

# Intelligent Software Programmable Logic Controller

Young Im Cho

Dept. of Computer Science, Pyongtaek University

111 Yongi-dong, Pyongtaek, Kyongki-do,

[yicho@ptuniv.ac.kr](mailto:yicho@ptuniv.ac.kr)

**Abstracts-** Recently, as the open control system controlled by PC instead of PLC is developed, the market of softPLCs on PC is rapidly growing, but the competitive editor and running engines for softPLC are rare in the world. Therefore, I developed an intelligent multiagent based softPLC (IMPLC). In IMPLC, the standard IEC 1131-3 PLCs (LD, SFC, FBD, ST) programmed by a user are converted to IL, which is one of intermediate codes, in order to make them interact. And then the IL is converted to the standard C code regarding some extension and transplanting, which can be used in a commercial editor such as visual C++. In IMPLC, the logical errors and syntax errors occurred by users are detected, so that the optimal PC control based softPLC can be possible. IMPLC provide easy programming platform to beginners as well as professionals. The study of code conversion is firstly tried in the world as well as KOREA.

## 1. INTRODUCTION

PLC program is a kind of an order given to PLC, so that PLC can work. PLC program is developed in many directions. If one is proficient in sequence control such as relay, rather than sophisticated knowledge, the PLC program can be adequately embodied. PLC program is structured to be used by selecting adequate one, according to features of programming languages depending on control content and applied fields [1].

PLC programming languages, however, are different depending on countries and computer types, and they have not been standardized; thus, a problem that it is not suitable to the openness and dispersed age is entailed. The standardization activity of PLC programming language centered on Europe in 1993 was promoted, accordingly. This standard is the IEC 1131-3, which had been reviewed for over 10 years in TC65/SC65B/WG7/TF3 of IEC, which was approved in 1992 and published as a document in 1993 [2].

In IEC1131-3, the following 5 languages were prescribed and handled. Namely, there are instruction list (IL) and structured text (ST) as the language of text system, while ladder diagram (LD) and function block diagram (FBD) are defined as a graphic system language. As important common

factor, sequential function chart (SFC) is defined [3, 4]. IL is widely used in Europe including Germany, and FBD is the similar type of language as circuit chart for application accompanied by signal flow of process control. ST is similar to the programming language developed as real time application; therefore, it is effectively used for the defining complicated function block.

Despite establishment of standard language, IEC1131-3, PLC programming language is a still difficult language for general public to use. It also is difficult to have general usability under the Web environment. The PLC programming language has a problem to find logical error when programmed by an expert, as well.

The preceding research result, ISPLC (Intelligent Agent System based Software Programmable Logic Controller) [6] to solve this problem, has studied the standard language regarding LD (Ladder Diagram), which is used more than 90% among the 5 languages [8] that were established as international PLC standard language, and it was the technology that changed this to IL (Instruction List), which is an intermediary code, and converted to standard C code utilizable in the existing commercialized editor (Visual C++). However, ISPLC tried only conversion regarding LD language; thus, conversion regarding other languages was not conducted. It has a weakness that the language types that are converted were very limited and not general.

Accordingly, in this paper, I intend to research and develop IMPLC (Intelligent Multi Agent System based Software Programmable Logic Controller) to enhance efficiency, so that general public, who are accustomed to high level of language by converting FBD (Function Block Diagram), SFC (Sequential Function Chart), ST (Structured Text), IL (Instruction List) to intermediary code IL, as well as LD language among IEC1131-3, and utilizable standard C code in commercialized editor. The reason to convert to IL intermediary code is to raise mutual relevance between codes by making all the PLCs uniform type. The reason to convert to C is to enhance re-utilization of C code.

In IMPLC, primary error search is conducted in the process of four languages (LD, FBD, ST, SFC) being converted to IL. Also, secondary error search is conducted in the process of being converted to C; thus possible number of errors is remarkably reduced, and logical error detection function can be

conducted. In view of all this, it is very efficient. By handling all the behavior modes in which users perform by multi agents, while providing GUI based interface, I intend to provide platform by which users who are accustomed to PLC, as well as beginners can conduct programming.

I also intend to decrease programming time very much, as well as build GUI based Web environment and search programming error, so that efficient control can be made by applying IMPLC to 3 way conveyer belt used in the real industrial site.

The structure of this paper is as follows: In Chapter 2, the need of IMPLC was presented by analyzing the domestic softPLC status. In Chapter 3, IMPLC was proposed, and in Chapter 4, case study, which was applied to system, was analyzed. Finally, in Chapter 5, conclusion was drawn.

## II. SURVEY OF SOFTPLCS

Among PC-based control products developed for the overseas industrial sites [7,8], there is KW system, which has been developed as commercial system frame, which is the system to provide visual interface in consideration of user's familiarity regarding the frame. There is also Embedded Super PLCs [9, 10].

In Korea, there is no case to be developed for industrial use, but Real Gain Inc. supplies PLC learning package (RealPLC). Using RealPLC, various languages can be learned with only PLC software, and a variety of monitoring and animation can be done, which provides a feature to learn premier PLC with smaller cost.

Especially, it provides various kinds of animation systems; thus, a program can be verified within PC without target system. Also, it is configured to make it possible to learn the entire international standard PLCs of SFC, IL and FBD, as well as LD. This product, however, is difficult to be utilized for industrial use, since it is limited for educational use [11].

There are PLC systems to be developed and used by LG Industrial Systems Co., Ltd. and Samsung. Although Samsung developed WinGPC for windows and LG Industrial Systems Co., Ltd. developed Master-K, general usability is insufficient, since they provide only interface for equipment for the

The following Table 1 has compared and analyzed strengths and weaknesses of the KW system, which is a domestic representative softPLC and interface of Real Gain system. Both systems provide the interface of standard languages set in IEC1131-3, however, compatibility between standard languages is not sufficient, and code conversion as high level language is not provided.

Table 1. Comparison of KW System and RealGain

Compare Advantage with Disadvantage between KW and Real Gain System			
KW System		Real Gain System	
[ Point of Tool Characteristic ]			
Advantage	Disadvantage	Advantage	Disadvantage
Support STL	Optional	Learning HL	LD based
Conversion	Hard Convert	Monitoring	Only Simulation
MultiProg,Fox	WinPLC ProComDS	Animation	Restricted Demo
Window based		Easy Experiment	Error Message
Evaluation,SL		Low Cost	
MultiTasking		Download	
[ Point of User and Customer ]			
Advantage		Disadvantage	
User Friendly (GS)		More closed Option	
Drag and Drop		Black Boxes	
Monitoring Simulation		Hard to Error search	
Select each language		User Handling /Expert's	
		High efficient PC based	
		Ladder/Simulation	

## III. IMPLC

### A. Introduction of IMPLC

In this paper, I intend to develop IMPLC (*Intelligent Soft Multi agent based Programmable Logic Controller*), which is intelligent PLC software agent system based on automatic, autonomous, specialized and integrated type of PC in consideration of required matters by generating optimal code via user-desired filtering of the result, after providing user-desired and suitable component on behalf of the user within PLC programming possible frame with regard to agent [12, 13] based PLC software system.

In this system, four standard languages provided in the IEC 1131-3 are converted to the IL, the remaining language; so that compatibility between languages is possible. As a result, the user control under the optimized environment by correcting logical error generated based on PC by compiling in the standard C compiler was conducted.

The components of IMPLC, which is a PC-based intelligent softPLC editor proposed in this paper, consist of agent group, component data storage and rule base architecture.

There are four agents in the agent group:

User Agent (UA), Scanning Agent (SA), Control Agent (CA), Error Check Agent (EA). Component data storage functions to recommend and manage the programming language used greatly by a user. There are five modules in rule base architecture:

Module converted to IL    Module in which IL is converted to C    Error check module    Memory mapping module    User learning file module.

The editor screen of IMPLC is like <Figure 1>. The overall concept of configured IMPLC editor screen is configured that standard languages (LD, SFC, FBD and ST) of IEC1131-3 can be converted to intermediary code type, IL and this IL is converted to high level language (C ) and compiled; thus, it is configured to be executed under the Windows environment. Editor is GUI based, and that error correction and code conversion are conducted intelligently by agents can be a feature.

### 3.2 Modules in IMPLC

Overall structural chart of IMPLC is shown in Figure 2.

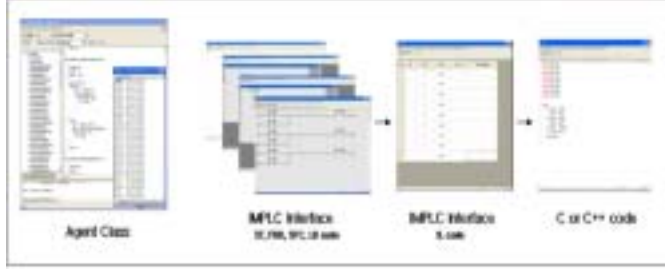


Figure 2. Overall IMPLC System Structure

Each module's features and roles of IMPLC system in this paper are as follows:

(1) Agent Group: This refers to a group of agents acting within the IMPLC system. The agent group of IMPLC in this paper consists of 4 agents. Each agent's role is as follows:

a. Scanning Agent: This plays a role of interface for interaction between user and control agent, and this agent is conducted when a user bring up a file. The performance and behavioral pattern of scanning agent is seen in Figure 3.

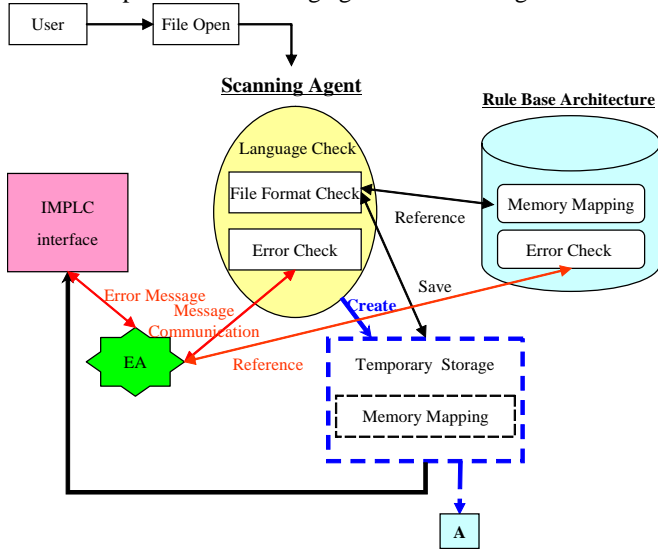


Figure 3. Scanning Agent

As for behavioral procedure of scanning agent, scanning agent checks the drawn up language by reading file extension, after identifying concerned file's presence (i.e. LD, FBD, etc). Then, it converts suitable to operator and operand and saves in the temporary storage by consulting memory mapping module of the rule base architecture. After searching logic or grammar errors through error check agent (EA) and message communication, this functions to report an error message to the user. If there is no error, the next program is performed.

b. User Agent: Like the scanning agent, this plays a role as interface for interaction between a user and control agent. When a user edits in the IMPLC, this agent is conducted by consulting component data storage. The user agent performance and behavioral pattern are seen in Figure 4.

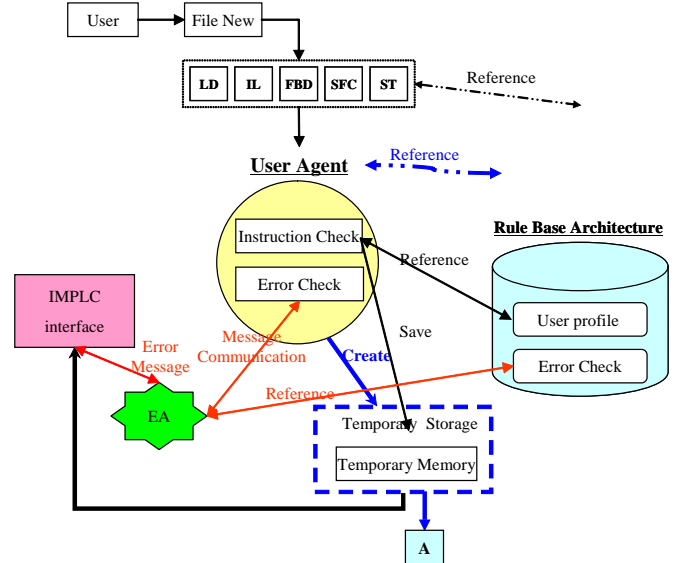


Figure 4. User Agent

As for the user agent's behavioral procedure, when a user selects language in the IMPLC, operator is loaded by consulting component data storage, and the user edits, it is saved in the temporary storage on real time. User agent recommends the component suitable to user's tendency by consulting user learning file module in the rule base architecture. Also, this agent searches error status through EA and message communication and functions to report an error message to the user.

c. Control Agent: This is the agent, which is executed by consulting rule base architecture when converting the converted IL to C, after converting each language to intermediary code type of IL through interaction between the scanning agent and the user. The performance and behavioral pattern of the control agent is seen in Figure 5.

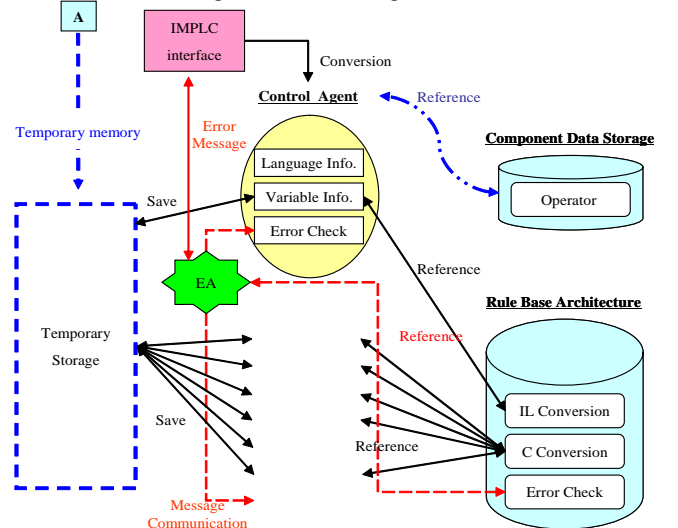


Figure 5. Control Agent

As for behavioral procedure of control agent, information regarding the language edited by a user, the location

information of operator edited by a user and operand of concerned language is saved in the temporary storage. And, the control agent is primarily involved in converting IL by consulting conversion module to IL of the rule base architecture with the saved information. After searching IL information (information on label, operator, operand, comment and function and function block) to convert IL to C, the control agent is involved in the process of converting to C by consulting the conversion module to C of the rule base architecture, secondarily.

d. Error Check Agent: The performance and behavioral pattern of error check agent is seen in Figure 6.

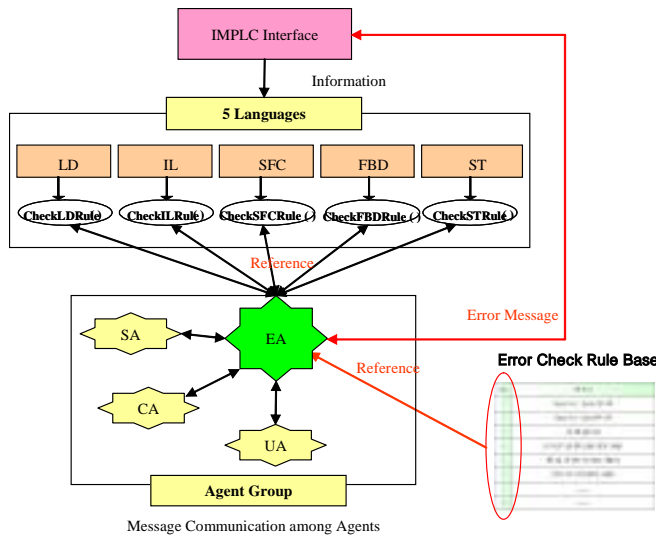


Figure 6. Error Check Agent (EA)

When a user edits or brings up a file, a check is conducted if each language is suitable to the standard language rules of IEC1131-3 by consulting rule base architecture error check module. Internally five languages' standard language rules were defined through classification. The most remarkable feature of the error check agent is that it reports error status to the user through message communication with the scanning agent, user agent and control agent.

(2) Rule Base Architecture: This is in charge of major language conversion function of IMPLC.

a. Memory Mapping Module: Based on the standard of IEC1131-3, memory mapping rule that converts all the saved files read and saved from IMPLC to automatic type has been modularized. The reason of the need of this is to make it possible to use the PLC file edited in other external editor, as well as the file edited by IMPLC. For memory mapping, the module that has the rules to convert automatically is needed. In this paper, this was defined to be automatic memory conversion. In memory mapping, graphiCs (LD, FBD and SFC) were excluded, since graphic saving mode is different by each company, and conversion is impossible. Thus, only text based language (IL and ST) are converted.

b. Conversion Module to IL: This is the module to be involved in the process to convert each language to IL. As seen in Figure 7, each language is converted to IL through scanning conversion algorithm.

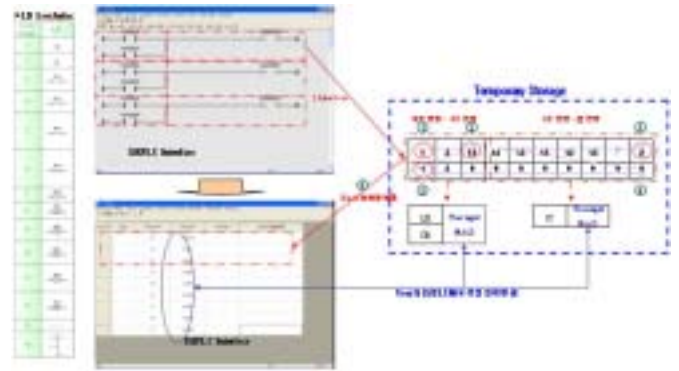


Figure 7. Conversion Process to IL

The above Figure 7 is an example of LD language. To find out location of contact point, the contact point is divided into starting contact point, OR contact point and end contact point. To get location and information of operand and component information of each language, component data storage is consulted. In this paper, contact point is divided by programming language and each is provided index, and is saved in the component data storage to be used in the case of language conversion.

c. C Conversion Module: This is the module to be involved in the conversion process of IL to C. As seen in Figure 8, IL is converted to C through scanning conversion algorithm.



Figure 8. Conversion Process to C

This is the module to convert IL conversion to C. To convert to C, the information on label, operator, operand and comments of IL is needed. Through C scanning conversion algorithm with the above information, IL is converted to C. The information on label, operator, operand and comments were defined by classifying grammars defined in IEC1131-3.

d. Syntax & Logical error Check Module: This conducts scanning all the editing work prepared by a user with error check module. This is the module that the error message is proposed to the user through CA, UA and SA and message

communication by EA's consulting error rule base. The error rule base is defined by classification of standard rules of IEC1131-3. This classification is periodically scanned by the error check agent (EA).

e. User Profile Module: This module makes user's program editing process pattern and saves component or classified file type with increase of frequency as individualized code. In this module, a user presents a suggestion with the components on the editing process of the previous program and history of class pattern on the interface.

(3) Component Data Storage: This is the module with grammar configuration factors such as operators and operands of five standard languages in the IMPLC system. This module helps a user in editing in the interface by providing user-desirable data, namely components, to UA.

IMPLC system helps for a user to easily conduct programming through control agent's involvement in the delivering user-desirable data, converting each language to IL and then to C by consulting of user agent, scanning agent, the rule base and component data storage to be suitable to the questions of the user in the case of user's editing or bring up a file through interface. Also, reuse becomes possible with the code converted to C finally.

## IV. SIMULATIONS AND EVALUATIONS OF IMPLC

### A. 3way Conveyor Belt

In this paper, I intend to simulate by applying the system of IMPLC to 3 way conveyor belt control system used in the industrial site. This system is used when moving or loading conveyed goods. Its input and output list is as follows:

The configuration chart of the 3 way conveyor belt system developed in this paper is seen in Figure 9.



Figure 9. 3 way Conveyor Belt System Modeling

In Figure 9, three red colors indicate sensor of the 3 way conveyor belt. Timer setting time for 1, 2 and 3 way was three seconds. When we check the conveyed goods dropping sensor, %IX0000, the 1 way belt, %QX0000, is on before the conveyed goods meet the vertical sensor, %IX0001. Once 1 way belt meets the vertical sensor, %QX0001, it stops. When conveyed goods go down to the 2way belt and %QX0001 is on,

and when it meets the 2 way vertical sensor, %IX0002, it stops 3 seconds later. When conveyed goods go down to 3 way belt, and %QX0002 is on, the conveyed goods are on before it meets 3 way belt vertical sensor, %IX0003, 3 sec later [13].

The behavioral process drawn up and implemented on the IMPLC is seen in Figure 10.



Figure 10. 3 Way Conveyor Process on the IMPLC

The above Figure 11 indicates the process that a user draws up as LD, converts it to IL and then to C code, in an example of 3 way conveyor belt, step by step. The monitoring of this process is seen in Figure 11.



Figure 11. 3 Way Conveyor Belt Monitoring

### B. Evaluations of IMPLC

To measure time to be required in logical error debugging of IMPLC and existing software, I experimented, increasing the number of operator patterns of the LD programming language in 3 way conveyor belt. As a result of the experiment, since IMPLC should check control-related operators in specified sector within the program and identify cause according to the message flow order, depending on logical error, I found a tendency to have

$\sqrt{n}$  ( $n$ : The Number of LD Ooperation) of debugging steps on average was shown, although the program gets complicated. However, because error causes should be identified, while all the control-related components should be checked and correlation of all the operators should be analyzed, according to control flow order when execution was not



conducted after drawing up LD program in the case of existing software, PLC, I found it had ~~4~~ number of debugging steps. Accordingly, as LD contact points increase, error detection possibility diminishes. When I indicated this with a graph, the number of debugging steps was found to increase in exponential function.

When using IMPLC, accordingly, logical error was remarkably reduced compared with using the existing software PLC.

## V. CONCLUSIONS

In this paper, IMPLC, which is softPLC, was embodied. By applying IMPLC to agent-based 3 way conveyor belt, along with mapping of IMPLC into each language, C or Visual C++, each language was converted to IL and then IL was converted to C in IMPLC. Thus, there was a merit to check logical error by the intelligent agent on the standard C compiler.

Although IMPLC provides the code conversion such as IEC1131-3 standard language  $\rightarrow$ IL $\rightarrow$ C code under integrated environment, code conversion in a reverse relation is not performed. Thus, actual user should correct the standard language program again from the converted code, when an error is detected, which generates a weakness that utilization capability of development frame should be superior. Namely, IMPLC explains the code conversion relationship to users in semi-automatic form, rather than full automatic form. Also, the function of the agents used in IMPLC is somewhat restricted; thus, more efficient agent algorithm should be developed.

To solve restrictions of IMPLC and make it more intelligent system, more efficient and structural environment, user's performance ability and more effort for expert system are greatly required. For actual industrial use, the development of two way conversion module between the standard language with IL as intermediary code and IL, and between IL and standard languages is required. When actually applying IMPLC through PLC execution engine and network communication, I think error detection function without a problem on the program will be more strengthened. Also, we need to develop an integrated system that can be used in industrial sites by adding network communication function to IMPLC additionally.

## REFERENCES

- [1] PLC Theory and Practice, Internal Education Data of Samsung Electronics Co., Ltd.
- [2] Norme Internationale International Standard, CEI IEC 1131-3, Premiere edition, First edition, 1993.
- [6] Cho Young-Im, Shim Jae-Hong, "ISPLC: Intelligent Agent based Software PLC", Autumn Theses Collection of Korea Multimedia Society, Volume 6, No. 2, pp.557-560, 2003.11.21-22.
- [7] [www.angelfire.com/in/bsommer/softplc.html](http://www.angelfire.com/in/bsommer/softplc.html)
- [8] IsaGRAF user's guide, Version 2.1, CJ International, 1994
- [9] [www.intellution.co.kr](http://www.intellution.co.kr)
- [10] [www.deltaww.com](http://www.deltaww.com)
- [11] Realgain Research Center, PLC Lab Practice, Chungmungak, 2003
- [12] Russell and Norvig, Artificial Intelligence a Modern Approach 2/E Chap 2, Prentice Hall International Co., 1994.
- [13] [www.fipa.org/repository/managementspecs.html](http://www.fipa.org/repository/managementspecs.html)