International Journal of Computer Applications (0975 – 8887)
Volume 36– No.9, December 2011

Meeting Expectations with Dynamic Link Libraries: Development and Applications

D.A. Adenugba
Department of Physics
The Federal University of Technology, Akure
P.M.B 704, Akure, Ondo State. Nigeria.

ABSTRACT

Expectations, be they scientific or technology or business, are to be met accurately and promptly with dynamic link libraries. In this paper, two class libraries for validation and power law relation (PLR) were developed with Microsoft Visual Studio 2010. The validation class was found to be accurate and handy for quick input data check. The exposed functionalities of PLR was applied to Chebil model to generate information and rainfall rate exceeded 0.01% of time ($R_{0.01}$) data, much-sought-after data for rainfall attenuation prediction model, for seventeen Meteorological sites in Nigeria covering a period of thirty years. Microsoft DataGridView control was customized for current and anticipated use and the methods and properties work to specifications. Not only will software developers benefit from the use of our classes, Atmospheric Scientists, Communication experts, system Engineers and Researchers will find our $R_{0.01}$ data indispensable in their works.

General Terms

Rainfall rate exceeded 0.01%, Server, Dynamic Link Library, Class Library

Keywords

Chebil model, DGV control, Power law relation, Validation

1. INTRODUCTION

Expectations, either social or scientific or business or technology could be met swiftly and accurately with the use of software applications. Social comfort and fitness, scientific and technological breakthrough and self-reliance, business and economical advancement are required in all communities, especially in developing nations. Everybody has a dream for success, growth and progress which could easily be actualized through software packages. The building blocks for software applications are class libraries. Accurate and flexible class libraries are much-sought-after tools by software developers everyday all over the world.

Classes are templates used to create new objects; they are made up of members such as fields (otherwise called data members that hold internal state of object), methods, events, and properties, which may be declared as private or public. Although a class itself is not an object, its entire members define object state and its functionality. A class is "seen" through its public members, while the private members are "unseen" outside the class. Even though a class is not an object itself, an instance of a class is said to be an object and by creating an object of a certain class, you are instantiating an object of the class using a New keyword. The New keyword is a unary operator which takes a type identifier as its operand to produce a reference to a newly created object of the given type. A method, however, that has new name is a constructor [4]. Every class has a default constructor where members are initialized. Besides, a class could be a sealed class declared with the keyword NotInheritable such as Array class, so a new instance of it cannot be created. Base class is a class declared without the Inherits keyword modifier, whereas derived class is a class that is inherited from a base class (that is, its parent class). Abstract base class, also known as Virtual class, is a class that has to be inherited by another class. It is declared with MustInherit keyword, which informs Common Language runtime (CLR) that the class cannot be used as is. Virtual methods are just placeholders for name(s) [3-7].

Data validation is a regular exercise in software development. Failure to capture exceptions (errors) where users could make mistakes leads to software failure and disappointment. Controls unexpected behaviour should be tracked and reported to user or other part of the application. Application's crash is developer's failure to trap all known and unknown exceptions. Thus, adequate checks should be provided to prevent application crash. A dedicated class library, dvValidateCls is thus appropriate for this very significant application chore, which we shall develop in this work.

Power law relation (PRL) is commonly used in expressing many prediction models in communication, hydrology and Meteorology besides its use in Engineering, Mathematics and Physics. VB.6.0 codes had been developed for this very significant parameter [1]. Though adequate for few computations, the method could not timely handle large data. This shortcoming will be addressed in this work and the existing method will be rewritten in VB.NET. It will be polymophized, so that flexibility of data entry could be easily achieved. Its application to Chebil model is expected to generate rainfall rate exceeded 0.01% of time ($R_{0.01}$) data and useful information from long-term mean annual rainfall rate, R. Besides, Microsoft DataGridView (DGV) control will be customized for some common tasks.

2. DATA VALIDATION

Data validation is an essential part of software development that any serious software developers cannot toy with to prevent application crash. Among the commonly checked items are numbers, zero and string. The class library, dvValidateCls developed for checking these common items, contains fifteen methods. The dvValidateZero method has five overloads accepts two to six arguments and determines if the supply number(s) is zero or not. If zero an informed exception message is returned indicating which argument is zero, else ok is returned. The last boolean argument, if true immediately issues the exception message before returning to the caller. If false, no exception message is raised, still the exception message is returned.

The next method dvValidateNumericZero is equally overloaded as the previous one and functions the same way to determine if the argument(s) is zero or not. But in addition, it checks if the object argument(s) is numeric or not. The last overloaded dvValidateNumeric is identical to the two previous ones; however, it checks if the supply argument(s) is simply numeric or not and reports back. Each method has its use at various times. There are occasions when only to find out if an object argument is only numeric, and not to both if zero or not. The three overloaded methods have been repeatedly called at different times to assist us to filter input(s) before using it/them, and they save a lot of space and time to validate input(s).

The Yoruba people said it is because of the afternoon thirst that one fetches water early in the morning. For this reason, apart from the methods used in this work, in anticipation for future software development, some common properties and methods are developed as Table 1 shows. The task(s) of each item which name is in column 2 is given in column 3 to which you are referred.

Table 1: Some Common Properties and Methods

S/n	Name	Task(s)
1.	dvAutoDGV	This method has a reference DGV control argument that is to be resized by calling AutoResizeColumn property of the control.
2.	dvCompleteMsg	This read only property returns a message indicating the completion of an evaluation. It is repeatedly summoned in client application.
3.	dvGetRowColl_MissingRow	It has five arguments that are declared as ByRef except the first which is the DGV control to find its row and column and the last boolean, which simply indicates if exception message is to be displayed or not. This reference declaration is required so that obtained values could be returned to function or sub procedure calling this method. Not only are integers returned as row and column numbers, informed message are given that could be displayed in calling function.
4.	dvInsertTitleInDGVRow	It inserts title in row1 of the supply reference DGVcontrol. The string for title is supplied next as argument. There are six overloads for one column string title to five strings columns; above which the title have to be loaded in a listbox control as argument, any other number of string could be supplied as title for row one of the DVG control.
5.	EnumFMTType	This is a customized enumerate class with five members to assist user to format result.
6.	dvGetSmallNumb	It returns the smaller of two numbers. It is very handy when identical number is to be used in a loop.
7.	dvReport	It accepts a reference label and changes its fore colour as well as text properties to the one's supply. The second overload changes, also the back colour property. The last overload, in addition to the aforementioned alterations, changes the location. This is a useful method for showing messages in a client application.
8.	dvFMTForm	With a single call to this method windows form state, key preview, Back color, fore color and form Border style could be set. It is very useful for start-up windows form, and we extensively utilized it.
9.	dvConvertDegreeToRadian	It converts degrees to radians. Quite useful in Trigonometry functions ,and wherever such conversion is required.
10.	dvConvertRadianToDegree	It converts radians to degrees. Quite useful in Trigonometry functions ,and wherever such conversion is required.

3. POWER LAW RELATION

In its basic form, PRL is expressed thus: aR^b ; a and b are the coefficient and exponent respectively, R is a numeric number either integer or real. Methods with different data types and/or arguments numbers were developed for PLR in a class library, dvPLRCls. The previous method was rewritten in Microsoft Visual Studio 2010 and overloaded to allow more methods of varying data types to be included. PLR has a simple form that makes it attractive to be used in various areas including attenuation prediction model.

The new power law relation method has a reference DataGridView control as argument which contains the inputs: a, R, and b in that order; and after computation, the result is stored in it. It assumes the title is already in row one, thus row one is not read for computation. A column is dynamically inserted in the control for result after the number of rows and columns in the control have been obtained and checked for correct inputs, using dvGetRowColl_MissingRow method (see Table 1, Sn 3). If the number of columns is less than three, for instance, an exception message is thrown and the method gracefully exits. To be sure the correct title is in row 1; dvInsertTitleInDGVRow method is summoned to insert the title in each column of row 1, zero-based. The format type is set to scientific; and a loop is entered where the entire inputs are read, and passed to former method or anyone of the new methods to calculate PLR. After finishing computation, dvAutoDGV method is summoned to resize the DGV control.

Another method accepts two DGV controls for rainfall rate input and outputs. It works the same way as the previous but the results are stored in the second control. Before calling this method a method is called to set a and b values for use. The third method allows the supply of a and b as the first two arguments, the next two DGV control arguments are for R input and results respectively. Other three overloads share the same steps, but one permits user to supply title for row 1. The second allows user to specify the format to use instead of the hard coded, scientific format. The last combines the supply of title string and format. The PowerLawRelatnWkg method helps us to generate PLR workings which could be employed to teach and learn real time. The result, as outputted from our package for a = 2, b = 4 and R = 10, is shown below.

```
Power Law Relation Computed
PLR=a*(R^b)
PLR= 2*(10^4)

PLR= 2(10000.00)
PLR= 20000.00
```

The PLR working always gives the formula and the substituted values which makes it very useful for teaching-learning real time. By this effort, flexibility is freely given to user to call any of the new methods to compute PLR promptly and view both inputs and result together in a DGV control.

3.1 Data, Data Entry and Output Flexibility

Data from seventeen Meteorological stations in Nigeria, which span a period of thirty years (1980-2010), are used for this work. The sites' characteristics are shown in Table 2. Input data could be randomly and interactively supplied, besides being uploaded from Excel file. One thing that is clear is that user is not restricted to only one way of data entry. Flexibility of data entry gives comfort to the users of our package.

Data are read, validated and used to compute result(s).

Table 2: Site Characteristics

S/N	Station Name	Latitude ^{o}N	Longitude ^{o}E	State	GeoPolitical Zones
1	Akure	07 15	05 11	Ondo	South-West
2	Bauchi	10 30	10 00	Bauchi	North-East
3	Calabar	04 58	08 21	Cross-River	South-South
4	Enugu	06 28	07 33	Enugu	South-East
5	Ilorin	08 26	04 29	Kwara	North-West
6	Jos	09 56	08 05	Jos	North-Central
7	Kaduna	10 31	07 26	Kaduna	North-Central
8	Kano	12 00	08 31	Kano	North-West
9	Lagos	06 27	03 23	Lagos	South-West
10	Lokoja	07 47	06 44	Kogi	North-West
11	Maiduguri	11 51	13 05	Borno	North-Central
12	Minna	09 37	06 32	Niger	North-Central
13	Onitsha	06 10	06 47	Anambra	South-East
14	Owerri	05 29	07 02	Imo	South-East
15	Sokoto	13 05	05 15	Sokoto	North-West
16	Warri	05 31	05 45	Delta	South-South
17	Yola	09 13	12 27	Adamawa	North-East

The output(s) could be stored in the same DGV control containing the input(s) or in another reference DGV control. Whichever one, additional column(s) is inserted dynamically in the DGV control for result storage by calling dvAddAnyColumn method. Table 4 conserves space especially for large inputs requiring equally large outputs. This Year-RateMonth data style is the Meteorology data style, which is quite space efficient.

3.2 Applications

The applications of PLR will be seen in Chebil model and Mathematics in this section.

3.2.1 Chebil Model

Chebil (1999) model for converting long-time mean annual rainfall rate data R to rainfall rate exceeded 0.01% of time ($R_{0.01}$) is expressed in PLR as follow:
$R_{0.01} = 12.2903R^{0.2973}$

$R_{0.01}$ is a mandatory input for ITUR rainfall attenuation prediction model and its availability from Meteorological long-time-mean annual rate will aid speedy calculation of attenuation due to rainfall. Results were calculated for all the seventeen stations shown in Table1 and for the entire thirty years using our new PLR method and they are found to be correct as Tables 2-5 depict.

Tables 2-3 are the March 1991-2000 results for Calabar and Enugu sites; and up to one hundred thousand R data could be processed for corresponding $R_{0.01}$ to be calculated. Table 4 is $R_{0.01}$ Maiduguri site result. At this site, for most parts of January, February, March and November, there is no rainfall; and throughout December no rain, hence these months are omitted from the Table. If rate is not numeric, NR for No Result is returned, but for R=0 there is no calculation but R=0 is inserted in the DGV control to indicate what is wrong. Table 5 shows the result for mixed a and b, and R for Calabar site.

Table 2: $R_{0.01}$ result for Calabar site using Chebil Model

a	R	b	R0.01
12.2903	88.3	0.2973	46.56897
12.2903	271.5	0.2973	65.03127
12.2903	177.7	0.2973	57.33167
12.2903	167.3	0.2973	56.31289
12.2903	366.1	0.2973	71.07562
12.2903	161.5	0.2973	55.72527
12.2903	139.4	0.2973	53.33985
12.2903	174.0	0.2973	56.97414
12.2903	203.0	0.2973	59.64596
12.2903	95.9	0.2973	47.72623

Table 3: $R_{0.01}$ result for Enugu site using Chebil Model

a	R	b	R0.01
12.2903	63.4	0.2973	42.20109
12.2903	111.5	0.2973	49.91342
12.2903	62.8	0.2973	42.08195
12.2903	9.7	0.2973	24.1507
12.2903	90.2	0.2973	46.86465
12.2903	48.6	0.2973	38.99415
12.2903	111.6	0.2973	49.92672
12.2903	25.8	0.2973	32.30257
12.2903	30	0.2973	33.78397
12.2903	32.3	0.2973	34.53412

Table 4: $R_{0.01}$ result for Maiduguri site using Chebil Model

Year	Apr	May	Jun	Jul	Aug	Sep	Oct
1991	2.37E+01	4.49E+01	4.84E+01	4.69E+01	5.99E+01	8.59E+00	1.63E+01
1992	1.06E+01	4.05E+01	3.69E+01	5.02E+01	6.68E+01	4.19E+01	2.73E+01
1993	2.03E+01	4.20E+01	2.99E+01	5.97E+01	5.45E+01	3.92E+01	R=0
1994	1.75E+01	2.51E+01	3.94E+01	4.73E+01	5.64E+01	4.90E+01	3.11E+01
1995	2.58E+01	1.93E+01	3.81E+01	6.53E+01	5.75E+01	5.61E+01	2.20E+01
1996	1.98E+01	3.49E+01	4.11E+01	5.90E+01	5.72E+01	5.31E+01	3.17E+01
1997	2.98E+01	3.64E+01	5.37E+01	5.31E+01	5.22E+01	4.16E+01	3.02E+01
1998	1.39E+01	3.19E+01	4.15E+01	5.75E+01	6.48E+01	5.47E+01	1.30E+01
1999	R=0	3.17E+01	3.23E+01	6.97E+01	5.82E+01	6.20E+01	3.43E+01
2000	R=0	2.63E+01	3.51E+01	5.95E+01	6.33E+01	4.19E+01	3.46E+01

Table 5: $R_{0.01}$ result for mixed a and b; R from Calabar sites

a	R	b	PLR	Source
12.2903	479.0	0.2973	76.98857	Chebil
4.1000	387.7	-0.2100	1.172735	MP
10.0000	281.0	3.0000	221880400	Arbitrary
8.0000	294.2	2.0000	692429.1	Arbitrary
11.0000	375.2	-0.1000	6.08088	Arbitrary

3.2.2 Mathematics

Random number could be used to calculate PLR. Table 6 shows accurate result of some arbitrary data.

Table 6: Power Law relation methods result

a	R	b	PLR
12	6	15	5.64E+12
11	5	9	2.15E+07
3	9	8	1.29E+08
2	3	15	2.87E+07
9	5	19	1.72E+14
2	6	11	7.26E+08
2	7	4	4.80E+03
6	12	2	8.64E+02
9	12	8	3.87E+09

4. RESULTS AND DISCUSSION

The reference DGV control, which holds up to one hundred thousand data (our package capacity) that could be processed at a single location, was used in our new PLR methods. This aids timely computation of result for PLR especially for very large data unlike the previous method which returns only a single result each time. In Tables 2 and 3, for a given site, a and b have to be repeated for all R values, which is not efficient; but for different a and b values, the method is very efficient as Table 5 for different a and b shows. Let it be mentioned again that a, R and b are expected to be in the DGV control in this order for correct result.

Programmatically, $R_{0.01}$ data could be made available for use by applying our PLR methods. This eliminates the use of rate maps to determine $R_{0.01}$ for use where not available. Figures 1-6 are typical plots of $R_{0.01}$ for six of the sites. It was observed that at the commencement of rain around March in the south and April in the North and at the end of the raining season in September-October, $R_{0.01}$ values are generally low. The highest values of $R_{0.01}$ are obtained between June and July when rain is at its peak.

The frequency of rainfall is also seen in the cluster pattern of the graphs; more rain in the south, especially at coastal sites like Lagos and Warri where there is rainfall throughout the year. All the northern sites plot patterns are not as densely packed as the southern sites owing to fewer rainfalls. Jos site, as could be seen in figure 3, has more rainfalls than Bauchi site.

The combined plots for Kaduna (Kad), Sokoto (Sok), Yola (Yol) and Onitsha (Oni) sites for June-August are shown in figure 7. The summary result in Table 7 indicates that $R_{0.01}$ is generally low at the beginning of rain (January-March) and high around June-September. Warri site has the highest value at the beginning, whereas Calabar site is a little higher than Warri at the end of the raining period. Although there is no rainfall throughout the year in the north, the intensity of rainfall is still high.

Table 7: Summary of $R_{0.01}$ (mm/h) Result

S/n	Site	Range of $R_{0.01}$
1	Akure	10.00 – 76.20
2	Bauchi	10.00 – 78.90
3	Calabar	6.20 – 89.90
4	Enugu	10.00 – 82.10
5	Ilorin	8.59 – 85.40
6	Jos	6.20 – 73.30
7	Kaduna	6.20 – 80.20
8	Kano	8.59 – 82.50
9	Lagos	8.59 – 85.40
10	Lokoja	8.59 – 74.40
11	Maiduguri	8.59 – 71.40
12	Minna	8.59 – 77.90
13	Ontisha	6.20 – 79.30
14	Owerri	6.20 – 84.30
15	Sokoto	9.36 – 71.60
16	Warri	16.30 – 89.50
17	Yola	11.10 - 70.40

4.1 Exceptions

Exceptions are errors that are expected to be trapped in an application in order to prevent its crash. According to Dave (2002), an exception is any occurrence that is not considered part of normal, expected program flow. Consequently, a minute exception will prohibit a program from continuing its current normal activity. Exceptions are harmful to all programs as their presence could cause program to malfunction and/or crash the application. Just as harm is not far from a person who eats fish bones in the dark, so is a programmer who develops a package without error check(s) at all levels where error(s) is likely to occur.

What a foreign object is to a human body is what an exception is to a program. Unless it is removed, the program is sick and will decline to function. There is availability of exception classes that provide functionalities for exception types. Message, one of the properties in the exception class, was often summoned to describe what is wrong and offer possible solution, if error(s) is unknown ahead of time. However, where certain errors are known ahead of time, we provided our own informed description of the error and solution to it.

Another property we repeatedly employed is ToString that returns a text representation of any exception caught. Besides, we utilized ArithmeticException property to trap DivideByZeroException, NotFinitNumberException and OverflowException. It is enough to say that enough error checks are inserted in all our functions, sub-procedures and methods; for more on Exception see [3-6]

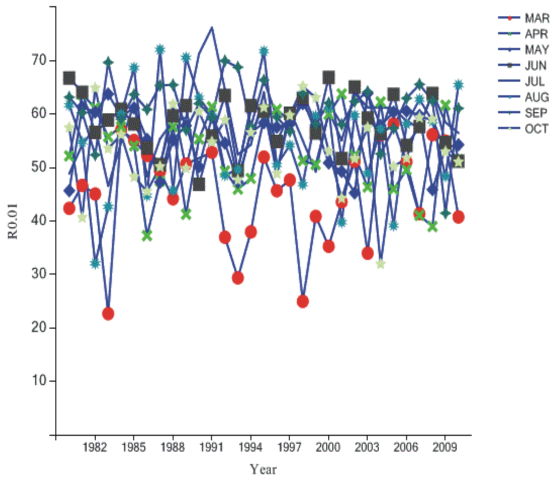

Figure 1: $R_{0.01}$ Plot for Akure Site

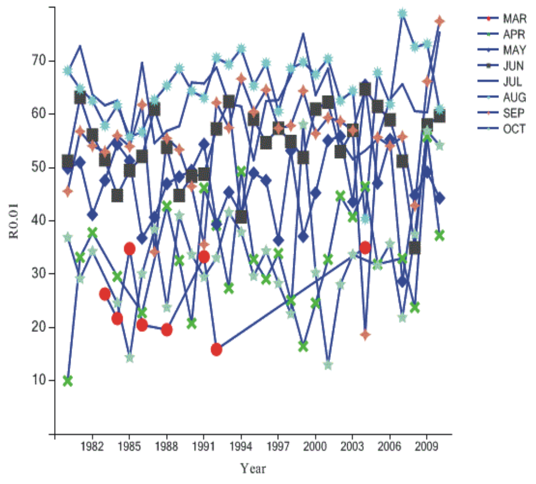

Figure 2: $R_{0.01}$ Plot for Bauchi Site

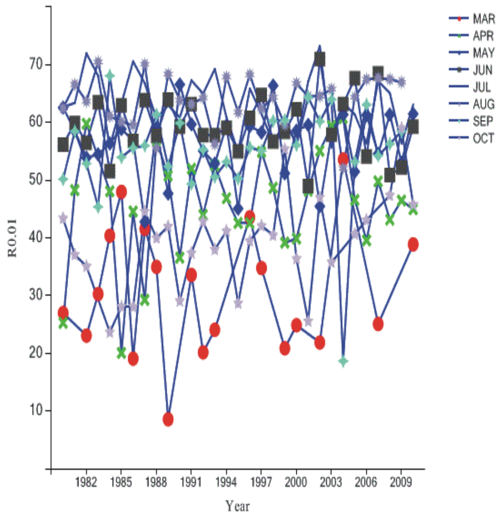

Figure 3: $R_{0.01}$ Plot for Jos Site

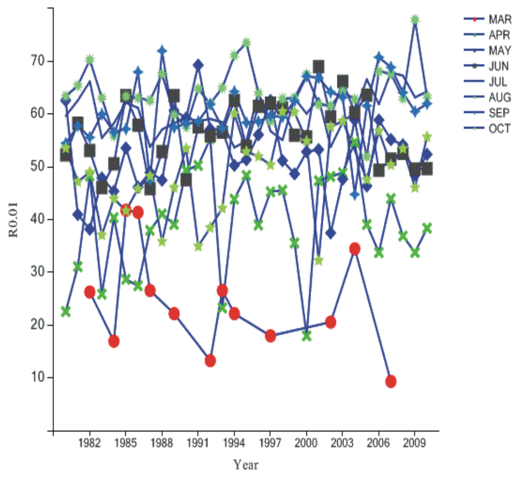

Figure 4: $R_{0.01}$ Plot for Minna Site

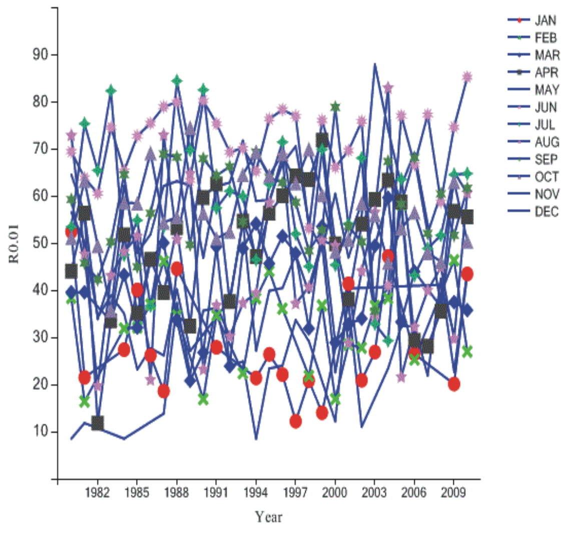

Figure 5: $R_{0.01}$ Plot for Lagos Site

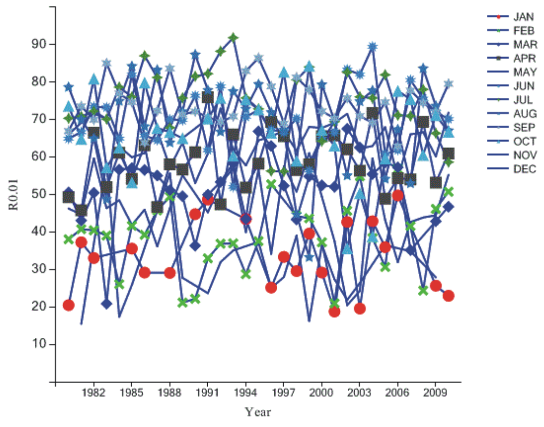

Figure 6: $R_{0.01}$ Plot for Warri Site

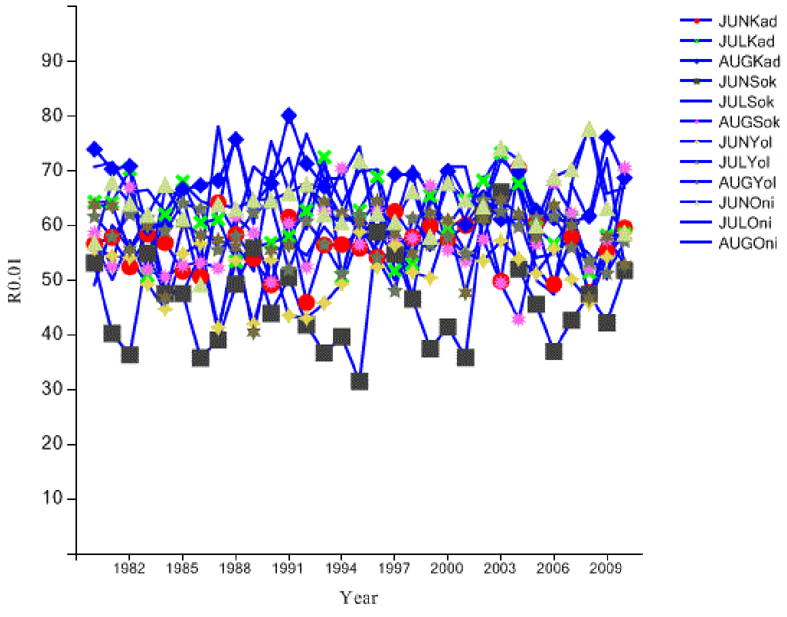

Figure 7: $R_{0.01}$ Plot for Combined Kaduna (Kad), Sokoto (Sok), Yola(Yol) and Onisha(Oni) Sites for Jun-Aug

5. CONCLUSION

Swift Software development expectations could be met with flexible and accurate servers. Our validation server works to expectation; and by further customizing Microsoft DGV control, developers are empowered to quickly develop applications. The PLR class library could be called with diverse data formats to output accurate PLR results. Its application to Chebil model yields $R_{0.01}$ data which are much-needed in rainfall attenuation prediction models for thirty years and for seventeen sites in Nigeria. Microsoft DGV should further be customized for more use, both scientifically and socially.

6. REFERENCES

[1] Adenugba, D.A.2007. An Atmospheric Server for some Atmospheric Parameters: Development and Applications. J. Res. Sci. Mgt. 5(1); 91-104.

[2] Chebil, J and Rahman, T.A.1999. Development of 1min Rain Rate Contour Maps for Microwave Applications in Malaysia Peninsula. Electronics Letts. (35) 1712 – 1774.

[3] Craig Utley 2001. A Programmer's Guide to Visual Basic. NET. Sams Publishing.

[4] Dave Grundgeiger 2002. Programming Visual Basic.NET. O' Reilly and Associate, Inc.

[5] Evangelos Petroutsos 2002. Mastering Visual Basic.NET. Sybex, Inc.

[6] Thearon and Bryan. 2010. Beginning Microsoft Visual Basic 2010. Wiley Publishing Inc.

[7] Tim Patrick (2008), Programming Visual Basic 2008. O' Reilly and Associate, Inc.