Difficulty Analysis of Compiler Error Messages from a Viewpoint of Communication Protocol Layers

by Yoneharu Fujita, Mikiko Tanaka and Keisuke Nishijima

Address:
Department of Computer Science and Intelligent Systems,
Faculty of Engineering, Oita University,
Dannoharu 700, Oita City, Oita Prefecture,
Japan 870-1192
Tel.(097)554-7879, Fax.(097)554-7886
Yoneharu Fujita(e-mail: fujita@csis.oita-u.ac.jp)
Mikiko Tanaka(e-mail: mik_tanaka@itg.hitachi.co.jp)
Keisuke Nishijima(e-mail: nisijima@csis.oita-u.ac.jp)

keywords: communication, protocol, message, difficulty

Abstract

In this paper, we describe about difficulty in understanding of error messages given by a compiler from a viewpoint of communication protocol layers analogous to that of computer networks. In the case of human-human or human-machine, layered communication protocol model is introduced and it is shown that several factors which cause communication obstacles are explained based on the model. As an experiment, we gave several programming subjects of C to a student, then we analyzed the error messages given by a C compiler against the sample programs written by the student. The result is that the error messages from a compiler are easier to be understood if they are can be processed at lower layer for users. Furthermore, we gave some proposals for improvement of error messages.

1. Introduction

Currently, communication between human and non-human agents becomes important because of many intelligent non-human aids come into wide use. Initiative of communication between a human and a machine until now is taken by a human because machines have been quite passive in their behaviors. Current situation is greatly different from past by the emergence of computer driven machines. Such machines give humans natural language or some symbolic messages which require the humans some complex interpretation to get information from the messages in order to use the machines properly, that is, communication between machine and human became more complex.

This reports describes that there can be assumed to exist some protocol layers of communication between human-human or human-machine which are analogous to that of computer communication. Then it is described that difficulty in understanding error messages given by a compiler are closely related to layers required to process them.

2. Protocol Layers of Communication

Protocols of communication in general are agreements of format on which some physical/symbolic media carries information. In this sense, there exist great many protocols which are not explicitly described. Furthermore, various information processing systems are considered to be realizations of such protocols. It is well known that different information is obtained from one sentence by different processing algorithms, that is, interpretation protocols. These facts show the importance of problems of protocol selection for success of communication. A typical and rather high level example which shows this importance is an utterance "Can you reach the salt?" If the receiver of the question takes the surface semantic layer, the communication fails. It is considered to be used very complex and diverse protocols in communication between human and human. As a first approximation, we assume following six layers which are physical, code, symbolic, relational, inferential and motive. Of course these are not sufficient to describe all of human communications, but these layers are useful to analyze communication between human and current machine.

PL(Physical Layer): Physical/Chemical channels which correspond to functions of visual, auditory, tactile, smelling and tasting senses.

CL(Code Layer): Resulted code of interpretation of PL signal by perceptive organs.

SL(Symbol Layer): Composition of codes which correspond to conceptions.

RL(Relation Layer): Information and knowledge which are represented as relations between conceptions. The "relation" at this layer is supposed to be expressed as a frame.

IL(Inference Layer): Inference which derived from relations. Device of inference is knowledge of implication or cause-effect forms.

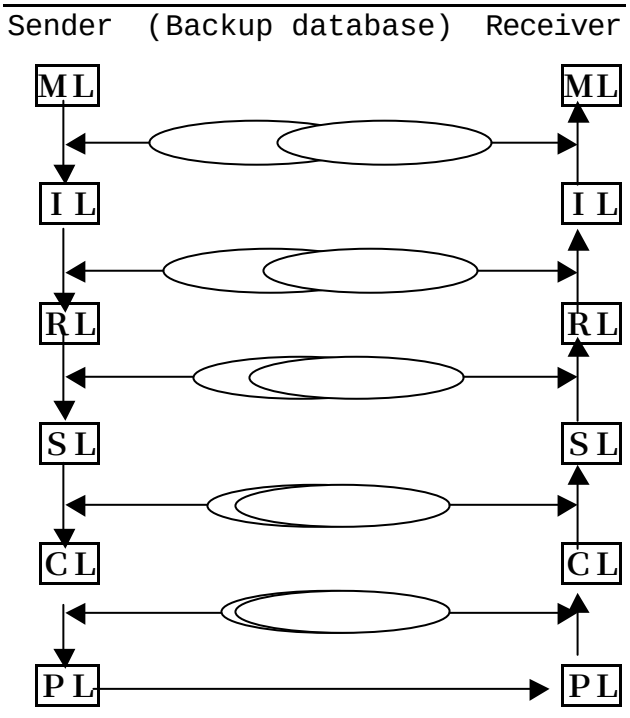

Fig.1 A multi-layered communication protocol model.

ML(Motive Layer): The layer of motive or intention of utterances.

In Figure 1, the ovals show backup database to interpret input, which correspond to communication protocols.

The following examples are failures in communication at each layer, which illustrate the characters of the layers.

[Example 1] Suppose a Japanese A and an American B are at a same breakfast table.

(1) A→B: D-ga-o ga y-ga-o-ga ga-ga l-ga-k ga-ga-ga?
(2) B : ....?

Explanation: This case shows that B couldn't hear A's utterance because noisy environment.
(failure at physical layer)

(3)A→B: Do you like lice?
(4)B→A: Oh, you are kidding me!

Explanation: This case shows that, A couldn't pronounce "r" and "l" differently as an ordinary Japanese. A intended "kome(rice)" but he couldn't translate the code "r" to the correct physical signal.
(failure at code layer)

(5)B→A: Do you like insects?
(6)A→B: Yes, Their songs sound beautiful.
(7)B→A: Songs?

Explanation: B feels sound of insects noisy. This is caused from the difference of concepts of insects.
(failure at symbol layer)

Concept of sound of insects is:

A: Insect(IS_A: Animal, INSTANCE: Cricket, SOUND: Beautiful)

B: Insect(IS_A: Animal, INSTANCE: Cricket, SOUND: Noisy)

(8)A→B: I am keeping some crickets.
(9)B→A: Why?

Explanation: Ordinary, they have no habit of keeping crickets for appreciation in B's country, that is, B couldn't imagine a relation of keep/kept between human and crickets.
(failure at relation layer)

(10)B→A: Are you a biologist?
(11)A→B: No. I'm a computer scientist.

Explanation: B used the following knowledge:

Keep(AGENT: x, OBJECT: y) & Human(x) & Insect(y)
→ Observe(AGENT: x, OBJECT: y)
Observe(AGENT: x, OBJECT: y) & Human(x) & Insect(y)
→ Biologist(x),

on the contrary, A used the following knowledge:

Keep(AGENT: x, OBJECT: y) & Human(x) & Insect(y) →
Appreciate(AGENT: x, OBJECT: sound(y)),

that is, different knowledges are applied to the same event.
(failure at inference layer)

(6) A→B: Have you already adjusted your watch?
B→A: Oh, it is 8:30 now.

Explanation: A confirmed B about adjustment of the watch to Japanese standard time out of A's kindness, whereas, B received it as asking the time.

(failure at motive layer)

3. Height of Layers Required in Communication and Difficulty

Communication is realized in various layers. Difficulty of communication relates to the height of layer required in processing messages.

[Example 2] A simplest case of communication between humans.

A: Good morning.
B: Good morning.

In this case, A and B are considered to have used only code layer, that is, B is required only to discriminate the words. B may be a parrot to perform communication. In many cases of peoples in a common knowledge group, communication is often performed on very low layers, even if the contents of communication are very complex and abstract. This shows that the complexity or difficulty of communication is measured by the height of the layer which is required to perform communication.

[Example 3] Suppose it is night.

A: Good morning.
B: Are you kidding me?

At least, B is must know the meaning of "morning" in this case. This process is not at the code layer but at the symbol layer.

4. Error Messages of a C compiler

In the case of communication between human and machine can be considered in a similar viewpoint. We will discuss in detail of the case understanding C compiler's messages.

[Example 4] Let's consider an warning message "warning: passing arg 1 of `printf' makes pointer from integer without a cast." given by a C compiler against the following statement:

```
printf('¥n');
```

This message contains some issues to be considered.

First, though this message is in a form of warning, the compiled program causes a fatal error "Segmentation Fault" if it is executed. An straight interpretation of the term "warning" makes the state serious. This term should not be interpreted at the layer of symbol but at a higher layer.

Second, some terms such as "pointer" and "cast" are not familiar to novice users. These terms cause a symbol layer obstacle.

Furthermore, the sentence "makes pointer from integer" throw novice users into confusion at the relational layer. To understand the relation "make" between "pointer" and "integer", it is necessary to know the architecture of computers. Some inferences based on the function and concept of "printf", "pointer" and "integer" are necessary for a correct understanding of the situation.

[Example 5] An example of logically definite but syntactically ambiguous message.

```
1. #include <stdio.h>
2.
3. main()
4. {
5.    int c;
6.    c=getchar();
7.        if((c>=’0’)&&(c<=’9’))
8.            c=t;
```

```
9.       else
10.          c=f;
11.   putchar(c);
12.}
```

```
<error messages>
e1.c: In function ‘mail’:
e1.c:8: ‘t’ undeclared (first use this function)
e1.c:8: (Each undeclared identifier is reported only once)
e1.c:10: ‘f’ undeclared (first use this function)
e1.c:10: (Each undeclared identifier is reported only once)
```

Straight interpretation of the above error messages leads to insertion of declarations of identifiers t and f, which remove syntactic defects of the above program, but the modification causes substitution of indefinite value to c and will cause fatal error at executing “putcahr(c)”. The statement c=t is logically inadequate because the value of t is not given. In order to keep logical consistency, “t” must be definite, that is, if there is no declaration of “t”, “t” must be a constant ‘t’. The same is “f”.

5. Classification of Messages Based on Required Layers

We have given students several C programming subjects and collected error messages, then analyzed them from the viewpoint of communication model described in chapter 2. 13 programs caused error messages. 2 of them required code layer processing, 1 of them required symbol layer processing and the others required inference layer processing to understand them. This result shows the difficulty of understanding C compiler's error messages.

In a comparison with the above result, we applied "lint" which is a detail syntactic analyzer of C source program to the same example programs, then we obtained more code layer messages but a few messages related to linkage loader were not included as a matter of course.

Furthermore, we extracted error messages from a part of source program of C compiler and analyzed them.

6. Improvement of error messages

From the above analyses, we obtained some ideas of improvement of compiler error messages which are based on several different methods.

(1) Preparing hyper text of error messages to explain terms in the messages. This is mainly a measure to the symbol layer obstacles.

(2) Preparing a database which is constructed by pairs of frequently generated error statements and corrections of program statements corresponding to the error messages and their statistics. This is a measure to obstacles at all layers. The following example is this type of message.

Doubtful expression: printf('¥n'); This is often a miss typing of printf(“¥n”);.

(3) Making error messages with more concrete form. To realize this, it is necessary to know more about mechanism of a compiler and information from lower functions of a compiler. This is a measure to the requirement of preparing messages understandable at code layer. The

Proceedings of 9th IEEE International Workshop on Robot and Human Communication, pp.18-22, Sept.(2000).
[2] Y.Fujita, ”Communication protocol layers and understanding of communication”, proceedings of 17th annual conference of JCSS, P2-15, pp.158-159, Jun.(2000),(in Japanese).
[3] R.Wilensky, Y.Arens and D.Chin, “Talking to UNIX in English: An Overview of UC”, CACM, Vol.27, No.6, pp.574-593(1984).