USE OF VISUAL C++ IN CREATION OF INTELLIGENT PROGRAM

Rashika Kaul, Saurabh Kumar
IT Department
Dronacharya College Of Engineering, Gurgaon, Haryana

***Abstract*- Visual c++ is a strong developing tool which was developed by Microsoft in February 1993 .Its operating system is windows. It is a frontend language which supports GUI. Visual c++ is an easy and cheap way for code quality enhancing. In our research paper we will focus on the history of visual c++ , various versions and various issues regarding it, also how it allows multiple applications to make use of the packages while only having to install it once. We will also study why visual c++ has a huge demand in market and also various advantages of it. Also we will study about the difference and the similarity between VB and VC++**

I. INTRODUCTION

Visual c++ is a commercial product from Microsoft which was developed to avoid problems of CUI developed by MFC. MFC is a collection of classes supported by visual c++ functions. It is available in different languages i.e English, Japanese, Korean, German etc. To function correctly many applications require packages.. Visual C++ redistributable and runtime packages are there which are installed for standard libraries that many applications use.

II. HISTORY

Microsoft C/C++ was the predecessor to Visual C++. Microsoft QuickC 2.5 and a Microsoft QuickC for Windows 1.0 were also there. The Visual C++ compiler is still known as Microsoft C/C++ .There are various versions like the 16 bit versions which had Microsoft C 1.0 which was Microsoft's first C product in 1983 and was based on Lattice C, C 2.0 which added large model support. C 3.0, C 4.0, C 5.0, C 5.1 were also there. 32 bit versions and 64 bit versions were also there. The current version is Visual c++ 2013 which was released on October 17, 2013. It supports features like C++11 AND C99 and introduces a REST SDK.

III. ISSUES

Due to many major compiler releases container sizes have varied a lot. Microsoft therefore recommends using C or COM interfaces, that have a stable ABI between compiler releases.

Due to different versions of C users can compile their code with any of the available libraries. This can cause some problems when we use different components in the same program. Microsoft has recommended to use the multithreaded dynamic link library to avoid possible problems.

The compilers did not support the later revisions of the standard C99 and C11 until Visual C++ 2012 came into use which added support for various C99 features in its C mode .Visual C++ 2013 significantly improved the C99 support, though it is still not complete.

IV. ADVANTAGES

MFC is provided in the form of DLL which saves time and memory. Backend is the database in which we store the data and which is not visible to us. Frontend is visible to us on the screen which we can see on the desktop known as forms. Visual C++ has a huge demand in market due to its simplicity and also MFC is the heart of VC++ and it came into existence only to support MFC. The latest version of the Microsoft Visual C++ development system is designed to improve the productivity not only with C++ applications but also with Internet, intranet, and mixed-language systems.

V. SIMILARITY AND DIFFERENCE B/W VB AND VC++

The function of VB and VC++ are same, they both are frontend languages which support GUI and run on windows .VC supports MFC and VB does not support it. VB is outdated and is now replaced by VB dot net but VC++ has a huge demand in market due to its simplicity.

VI. VISUAL C++ LIBRARIES

- Breaking Changes (ATL)
- Breaking Changes (ATL/MFC)
- Breaking Changes (MFC)
- Breaking Changes (CRT)
- Breaking Changes in the Standard C++ Library

VII. VISUAL C++ COMPILER

The Windows 95, Windows 98, Windows ME, and Windows NT platforms are no longer supported. These operating systems have now been removed from the list of targeted platforms. The compiler now no longer supports multiple attributes that were associated with ATL Server. The following attributes are no longer supported perf_counter, perf_object, perfmon, request_handler, soap_handler, soap_header, soap_method, tag_name