Journal of Computational Intelligence in Bioinformatics
ISSN 0973-385X Volume 4, Number 3 (2011) pp. 267-272
© Research India Publications
http://www.ripublication.com/jcib.htm

Design of a Reusable Software Component for Finding Restriction Enzyme Site

Subrata Sinha, Partha Pratim Kalita and Robin Das

Centre for Bioinformatics Studies, Dibrugarh University,
Dibrugarh-786004, Assam, India
E-mail: subratasinha@indiatimes.com

Abstract

Bioinformatics software development is a very fast and versatile sector. There are many utility software already exists in the market but for faster development of such utility software we need reusable components. Component reusability is about building a library of frequently used component, thus allowing new programs to be assembled quickly from existing components. It has produced greater schedule and effect savings than any other practice. In this paper we have tried to design such a component which can find the restriction enzyme cut site in a given DNA sequence and also it can provide sequence of any given restriction enzyme

Keywords: Reusable Software Component, Restriction Enzyme, ActiveX Dll, Bioinformatics COM.

Introduction

A restriction enzyme (or restriction endonuclease) is an enzyme that cuts double-stranded or single stranded DNA at specific recognition nucleotide sequences known as restriction sites.[1][2][3] To cut the DNA, a restriction enzyme makes two incisions, once through each sugar-phosphate backbone (i.e. each strand) of the DNA double helix. Restriction enzymes recognize a specific sequence of nucleotides[2] and produce a double-stranded cut in the DNA. While recognition sequences vary between 4 and 8 nucleotides, many of them are palindromic, which correspond to nitrogenous base sequences that read the same backwards and forwards. Over 3000 restriction enzymes have been studied in detail, and more than 600 of these are available commercially[4] and are routinely used for DNA modification and manipulation in laboratories.[5]

Some time it becomes very important to know about the cut sites of any given

Restriction Enzyme in a given DNA sequence. If a developer tries to have that feature in his software for windows platform than he will have to write the code from scratch. So it is very important to develop a component which can solve the purpose of the developer. We have designed such a reusable component which can accept the DNA sequence and Restriction Enzyme name and return the cut position as an array of integers. Also the component is capable of returning a sequence of a given Restriction Enzyme. Presently it has the records of 100 Restriction Enzymes.

The essence of COM is a language-neutral way of implementing objects that can be used in environments different from the one they were created in, even across machine boundaries. For well-authored components, COM allows reuse of objects with no knowledge of their internal implementation, as it forces component implementers to provide well-defined interfaces that are separate from the implementation.[6]

Components provide reusable code in the form of objects. An application that uses a component's code, by creating objects and calling their properties and methods, is referred to as a client.[7]

ActiveX is a framework for defining reusable software components (known as controls) that perform a particular function or a set of functions in Microsoft Windows in a way that is independent of the programming language used to implement them.[8]

ActiveX Component can be developed under ActiveX framework. An ActiveX component in general term that encompasses ActiveX DLL, ActiveX EXE, and ActiveX controls in the context of Microsoft Visual Basic. An ActiveX DLL component is an in process server. The DLL is loaded in the same address space as the executable that calls the server and runs on the same thread as the client. At any given moment, however, either the client application or the DLL is running. The benefit of DLLs is that they are faster because, in effect, they become part of application that uses them.[9]

Materials and Methods

Process of Creating DLL for Microsoft Windows Platform Using VB6.0

First ActiveX DLL project is selected from new project dialog. By default class module Class1 is added. We have renamed it to reclass. Then we have constructed ReSeq data structure, a two dimensional array in the class initialize() event, which contains 100 Restriction Enzymes and their sequence.

After the Preparation of ReSeq we have added three functions findcutsites, showAllEnzymes, and showSeq to our class with the following function signatures.

```
Public Function findre(x As String, y As String) As Integer()
Public Function showAllEnzymes() As String()
Public Function showSeq(x As String) As String
```

Procedure to add functions to class module: Add procedure selected from Tool menu of the IDE. Specify the name of the procedure, Type and Scope. Click OK

The argument and return type is specified in the functions as shown above. And we have defined each functions.The code inside the functions are given in the next section.

After function definition in the reclass we have selected Make refinder.dll from File menu of IDE.

Algorithm

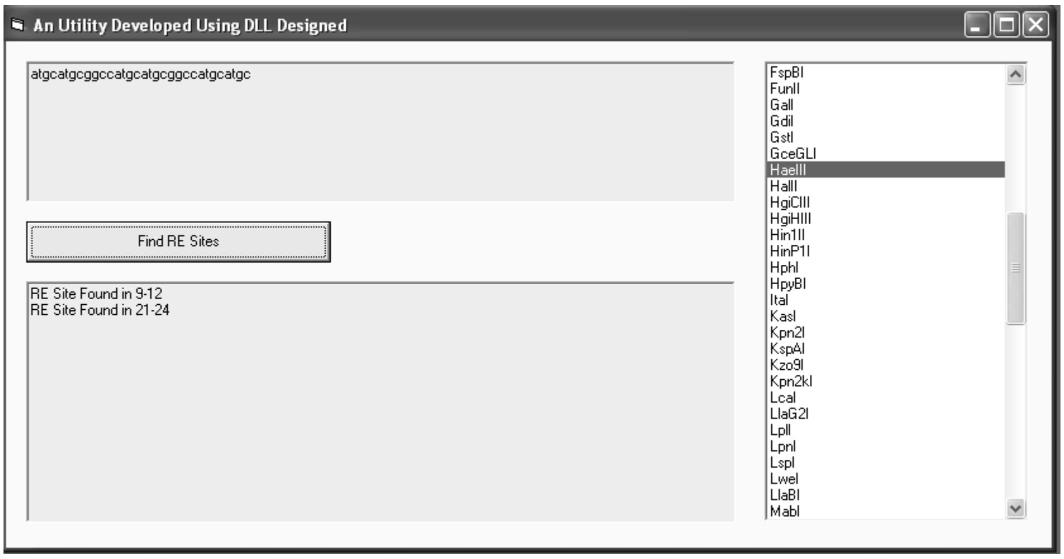
```
Note: ReSeq is a double dimensional array containing
Enzyme name and its sequence
Algorithm 1: findCutSites (dnaseq, rename) - This
algorithm accept two string dnaseq and rename.It returns
RE cut sites as an array of integers
1. [Initialize] set i=1, j=1, g(0)=0 and k=0
2. set reseq=showSeq(rename)
4. Repeat for i=1 to length (dna)
5. If SUBSTRING (dna, i, 1) = SUBSTRING (reseq, j, 1)
 Set i=i+1
 Set j=j+1
6. If j=length (reseq) then
 Set g(k)=i-j+1
 [Increment size of array g() by 1)]
          Set k=k+1
          Set j=j+1
   [End if]
7.        Else
   Set i=i+1
 Set j=j+1
 [End of step 5 if]
 [End of step 4 loop]
8. Return g()


Algorithm 2: (Showing all enzyme) This algorithm accept
RE Sequence and return RE name
 [Declare] x(100) as string
Repeat for i=0 to 100
  Set x[i]= ReSeq(I, 0)
[End of loop]
Return x
Algorithm 3: showSeq(enz)- This algorithm accept RE name
and return RE sequence

Repeat for i=0 to 100
If enz=Reseq(i, 0)then
Set seq=ReSeq(i, 1)
[End if]
```

```
[End for loop]
 3. Return enz
```

Result

The result is RE finder DLL which has a class RE finder which has in return three function; findCutSites() which accepts two strings DNA sequence and RE name .Then it calls showSeq() function which returns the sequence of the Restriction Enzyme. When the function findCutSites() get the RE recognition site which is a short string . It start finding the cut-sites in the DNA sequence and places all cut positions in a dynamic array, on completion of search, it returns the array. The another function is showAllEnzyme () which is a simple function, when called return an array of all enzyme name present in the data structure and another important function showSeq() accepts RE Name and search sequence for it in the data structure ReSeq.

Application using refinder. dll



```
atgcatgcggccatgcatgcggccatgcatgc
```

```
RE Site Found in 9-12
RE Site Found in 21-24
```

Schematic Diagram of Working of Refinder.Dll

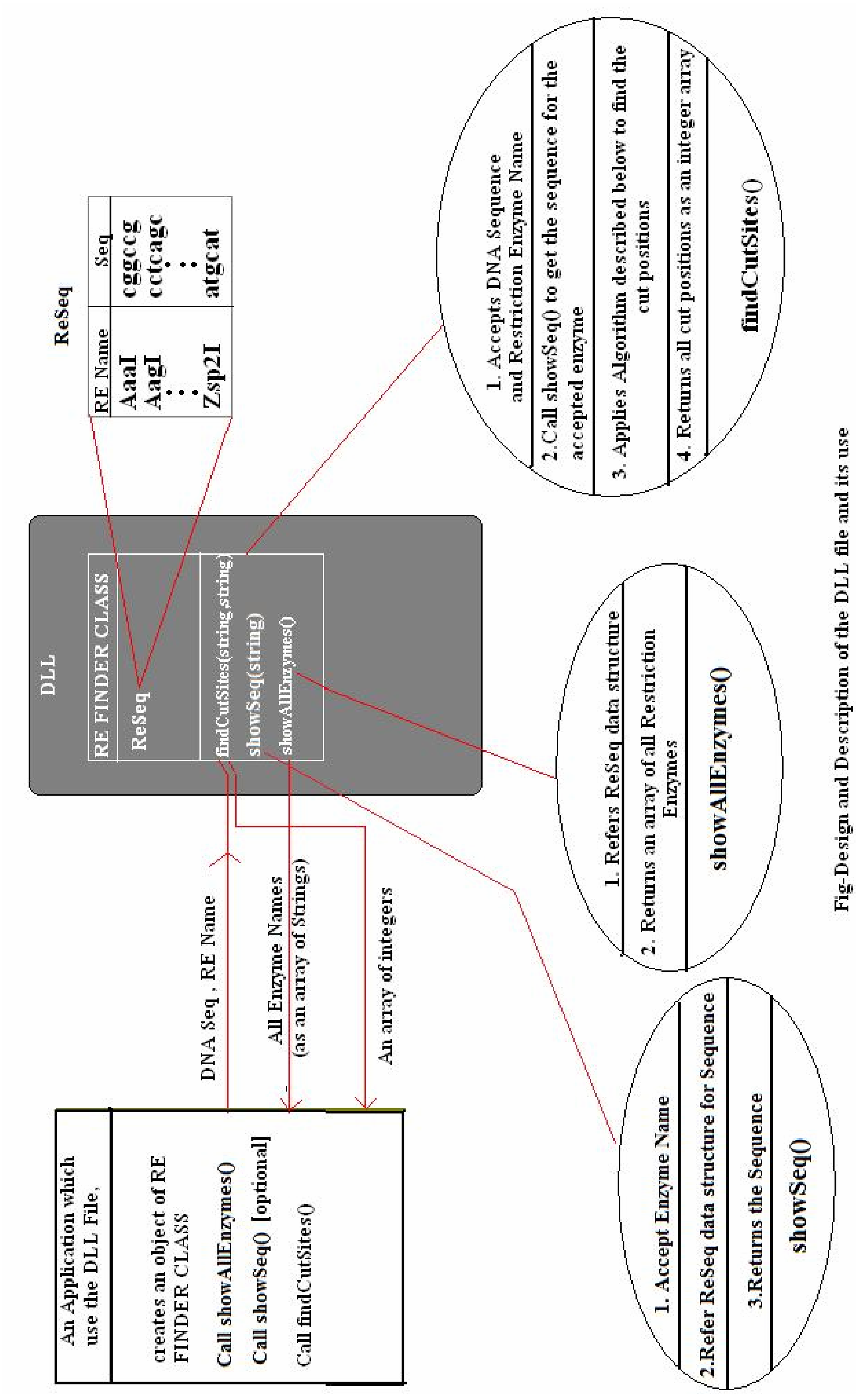

ReSeq
RE Name
Seq
AaaI
cggccg
AagI
cctcagc
Zsp2I
atgcat
DLL
RE FINDER CLASS
ReSeq
findCutSites(string,string)
showSeq(string)
showAllEnzymes()
1. Accepts DNA Sequence and Restriction Enzyme Name
2.Call showSeq() to get the sequence for the accepted enzyme
3. Applies Algorithm described below to find the cut positions
4. Returns all cut positions as an integer array
findCutSites()
1. Refers ReSeq data structure
2. Returns an array of all Restriction Enzymes
showAllEnzymes()
1. Accept Enzyme Name
2.Refer ReSeq data structure for Sequence
3.Returns the Sequence
showSeq()
DNA Seq , RE Name
All Enzyme Names (as an array of Strings)
An array of integers
An Application which use the DLL File,
creates an object of RE FINDER CLASS
Call showAllEnzymes()
Call showSeq() [optional]
Call findCutSites()

Fig-Design and Description of the DLL file and its use
(No existing diagramatic notation is used to describe)

References

[1] Roberts RJ (November 1976). "Restriction endonucleases". CRC Crit. Rev. Biochem. 4 (2): 123–64. doi:10.3109/10409237609105456. PMID 795607.

[2] Kessler C, Manta V (August 1990). "Specificity of restriction endonucleases and DNA modification methyltransferases a review (Edition 3)". Gene 92 (1-2): 1–248. doi:10.1016/0378-1119(90)90486-B. PMID 2172084.

[3] Pingoud A, Alves J, Geiger R (1993). "Chapter 8: Restriction Enzymes". in Burrell, Michael. Enzymes of Molecular Biology. Methods of Molecular Biology. 16. Totowa, NJ: Humana Press. pp. 107–200. ISBN 0-89603-234-5.

[4] Roberts RJ, Vincze T, Posfai J, Macelis D. (2007). "REBASE--enzymes and genes for DNA restriction and modification". Nucleic Acids Res 35 (Database issue): D269–70. doi:10.1093/nar/gkl891. PMID 17202163. http://www.pubmedcentral.nih.gov/articlerender.fcgi?tool=pubmed&pubmedid=17202163.

[5] Adrianne Massey; Helen Kreuzer (2001). Recombinant DNA and Biotechnology: A Guide for Students. Washington, D.C: ASM Press. ISBN 1-55581-176-0.

[6] [Website]http://en.wikipedia.org/wiki/Component_Object_Model

[7] [Website]http://msdn.microsoft.com/en-us/library/aa229332%28VS.60%29.aspx

[8] [Website]http://en.wikipedia.org/wiki/ActiveX

[9] Petroutsos E, Mastering Visual Basic 6, BPB Publication, 2001