these names are self-citation.

Double tax1; //sales tax rate (example of poor variable name)

Double tax2; //income tax rate (example of poor variable name)

Double sales Tax Rate; //no comments required due to

Double income Tax Rate; //self-documenting variable names

If in the second layer, strong contracts are used for naming of IDs, the work of programmer will be less in the fourth layer.

6. Naming of variables

Use the meaningful names for the variables' names and avoid using of generic names, such as the number or temp whose purpose is unclear.

The style of using capitalization should be the camel Case style, it means the first letter of ID should be the small and the first letter of the next word should be written in capital letters. Use plural names for the array name, it means that in the program's code it should be used test Scores instead of test Score (Java Programming Style Guide).

7. Naming of the constants

Use ALL_UPPER_CASE method for naming the constants. Separate words with the small line. For instance use tax Rate instead of TAX_RATE. Do not

use magic numbers in the program's code. The magic numbers are actual numbers like 27 which in the reader are created the impression that the number 27 is used for what. With this law, the blogger or reader will immediately recognize that this is a constant's name and does not change in the program. The following code snippet demonstrates the difference (Java Programming Style Guide).

```
Day = (3 + number Of Days) % 7; //NO! Uses magic numbers
Final int WEDNESDAY = 3;
Final int DAYS_IN_WEEK = 7;
Day = (WEDNESDAY + number Of Days) % DAYS_IN_WEEK; //Yes, self-documenting
```

Similarly, for naming of methods and parameters in this layer contracts are considered by INFC protocols.

8. Layer layout

Packing on the structure of the code is important in this protocol because it increases code readability. INFC protocol is characterized contracts for this layer.

Use three spaces character for denting the code. In the following code snippet, this contract demonstrates indented control structures.

```
Public class Hello World
  {
  ...public void greet User (int current Hour)
  ...{
  .....System .out .print ("Good");
  .....if (current Hour < AFTERNOON)
  .....{
  .....System. Out. Print ln (" Morning");
  .....}
  .....else if (current Hour < EVENING)
  .....{
  .....System. Out .print ln ("Afternoon");
  .....}
  .....else
  .....{
  .....System .Out .print ln ("Evening");
  .....}
  ...}
  }
```

Use blank lines to separate sections of the program's code. These sections usually are logical groups of programs' commands. So put a blank line at the beginning of each episode of program and use two blank lines before beginning any new method within the category. Also, use a space character on either side of an operator, after the comma in the arguments' list and the semicolon in the syntax of for. In summary, in this layer, empty and blank lines are used to improve readability of the code. The following code snippet is written according to the above contracts on INFC protocol:

```
// Prompt the user to enter the commission rate
  Simple IO. prompt ("Enter commission rate (as a percentage): ");
  User Input = Simple IO. Read Line ();
  Double commission Rate = Double .parse Double (user Input) / 100;

  // Compute and display the commission, rounding to the nearest cent
  Double commission = total Value * commission Rate;
  Commission = Math .round (commission * 100) / 100.0;
  System .out .print ln ("Commission: $" + commission);
```

Program lines may not exceed 80 characters. It means if the length of a phrase is greater than this number, we break the phrase based on the following principles (Sun's Code Conventions for the Java Programming Language).

Breaking after comma

Breaking after an operator

Aligning the new line with the previous line of the same phrase

The following code demonstrates the use of this law:

```
Some Method (longExpression1, longExpression2, longExpression3, longExpression4, longExpression5);

longName1 = longName2 * (longName3 + longName4 - longName5) + 4 * longname6;
```

9. Layer Comment

In this layer of INFC protocol, it can be divided the program to different parts using empty lines, then adding comments to each section gives the possibility to programmer or developed team to detect quickly the portions of the code where particular manner happens.

Add single line comments always when writing code since their adding after the completion of the program may not well articulated the program's logic. Comments that are provided by this layer are as follows (AmbySoft Inc.'s Coding Standards for Java).

9.1. File Header Comments

These kinds of comments provide the following information for programmers or community of developers:

9.2. The writer of program

Providing statistics about the difficulty of the program based on the required time to complete the program

9.3. Invested time to complete program

The following code snippet shows the file header comment in the presented protocol.

```
//*********************************************
*
// Assignment: Program 2
// Account: (Enter your cs110 account number
here)
//
// Author: (Enter your full name here)
//
// Completion time: (Enter the total number of
hours you
//              spent on the assignment)
//
// Honor Code: I pledge that this program
represents my own
// program code. I received help from (enter the
names of
// others that helped with the assignment, write
no one if
// you received no help) in designing and
debugging my program.
//*********************************************
**
```

10. Single-Line Comments

In INFC protocol, these types of comments provide summary explanations for each section of the Code that means for three to seven lines of code, a comment is placed. These types of comments are also said inline comments. Single line comments start with / / and inform the compiler that avoid the translation of this line. The following code is written based on the above contract.

```
// Compute the exam average score for the
midterm exam
  Sum of Scores = 0;
  For (int i = 0; i < scores .length; i++)
    Sum of Scores = sum Of Scores + scores[i];
  Average = float (sum Of Scores) / scores .length;
```

10.1. Instructional Comments

Instructional comments provide explanations for each line of code. This kind of comment started with / / and are put in the right hand of program lines. Instructional comments explain the logic of each line of the program. In general, the comment layer will not offer the use of instructional comment from INFC protocol. In other words, instead of writing opaque code and adding instructional comments for it, INFC protocol prefers the use of meaningful variables and attempt to write self-citation codes. The following program snippet is not self- citation, since it is not used meaningful IDs. That's why using the instructional comment such as the following has not be offered by the proposed protocol.

```
ss = s1 + s2 + s3;     //add the three scores into
the sum
a = float (ss) / x;     //calculate the mean of the
scores
```

Transferring program logic to the programmers' community by INFC protocol

In the real world using INFC protocol is important for program developers due to the following reasons:

80% of lifetime cost of a piece of software is spent to software's maintenance. The biggest factor for this cost is the misunderstanding by the programmers. It is show that any program rarely during its lifetime maintained by the original author, thus, when understanding the code is hard by the community of programmers, the community of developers can spend more time reading the program's code and may create false change it. Here INFC protocol increases the readability of programs and allows the community of programmers to understand the logic of the new code more quickly and thoroughly. In this case the cost of program maintenance will be reduced.

Using INFC protocol can be adopted the right approaches of writing the program from the beginning and can be avoided the bad habits of writing the program and re-correction of the program's code. INFC protocol causes that programmers' team to easily understand each other's codes and to detect the points which contain error. Also, reuse of code and sharing of it between team members is possible and it is from other benefits of the proposed protocol.

MTTC is the criterion by which it can assess the sustainability of the product. The less the MTTC, the high the ability to maintain the product will be. Fig.3 shows the effect of the proposed protocol on software maintainability.

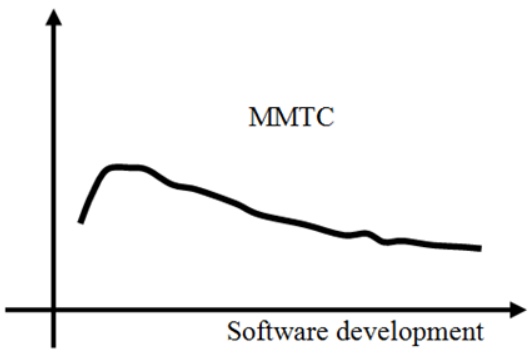

Fig.3: The impact of the proposed protocol to the ability of software maintenance

The impact of the proposed protocol to the ability of software maintenance

The accuracy and use of INFC protocol in programming reduce logical errors in the program's code. This feature leads to time reduction in debugging logical errors in the program.

Figure 4 shows the impact of INFC protocol on the logic error of software. Using of this protocol causes to reduce the logical error of program and to understand rapidly development team of the logic of program.

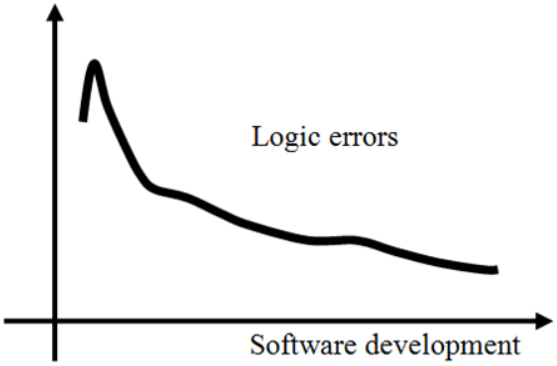

Fig.4: The impact of proposed protocol on logic error

10.2. The impact of proposed protocol on logic error

INFC protocol shares the program's logic between programmers' team members or community of programmers in the Fig.5 in order to reduce the life cost of software.

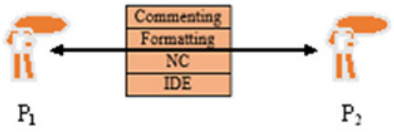

Fig.5: The relationship of programmers by INFC protocol

Fig.5 shows the using INFC protocol, P1 programmer can easily reach to the written logic of code by the P2 programmer, so using of this protocol improves program's readability, and programmers can easily communicate by INFC protocol.

Fig.6 shows the logic of program easily transmitted by INFC protocol to the community of programmers.

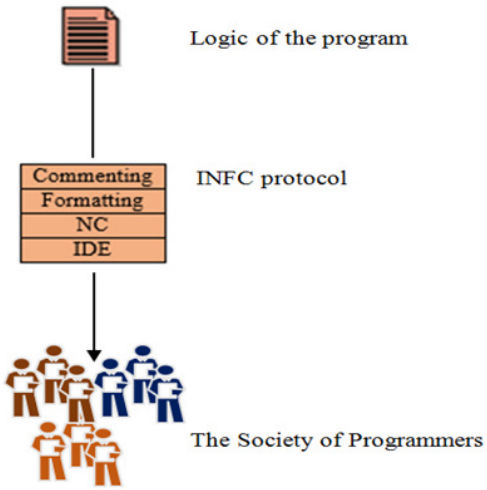

Fig.6: Access to the logic of the program by INFC

10.3. Future Work

For future work, we assume using data mining and ICNF protocols, program's logic can be extracted easily by the other bloggers' community. Because the most important information that lies within a code is the code's logic. In this case with detection of program's logic, the way opens for other developers' community.

11. Conclusion

Nowadays, since programming codes are very long and complex, each program through its life is rarely kept by the original author. So there is no mechanism to directly provide program logic for other programmers. In this case, developers' community can spend more time reading code. As a result, the cost of a piece of software's life will increase. In this paper, a stack protocol called INFC has been offered to improve readability and communication between programmers. Using of this protocol causes the program's readability will be independent of a language's syntactic which increases the ability to write. Extracting the program's logic, simple relationship between the developers' community, reduction of logical errors, maintenance capability, learning to write proper coding, high quality and easy to realize the code are the benefits of the INFC stack protocol.

References

A. Abran, and et al. "Guide to the Software Engineering Body of Knowl1088 edge", Los Alamitos, CA: IEEE Computer Society Press, 2001. This contains a 1089 detailed description of the body of software-requirements knowledge. It may 1090 also be downloaded from www.swebok.org.

AmbySoft Inc.'s Coding Standards for Java. http://www.ambysoft.com/javaCodingStandards.html.

B. Kovitz, L. Practical, "Software Requirements: A Manual of Content 1095 and Style", Manning Publications Company, 1998.

Java Programming Style Guide, http://www.javaranch.com/style.jsp

proach presented emphasizes deliberate up-front planning, requirements devel1123 opment, and architecture work followed by careful project execution. It provides 1124 long-range predictability of costs and schedules, high quality, and a moderate 1125 amount of flexibility.

S. Lauese, "Software Requirements: Styles and Techniques, Boston, Mass", 1093 Addison Wesley, 2002.

S. McConnell, "Software Project Survival Guide. Redmond, WA: Microsoft 1121 Press", 1998. This book presents one particular way to conduct a project. The ap1122.