

journal homepage: www.ojs.sabauni.net



Article

C-MPE: A Collaborative Multiprogramming Development Environment for .Net Framework

Anwar Saif ^{*1}, Saeed Mohammed², Saleh Mohammed², Osama Ali², Majdi Abdu²

¹Information systems department, faculty of computer science and IT, Sana'a University, Sana'a, Yemen

²Computer science department, faculty of computer science and IT, Sana'a University, Sana'a, Yemen

Article info

Article history:

Accepted: Nov. 2020

Keywords:

Collaborative development environment;
pair programming;
.Net framework.

Abstract

Working in a team has a great influence on the success of a multi-user activity. Software development is a type of these activities that are highly collaborative between the software development team. Collaboration between the project's team improves the user experience which leads to improving the project quality. Thus, collaboration between the development team requires a development environment with collaborative capabilities. In this work, we develop a collaborative multi-programming development environment (C-MPE) that enables software developers to share the software components between geographically dispersed teams connected via a private network. C-MPE permits developers and programmers to build their own environment with custom properties and link them with a non-open-source framework such as the .Net framework.

* Corresponding author: Anwar Saif
E-mail: anwarsaif@su.edu.ye

1. Introduction

The development of software applications has gone through multiple stages, starting with a single developer in the project to paring programming [1], and then collaborative programming. Pair programming is a method of collaboration in which two programmers work on a programming task on the same machine. It is a collaborative method of working that involves communication, clarifying ideas, discussing approaches, and coming up with better solutions. While pair programming is beneficial, it requires practice and is constrained by the physical existence of the pair. In a collaborative development environment [2] the teamwork of a project, even if distributed by time or location, may share knowledge, discuss, negotiate, brainstorm, and carry out some task. The development and integration of large software systems is a difficult process for a single developer. Thus, the systems are divided into modules and components and then assigned to teamwork. The enhancement in the computer networks and the Internet made it easy to build teamwork even if they are located in dispersed locations and take advantage of these advances in migrating from a single and local environment to a global and distributed environment. Moreover, recently, the COVID-19 pandemic imposed social distancing laws, which had a major impact on forcing the teamwork to communicate remotely through private networks or the Internet.

Many open-source control and development environments and tools have emerged and support collaborative programming, project management, and control. GitHub [3], is one of these tools that enable developers to use the benefits of code hosting to access a project centrally. Developers can develop projects and contribute to an existing project no matter where they are in the world. Other similar tools that

enable public and private development collaboration are SourceForge [4], cloud9 [5], Eclipse [6], Microsoft Visual Studio live share [7], and Space [8]. These tools offer the ability for team members to work remotely, in real-time, to write, debug, and run code. These tools are powerful tools but most of them are commercial ones, only limited capabilities are provided in the free versions.

The .Net framework [9] is a software development framework developed by Microsoft and runs mainly on Microsoft Windows operating systems. It is constituted of two main components called Framework Class Library (FCL) and Common Language Runtime (CLR). The FCL provides the user interface, data access, database connectivity, cryptography, web application development, numeric algorithms, and network communications while CLR is a virtual machine that provides services such as memory management, security, and exception handling. It is responsible for executing programs written for the framework. Fig. 1 shows the major components of the .Net framework.

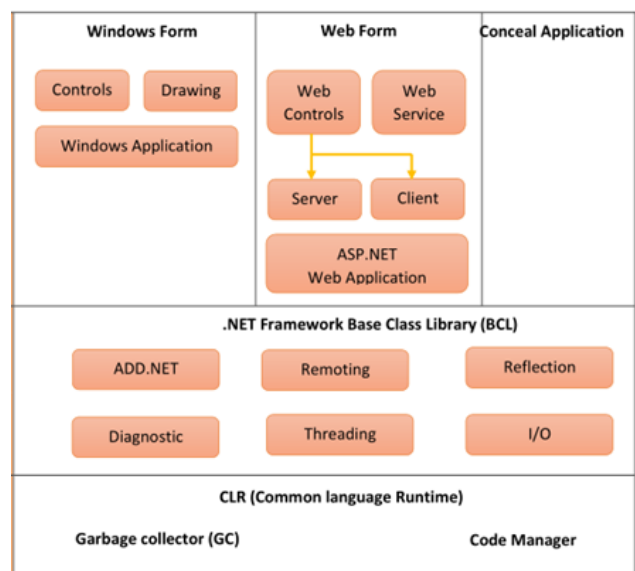


Figure 1: .Net framework architecture [9]

2. Related work

Many researchers have worked on developing an integrated environment that helps teamwork to communicate and share experience and code. Collaboration features help in improving communication between the developers which results in enhancing and speeding up the development process [10].

At the early stages of the collaborative IDE development, IBM developed a collaborative IDE called Jazz that adds a set of collaboration features for the Eclipse IDE [11]. Sangam [12] is another plug-in for Eclipse that introduces a shared editor and chatting for pair programming. Langton et al. [13] addressed this issue by presenting a tool called Group Homework Tool (GHT). This tool is developed to help programmers code in a synchronous collaborative manner. It concentrates on student group teaching assistants and tutors. GHT has many features such as HTML frames for an assignment definition, asynchronous code editor, chat, and a shared Whiteboard. The major drawback of this work is the high latency imposed during the communication. A collaborative IDE is described by Hupfer et al. [14]; the main purpose of the IDE is to provide a communication medium between team members enabling them to discuss issues related to the project. Hani et al. [15] presents the design and implementation of a collaborative IDE called ICI. It enables developers to edit, share, compile, run and debug software projects in real-time. Aditya Kurniawan et. al [16], proposed a real-time collaborative code editor called (CodeR) using web socket technology. CodeR provides workspaces to build and execute the source code in real-time in addition to the chatting capabilities. In order to obtain fictional collective programming scenarios, Chunhui Wang et al, [17], formed a group of physically-distributed programmers to

collaboratively solve a programming problem. He observed that any programmer's submission may possess information that could provide inspiration for others in the teamwork even if the submission is not completely correct.

The aim of this paper is to present our experience designing and implementing a collaborative integrated development environment called (C-MPE) that enables work teams connected by private networks to work in one project. It permits developers and programmers to build their own environments with custom properties and link them with the None-Open-source .Net Framework.

3. The Architecture of the C-MPE

C-MPE is a collaborative integrated development environment that enables developers to create projects, share programming-related tasks, and collaborate on a variety of software development activities in real-time such as editing, compiling, running, and debugging. It consists of two main parts; the first part is the IDE itself where all the project-related tasks can be accomplished. The second part is the core part of the C-MPE where we connect to the .Net Framework core and use it to compile and run the source code.

3.1. The integrated development environment

The integrated development environment is the first part of the C-MPE in which all development is performed. It combines the common activities of writing software such as editing source code, building executable, and debugging into a single application. In addition, it supports code auto-complement and color-based error detection, syntax highlighting, and autocomplete.

Figure 2 shows the IDE panes, the code source code pane, project explorer, output pane, and the list of the active developers.

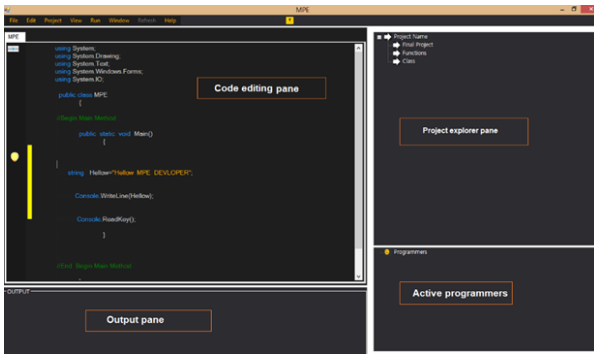


Figure 2: The panes of the MPE-IDE

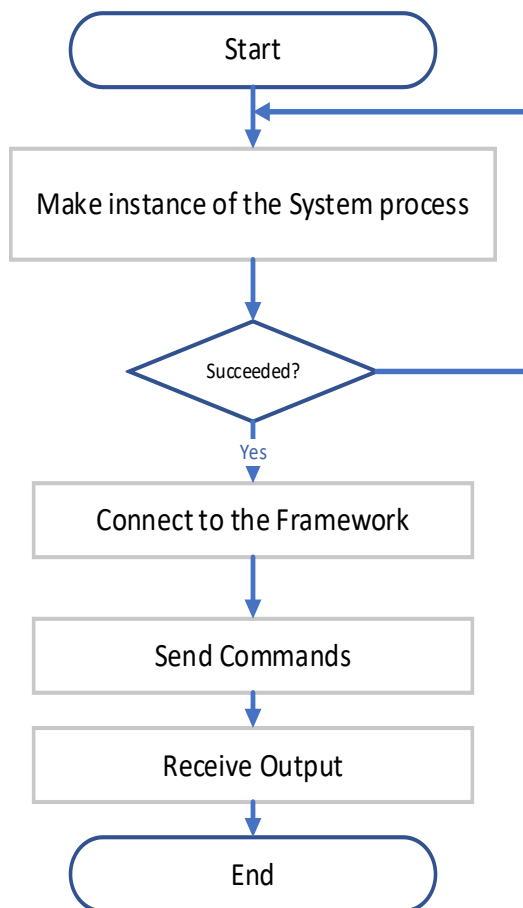


Figure 3: The connection method to the .Net Framework

3.2. Integration method with the .Net framework

The challenging part of this work is how to connect with the .Net framework in order to use the class libraries and language interface without

using the Microsoft platform cross-platform version of .NET that is maintained by Microsoft and the .NET community. Thus, we proposed and implemented a method that connects externally to the .Net core and uses its class library and management facilities. Figure 3 shows the steps followed to connect to the .Net framework from the IDE environment.

The method works as follows: First, it makes an instance of one of the operating system tools such as PowerShell or command line interface that allow communication with all system parts. This process emulates the operating system and gets all the power and privileges necessary for sending commands as an input and receiving the output results. After a successful connection, it starts sending commands and then receiving the output of the process.

3.3. How C-MPE works?

The method works as follows: First, it makes an instance of one of the operating system tools such as PowerShell or command-line interface that allow communication with all system parts. This process emulates the operating system and gets all the power and privileges necessary for sending commands as input and receiving the output results. After a successful connection, it starts sending commands and then receiving the output of the process.

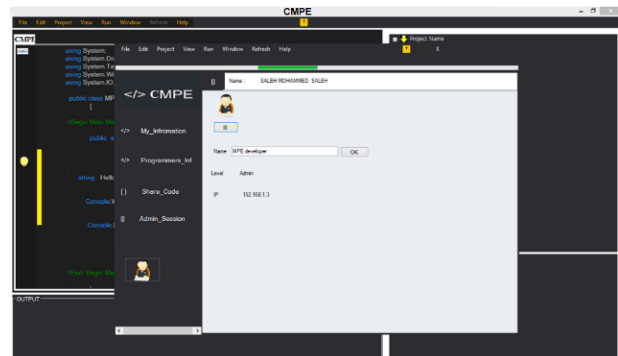


Figure 4: Adding programmer to the project

Once the project leader assigns the tasks to the programmers, the project will be cloned to the programmer machine. Then each programmer can work on the modules, classes, or methods assigned to him, Fig 5. If the programmer made any changes, he has to commit the changes and synchronize with the rest of the team in order to get the updated versions of the project. If any of the programmers worked offline, synchronization will start as soon as he connects to the project leader machine.

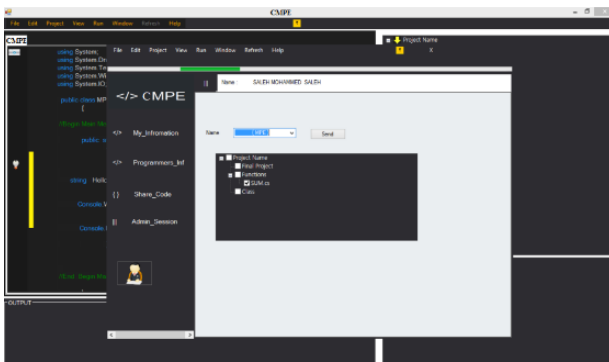


Figure 5: Assigning tasks to the programmer

4. Technical Challenges and limitations

The C-MPE is tested under different types of projects and worked fine, however it faces some technical and performance challenges. Connecting to the operating system instance results in a delay compared to the native connection found in the .Net framework. Moreover, the C-MPE is limited to working only on private networks. The current version of C-MPE supports only one C# language of the .Net framework and restricts the programmers to work only on the modules assigned to them.

5. Conclusion and Future Work

Collaborative software development gains popularity due to the enchantment in the network and communication technology. A group of developers can share the development of the

project even if they are located in different locations. Based on that, we have developed a collaborative development environment for the .Net framework that helps developers work together on a software product. C-MPE is constituted of two major parts. The integrated development environment and in addition to the core method that makes use of the .Net framework core services to build and execute the programs. The preliminary results show that the C-MPE can be used to develop and share applications between teams over a private network. In future work, C-MPE will be enhanced to support multiples languages of the .Net framework. In addition to the enhancement in the speed of the synchronization process. Moving C-MPE to use the Internet-based .Net framework is another forward step that will be taken into consideration in the upcoming enhancements.

6. References

- [1] Schümmer, Jan. Support for Distributed Teams in eXtreme Programming. 2001, Addison Wesley.
- [2] Calefato, Fabio & Lanubile, Filippo. (2011). Practice: Collaborative Development Environments.0.1002/9781118135105.ch15.
- [3] GitHub , where the world builds software [https:// www.Github.com](https://www.Github.com).
- [4] SourceForge , download, develop, and publish free open source <https://www.sourceforge.net/>
- [5] AWS Cloud9 Amazon Web Services <https://aws.amazon.com/cloud9/>
- [6] Eclipse, Enabling Open Innovation and collaboration <https://www.eclipse.org>
- [7] Visual Studio life share <https://visualstudio.microsoft.com/services/live-share/>

- [8] Space, The integrated Teamwork, <https://www.jetbrains.com/space/>
- [9] .Net Framework and Architecture <https://www.c-sharpcorner.com/uploadfile/puranindia/net-framework-and-architecture/>
- [10] H. B. Salameh and C. Jeffery, "Collaborative and social development environments: a literature review", International Journal Computer Applications in Technology, Vol. 49, No. 2, (2014), pp. 89.
- [11] Cheng, L-T., Hupfer, S., Ross, S., Patterson, J.: Jazzing up Eclipse with Collaborative Tools. In Proceedings of the OOPSLA Eclipse Technology eXchange Workshop, pp. 45 - 49, Anaheim, CA, October 2003.
- [12] Ho, C., Raha, S., Gehringer, E., Williams, L., Sangam: A Distributed Pair Programming Plug-in for Eclipse, Eclipse Technology Exchange (Workshop) at the Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA) 2004.
- [13] Langton, J., Hickey, T., Alterman, R.: Integrating tools and resources: a case study in building educational groupware for collaborative programming. Journal of Computing Sciences in Colleges, 2004. 19(5): p. 140 – 153.
- [14] Hupfer, S., Cheng, L.T., Ross, S, and Patterson, J. Introducing Collaboration into an Application Development Environment. In Proceedings of the ACM Conference on Computer Supported Cooperative Work, 2004, pp. 21-24.
- [15] Bani-Salameh H., Jeffery C., Al-Sharif Z., Abu Doush I. (2008) , Integrating Collaborative Program Development and Debugging within a Virtual Environment.
- [16] Aditya Kurniawan, Aditya Kurniawan, Christine Soesanto, Joe Erik Carla Wijaya, CodeR: Real-time Code Editor Application for Collaborative Programming, Procedia Computer Science, Volume 59,2015,Pages 510-519,
- [17] Wang, C., Zhang, W., Zhao, H. et al. Towards a fictional collective programming scenario: an approach based on the EIF loop. (2020), pp. 3671–3710.