OCD OF PROJECT 4

Remote Package Dependency Analysis

Instructor: Jim Fawcett

Xiao Chen 520578212

Contents

1. Exclusive Summary

The Document mainly discuss the operational concept for tool of Remote package dependency analysis. In the project, the tool for Remote package dependency analysis is based on the former project – type-based dependency analysis. By using WCF(Windows Communication Foundation) and WPF(Windows Presentation Foundation), the project 3 can be easily extended into a well organized UI looking and web-based remote tool for remote dependency analysis.

This project include the main components as follow:

1. Lexical Scanner – Used to get single token to be analyzed from target files
2. Semi Expression – Used to process token which been collected from Lexical scanner and generate semi expression which can be used to analyze exact type of the files and dependency of files.
3. Type Table generator – Using Semi Expression or token generated before to get exact type of files(include not limited to: “Class” ,”Enum”, “Struct”, “Delegate”, “Alias”) and save the result to be used by dependency analyze.
4. Dependency Analysis – Using Type Table generated before, by comparing struct inside files to get dependency relationship between files.
5. Strong Component – Using exist dependency between files, by using Tarjan algorithm we can find a Strong component between files.(*A strong component is the largest set of files that are all mutually dependent. That is, all the files whcih can be reached from any other file in the set by following direct or transitive dependency links. The term 'Strong Component' comes from the theory of directed graphs.*)

6. Main Client – By using WPF of Visual Studio, it is easily to generate a User Interface of the exact program.
7. Server – As we are going to analyze the files remotely it's important to generate connection between local and server. By using WCF of Visual Studio, we can generate connection between local client and remote server.
8. Massage Passing Communication Service – by using WCF to generate connection between local and server you have to define how does Data passing and what will it be to connect.

The uses of the remote dependency analysis are the source code files that saved on server. By config the server address and path, remote dependency analysis can easily visit server files and do specified actions such as getting type table, getting dependency relationship and getting strong component from server. This remote dependency analysis can be used in a lot of fields, includes finding dependency between files, getting to know the type of files, doing further analyses on the AST to construct code metrics. And some components like Lexical Scanner can be easily adapted into tools to analyze article, or to find characters inside a file.

Below are some of the critical issues that might happened in this project, and solution will be discussed in further critical issues section.

1. How to deal with an integral word, like "project", "int", "double". The scanner should output the words as an integration rather than divided letters.
2. How to deal with hyphen in char array? For instance, [a-d] is equal to [abcd], and what if there are a huge number of characters in the array?
3. Only use one getTok() may cause incorrect character sequence.

4. How to ignore the two semicolons within parentheses in a for(;;) expression and the terminated single character tokens in print line.
5. How to avoid generating dependency relationship when you meet 2 different files with same namespace name.
6. How to tell apart 2 different classes with the same name in 2 files.
7. How to avoid data transmission failure or information blocking due to excessive data volume.
8. Number of thread that can spawned at startup.

2. Introduction

Application obligation

The main responsibility of this application is to provide:

1. Provide an efficient way to find files and directories from a specified path, this path should have capability to be easily changed.
2. Get Token and semiexpression from a set of files.
3. Get typetable and make dependency analysis and strong component analysis in remote server.
4. In project 4, we will provide an efficient GUI for users to use. It carries the function we made in project 3.
5. Access path config from Environment configuration.
6. Provide ability to analyze code from remote files.

Concept and key architectural idea

The system will use the .Net framework, and Windows Communication Framework (WCF) tools are implemented in C# to establish communication between local server and remote server, and Windows Presentation Foundation (WPF) as a client process. Implement the GUI, Visual Studio 2017. The key modules are Build Server, Repository, Test Harness and Client Process. The functionality of each module is implemented in a separate package, which is described in detail in later chapters.

3. Uses and Users

The users of tool of remote dependency analysis can be schools, companies, students and professors. By changing some part of earlier function this dependency analysis can be easily adapt into tools for checking code, or even checking article content of an article.

Student and Professor

A professor may have a lot of students in his class, it's would be a great load for professor to grade homework from student. Remote dependency analysis tool can easily be adapted into a tool with capability of get code from remote folder. Student can upload their homework to remote server and Professor can grade them easier. With the type analysis we implementing in project 2, remote dependency analysis tool can be adapted into a tool that get to know the function and structure in student's homework, and it can find oversized function or public variable in their homework. This tool can also help student get to learn the structure and function of a program easily by getting function and dependency relationship from target program solution and this could let them easily understand how to develop a large project.

With the user friendly UI we designed, it can be used by everyone easily.

Companies

For a company, the project could be so large that new employee cannot handle and understand it easily. By analyze type and dependency relationship within project of a company, it could help new one get to know project easily. it can help engineers in company know what their colleague add or adjust to exist project and get to know the changes of relationship when these changes happened.

Developers

Although the student is the developer for this project, developers at companies would use the remote dependency analysis tool to better know their project. With remote server it is also easy for them to manage their files.

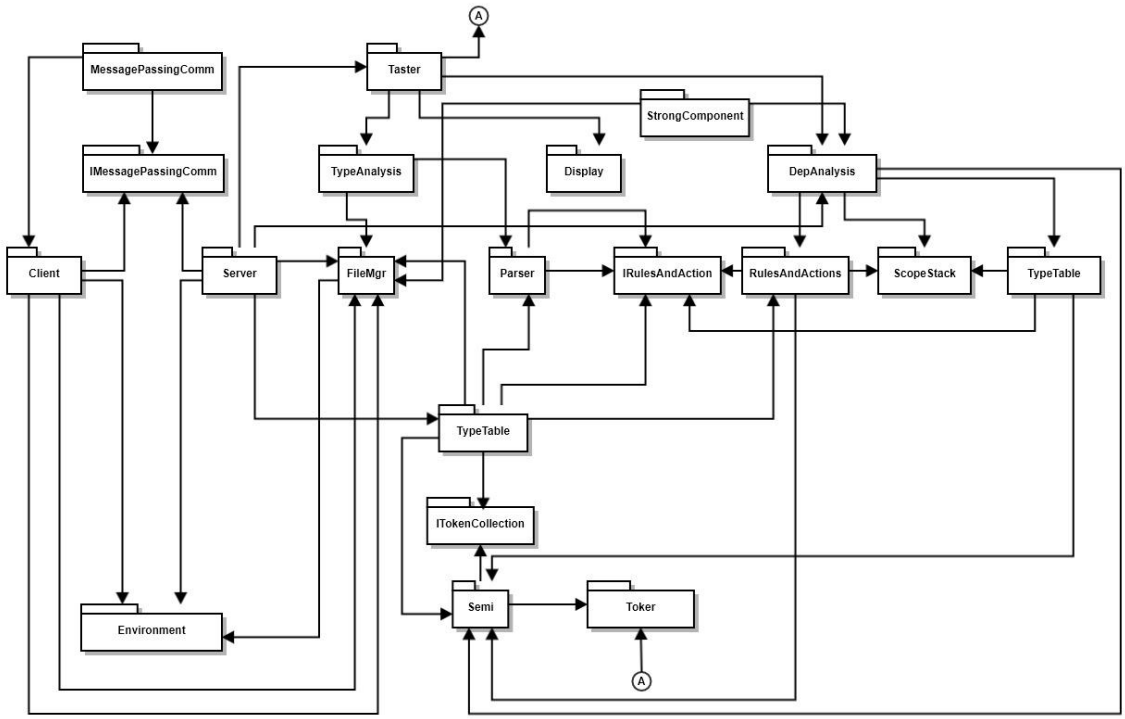
Quality Assurance Engineers

Quality assurance engineers are vital to the health of any set of code. They may use Dependency analysis tool to analyze the relationship between files and this will help them detect and locate bug easily. The GUI would need to provide them with vital information to the health of their team's code. For this project, the student is both the developer and the quality assurance engineer.

4. Partitions

This Project remote dependency analysis contains all of package from earlier project. Overall, it contains over 17 packages and all of them have different ability but all of them are useful to get analyze dependency relationship between specific files, analyze it from remote server, and do specified tests.

The figure below is the package diagram of this project.



Toker

This package mainly responsible for extracting words, called tokens, from a stream of characters and tokenize every single word and symbol. In this package, there is a Toker class which uses TokenState class and derived classes to implements the State Pattern for collecting the following token types: alphanumeric tokens, punctuator tokens, one and two character tokens, single and multiple line comments and quoted strings.

SemiExp

By calling getTok() from toker collecting tokens from files, SemiExp class will use some terminated rules that can control the retrieve processing by defining terminating characters: semicolon, open brace, closed brace, and newline with ‘using’ at the first character on that line.

ITokenCollextion

Itokencollection is the interface that hold function and properties of semi expression. This package did not actually implement any actual function. Package SemiExp will implement all function inside ITokenCollection. If needed, other package will instantiate it.

IRulesAndAction

IRulesAndAction is the package that hold rules of how we tell apart different type of structures like function, class, delegate, etc. The package defines two interface IRule and IAction which hold contract for rules and actions for these rules. And it has two abstract base class for these rules and related actions.

RulesAndAction

RulesAndAction Package contains all of Application specific code required for code analysis. It defines derived class of ARules to detect specific structures. It can detect NameSpace, Class, Delegate, Alias and Functions. By using these rules you can easily tell apart structures you need.

ScopStack

ScopeStack provides, via the class ScopeStack, the facilities to track position in code scope by pushing and popping type, name, and place. The type might be a namespace, class, struct, function, or control. The name is the namespace of the instance of that type, and place is the line number in a package being analyzed.

Paser

This package Paser, by using rules defined before, can get all the different structure from .cs files, and, by using ScopStack, it can finally generate a integrated code analysis form.

TypeTable

This package TypeTable, by using rules defined in other packages and calling semiexp.get(), will get the type information from specific files and by defining a dictionary in this package, we will finally get a whole typetable of these files and waiting for further process.

DepAnalysis

This package DepAnalysis, by using typrtable generated by Type Table package, through calling semiexp.get(), compare every semi expression from files we want with the structure we generated inside typetable. We will find the dependency relationship with in these files. The dependency relationship include but not limited to inherit, reference, instantiation etc. By getting the dependency relationship within these files, we can do some further analysis like: Strong Component later.

Display

Display manages static public properties used to control what is displayed and provides static helper functions to send information to MainWindow and Console.

StrongComponent

This package StrongComponent, by getting dependency table(a dictionary defined in DepAnalysis), use tarjan algorithm. It will walk through the dependency table and find all the related Strong Component.

FileMgr

FileMgr provides the needed function for typetable, depanalysis, strongconponent, to get target file from specified path. FileMgr also provide function for GUI and server to get files from target path.

Environment

This package Environment provides needed configurations for server and client to get path and address to get access to target directory.

IMPCommService

This package provide service interface for MessagePassingComm. In this package, it defines all the contract needed by WCF and for further use, these contracts are needed.

MPCommService

The Comm package implements asynchronous message passing communication using the Windows Communication Foundation Framework (WCF), which provides a well-engineered set of communication functionalities wrapping sockets and windows IPC. In this package, we provide function to generate connection between client and remote server. This will create a communicate channel between client and remote server, and pass message by thread. This message passing will not pass files, instead it will pass path of the files selected.

BlockingQueue

This package implements a generic blocking queue and demonstrates communication between two threads using an instance of the queue. If the queue is empty when a reader attempts to deQ an item the reader will block until the writing thread enQs an item. Thus waiting is efficient.

NavigatorServer

This package defines a single NavigatorServer class that returns file and directory information about its rootDirectory subtree. It uses a message dispatcher that handles processing of all incoming and outgoing messages. By defining different message dispatcher you can generate different reply for different request. For example you can process button click event of remote files and dirs. Processing of generating Typetable, dependency table and strongComponent will be defined in this class.

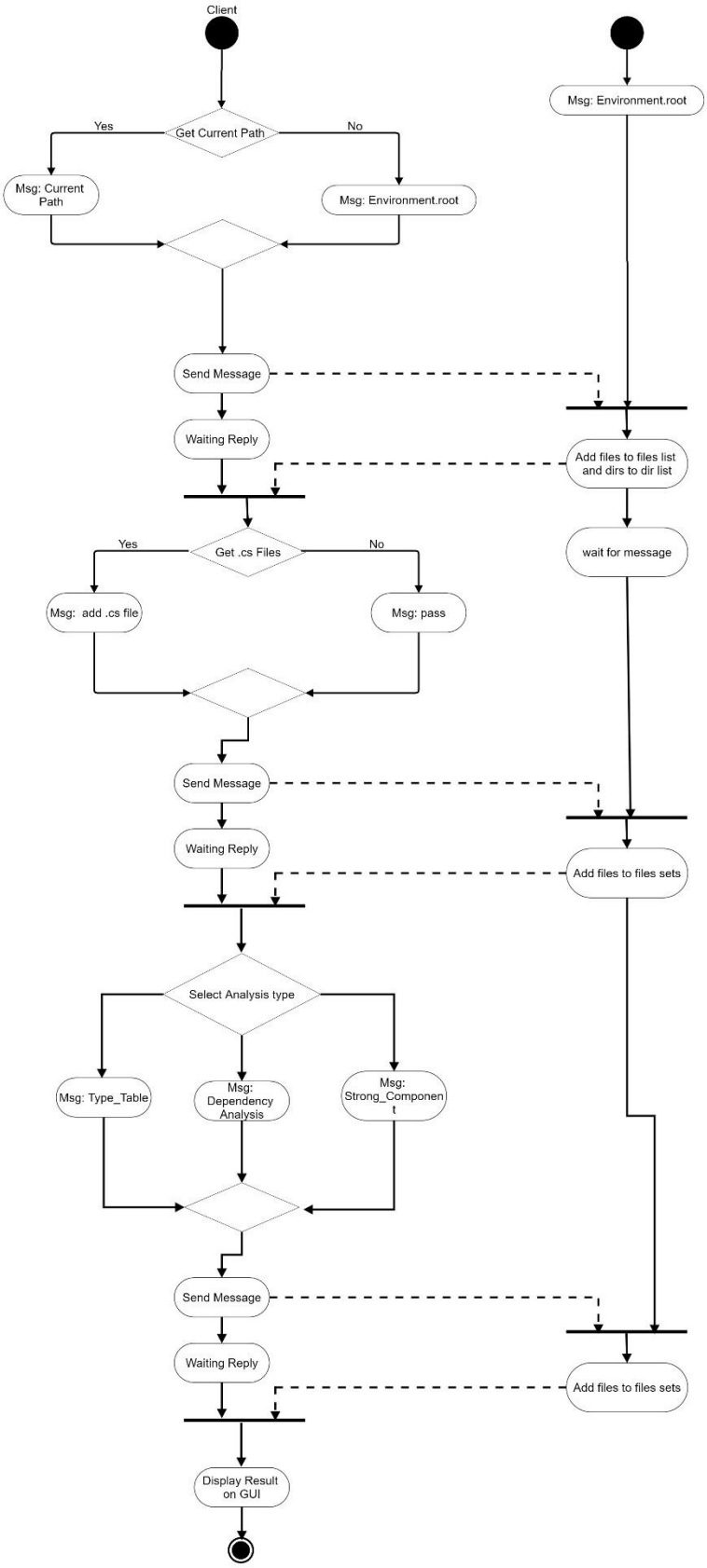
MainWindow

This package defines WPF application processing by the client. The client displays a local files and dirs view, a remote file and dirs. View and a files sets view and a view which can show Dependency and StrongComponent result. The interaction with user interface also defined in this package, which means that you can define all of button reactions and content of listbox and content of textbox in this package.

5. Application Activities

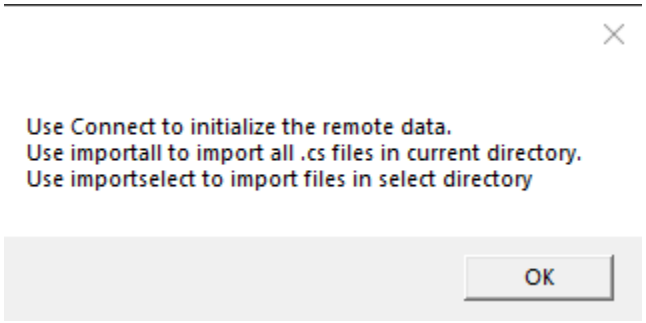
Figure 1 below briefly shows the activities performed in the Remote Dependency Analysis when a command was sent from client.

1. Client get a command to connect with sever
2. Send a request to server to get needed path
3. Server received the request send back needed path as a reply
4. Get files and directories path from server
5. Send a request to server to get .cs files from files sets
6. Server received the request send back all of .cs files' path as a reply
7. Get .cs files pending for analyze
8. Client send message send message to server to tell sever what kind of analysis should be done.
9. Server received the message and start to analyze target files
10. Server send back result of analysis to client
11. Client show result on GUI

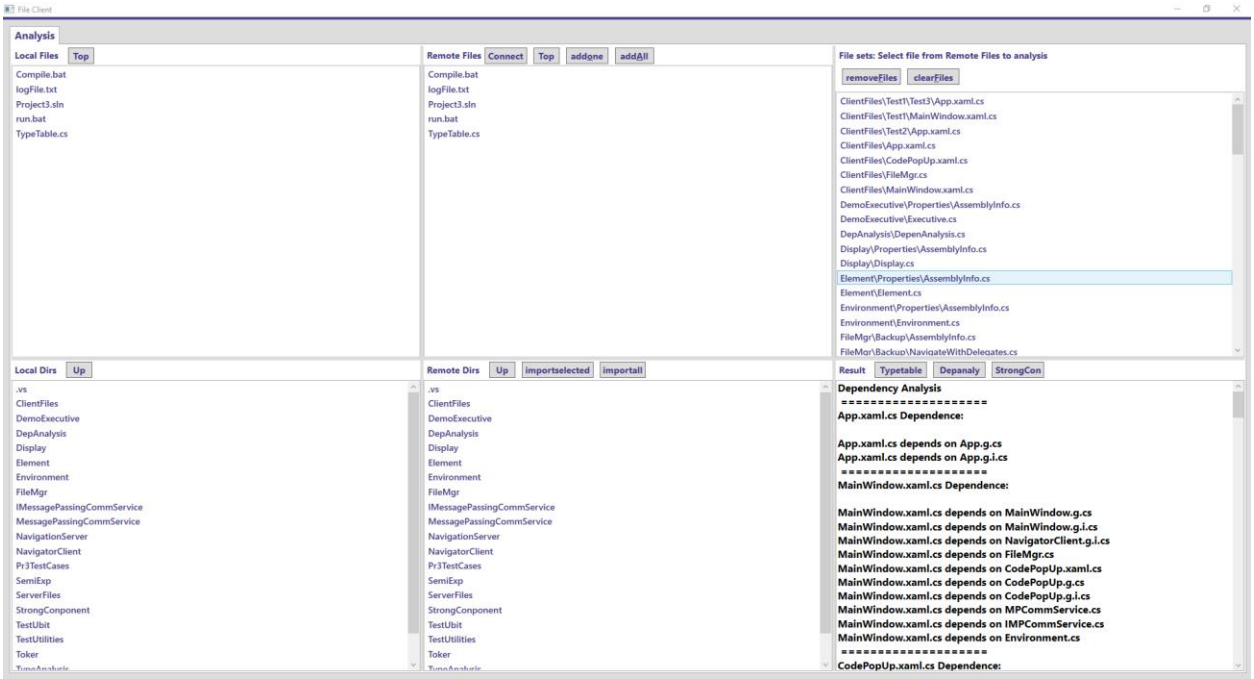

Client
Yes
Get Current Path
No
Msg: Current Path
Msg: Environment.root
Send Message
Waiting Reply
Yes
Get .cs Files
No
Msg: add .cs file
Msg: pass
Send Message
Waiting Reply
Select Analysis type
Msg: Type_Table
Msg: Dependency Analysis
Msg: Strong_Component
Send Message
Waiting Reply
Display Result on GUI
Msg: Environment.root
Add files to files list and dirs to dir list
wait for message
Add files to files sets
Add files to files sets

6. User Interface

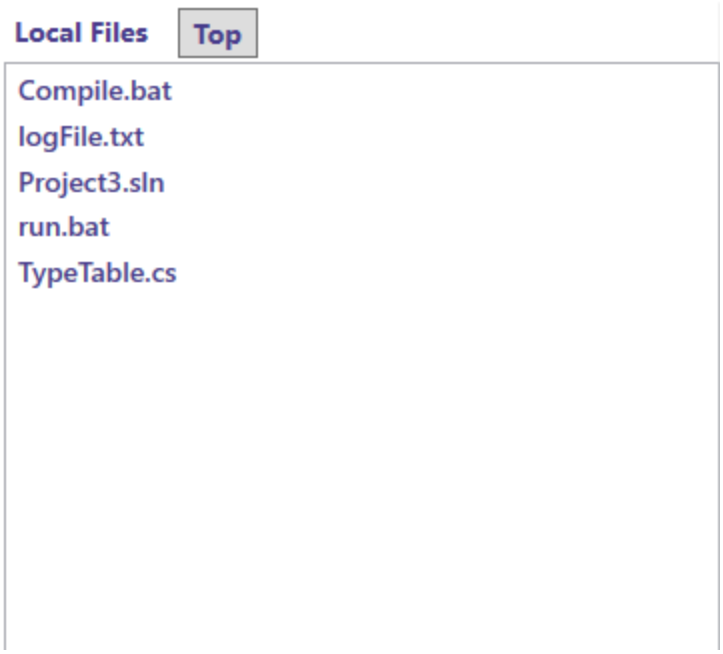
In this project we provide a user's friendly UI for user to use. UI have 2 main components, first part is a message box which contains simple instruction regarding how to use this remote dependency analysis tool. Second part is a main UI that support all function we made in project 3 and the UI support select files from target folder, select all files from target folder and import all files from a directory.



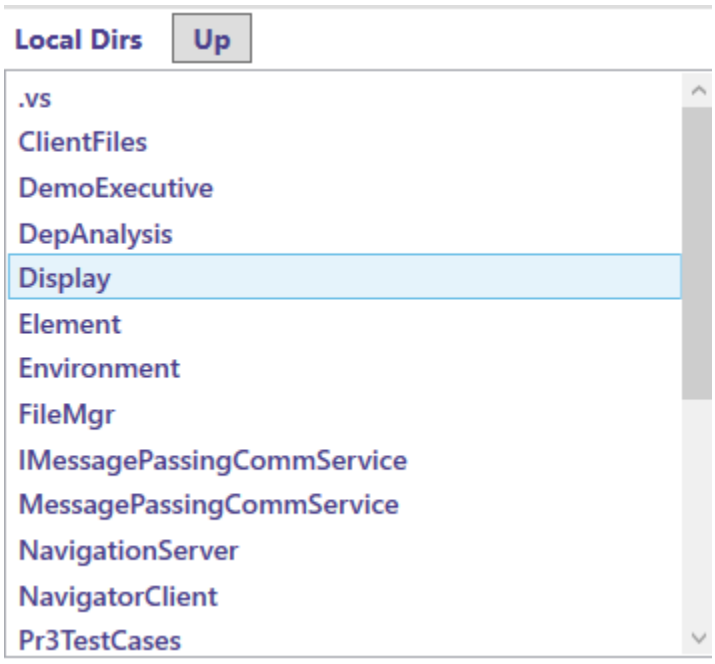
The message box above is the simple instruction regarding how to use this remote dependency analysis tool.



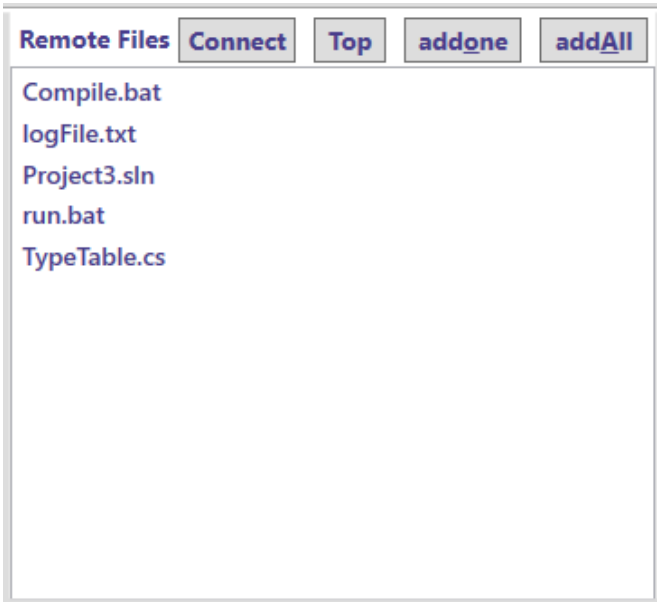
The picture above shows overall UI of this remote dependency tool.



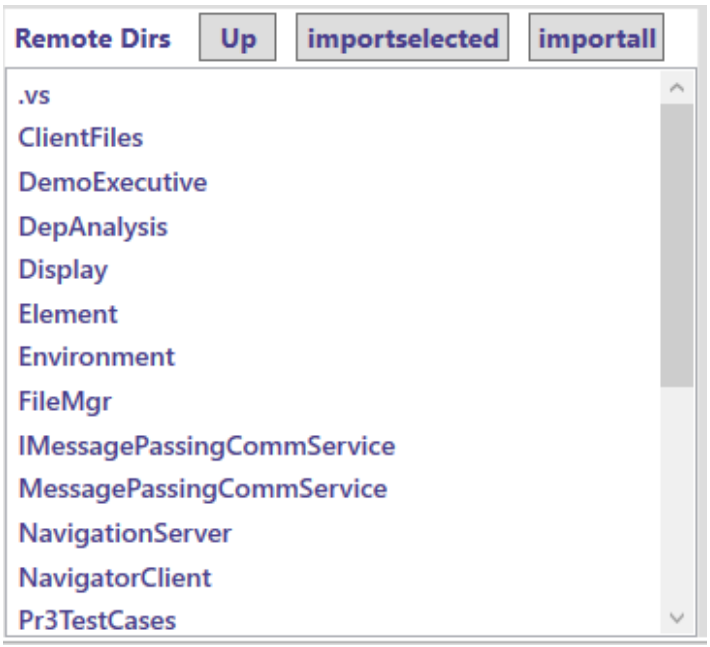
The picture above shows the part of the remote dependency tool. This will get all files from current directory. As soon as you enter another directory, this list box will be updated. Top button support a function that you can get back to the root path of local directory. Double click files in list box will get a pop up window shows file content inside.



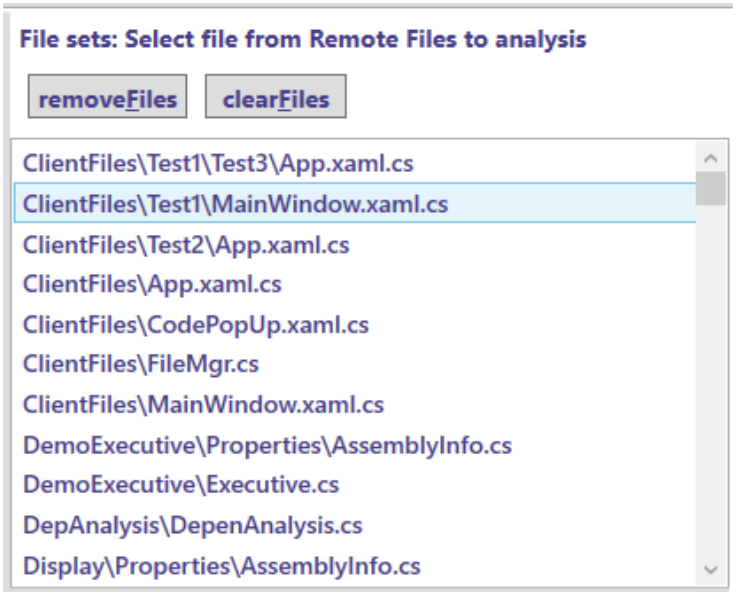
The picture above shows the part of the remote dependency tool. This will get all directories from current path. Double click directory in the list box will move into this folder. Up button support function to move back to parent directory of current directory.



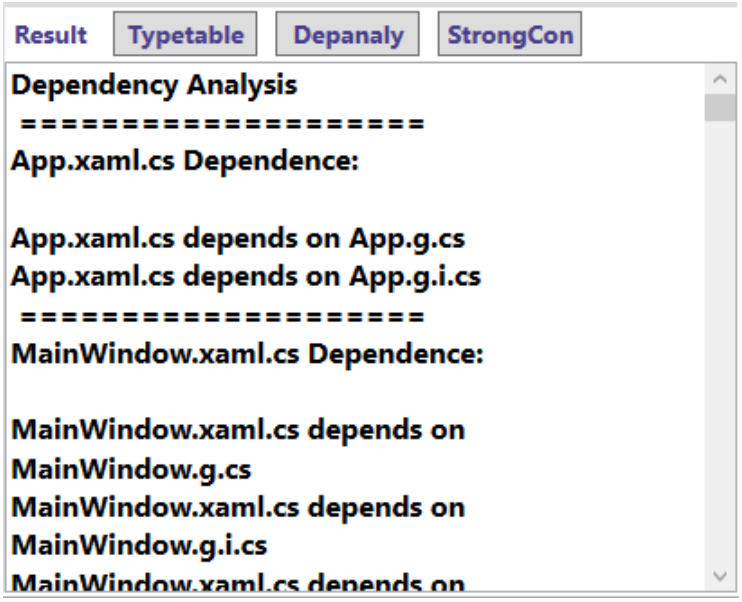
The picture above shows the part of the remote dependency tool. This will get all files from remote server root directory. As soon as you enter another directory, this list box will be updated. Connect button support a function that initialize the Remote Files listbox and Remote Dirs listbox. Top button support a function that you can get back to the root path of remote files directory. Addone Button support a function that you can add selected .cs file into file sets in order to be processed. Addall Button support a function that you can add all .cs file in listbox into file sets in order to be processed. Double click files in list box will get a pop up window shows file content inside.



The picture above shows the part of the remote dependency tool. This will get all directories from remote serve. Double click directory in the list box will move into this folder. Up button supports function to move back to parent directory of current directory. Importselected button supports a function to import all files in selected directory into file sets list box in order to be processed. Import all supports a function to import every single .cs files in current directory and all of it's subdirectories.



The picture above shows the part of the remote dependency tool. After import files from remote server, file sets listbox will get all files pending for analysis. moveFiles button supports a function to remove single file from file sets and clearFiles button supports a function to remove all files in file sets listbox.



The picture above shows the part of the remote dependency tool. This will analyze the file inside File sets. Typetable button supports a function to get typetable from files of file sets list box and show type table in result text box. Depanaly button supports a function to get dependency relationship from files of file sets. StrongCon button supports a function to get strong component from the files of file sets.

7. Critical Issues

Performance:

1. In lexical scanner, how to deal with an integral word, like “project”, “int”, “double”. The scanner should output the words as an integration rather than divided letters.

   Solution: the longest principle can solve this question. It is a kind of advanced search identification according to lexical definition. For instance, there could be some special token like”++”,”+=”, when scanner meet a” +”, it will continue to search for next character to see if it is the expected one. After checking for the next one, it will come to decide that what will it finally generate.
2. How to ignore the two semicolons within parentheses in a for(;;) expression and the terminated single character tokens in print line.

   Solution: In semiexpression we are using some kind of way to get semi like token, which is peek token one by one ubtil we come with the terminate marks. Which means that we can trace back the token that we have peeked before. When we meet “;”, it should track back to see if there is”(” in earlier tokens. If there was one, the terminate ,ark should be “)”, rather than “;”.
3. How to avoid generating dependency relationship when you meet 2 different files with same namespace name.

   Solution: When you analyze dependency relationship, not only you have to compare if there is same keyword inside a semiexpression(in dependency analysis, I use semi exp to analyze), you have to analyze the file name and namespace of the file to make sure that same namespace name will not be regarded as a dependency relationship.
4. How to tell apart 2 different classes with the same name in 2 files.

Solution: Same as the issue before. When you analyze dependency relationship, it is important to record file name and namespace name to tell apart the special situation.

5. How to avoid data transmission failure or information blocking due to excessive data volume.
   Solution: when you analyze a large set of files, when you are expecting result from servers, it might be a block of data because of excessive data volume. When you come up with the situation, it is a good solution to split a huge message into several small message can send it back to client or server.

Usability:

6. If you meet the file with extend name being not ".cs" and you click it in client what will it be
   Solution: When you select a file and send it path back to server, server should judge if it is ".cs" file. If it is, it can be analyzed
7. When files using aliases, the Semi can't directly get the type and type name. In this case, how can we get correct dependency relationship in these files.
   Solution: when you find "using" in a semi expression, you need to check if there is ":" in semi expression too. If there is then you the file have used alias.
8. When you are doing dependency analysis what should you use, Token or Semi Expression?
   Solution: I prefer to use Semi Expression. In semi package there is function called "contains", which can easily judge whether there was target string inside this semi expression. And semi can get whole paragraph of cade which can provide ability to trace back to earlier token. This is what token can not be done.