

EVOLUTION...



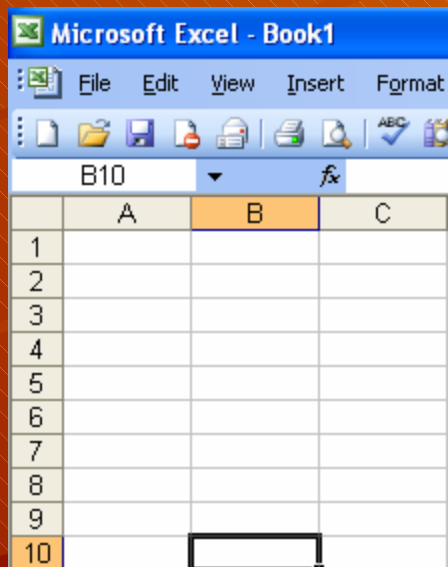
The ICISworkbook

Warren Vincent E. Constantino, Systems Analyst / Programmer
Crops Research Informatics Laboratory, IRRI-CIMMYT Alliance

Text Files: An Approach to Speedup ICIS Workbook's Load & Retrieve Functions

Premise:

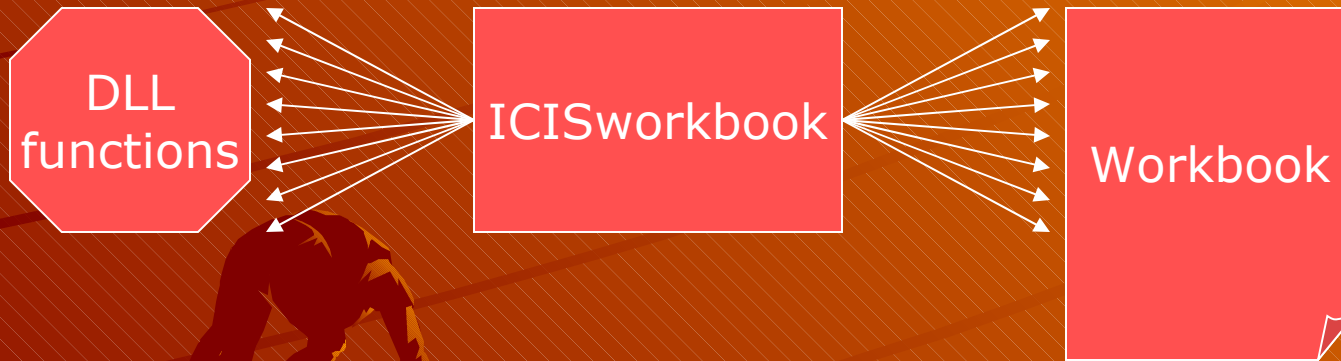
It has been observed that there is an overhead in execution time when accessing MS Excel's rows & columns programatically & even with the screen updating turned off. This slows down the Load & Retrieve functions. The objective therefore was to eliminate this overhead to make the functions run faster.



Text Files: An Approach to Speedup ICIS Workbook's Load & Retrieve Functions

Implementation:

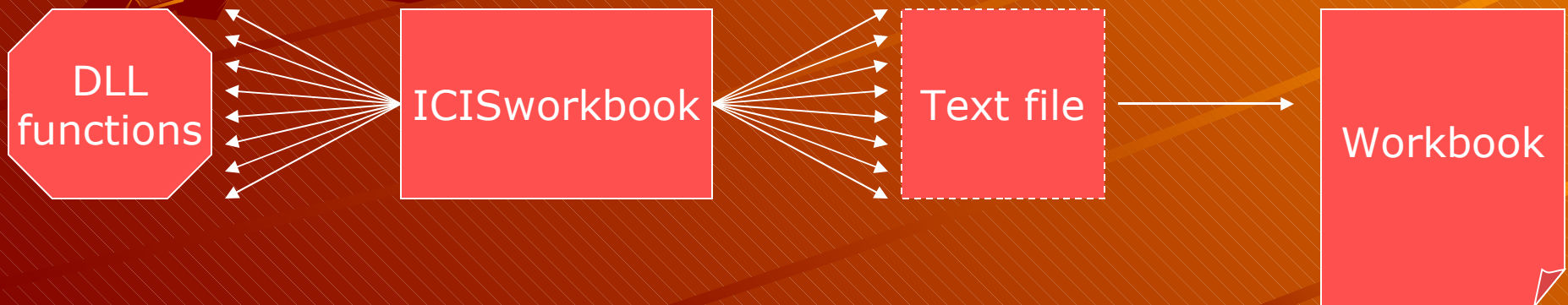
From: Calling DLL functions & writing directly to a workbook



66% less time
in Loading

98.6% less time
in Retrieving

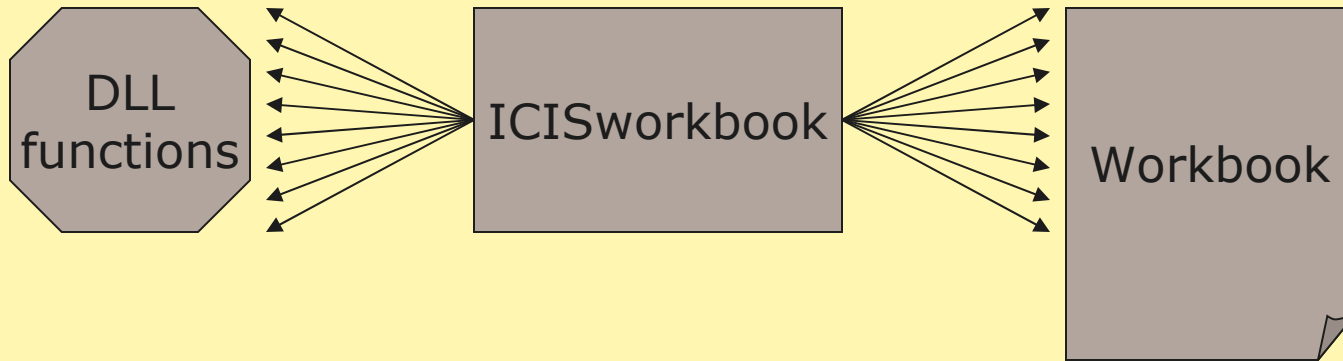
To: Calling DLL fxns & writing to a text file, then text file converted to a workbook



The ICIS Workbook: What was done?

Implementation:

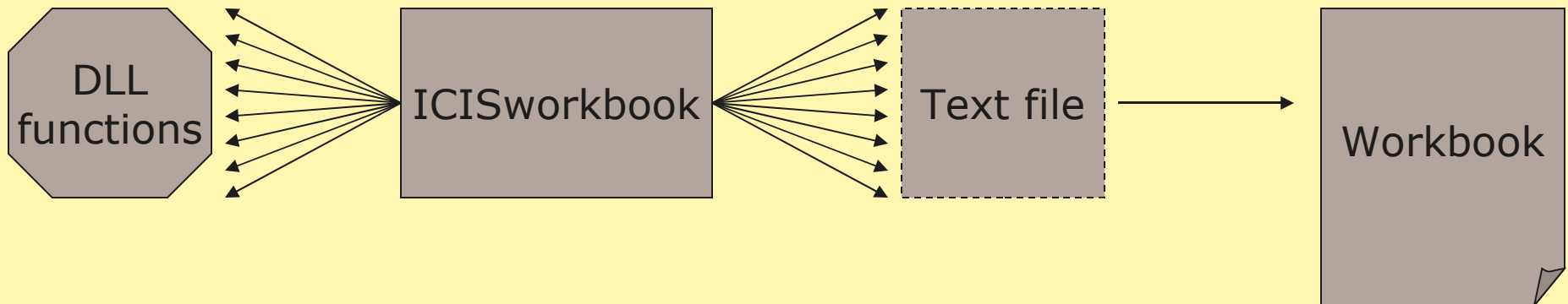
From: Calling DLL functions & writing directly to a workbook



**66% less time
in Loading**

**98.6% less time
in Retrieving**

To: Calling DLL fxns & writing to a text file, then text file converted to a workbook



The ICIS Workbook: What was done?

Actual Implementation:

Number of Rows:	Time of Retrieval:	
	<u>Old Time</u>	<u>New Time</u>
1500	95 sec.	9 sec.
3000	222 sec.	21 sec.
6000	613 sec.	44 sec.

Retrieving directly on
an Excel Workbook


Retrieving first on a text
file then converting it to
an Excel Workbook

**10X
faster**



The ICIS Workbook: What was done?

Is the job done? Not Yet!



The ICIS Workbook: What ~~was still to be~~ done?

Is the job done? Not Yet!

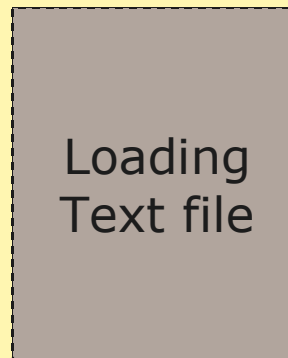


The ICIS Workbook: What can still be done?

Loading a study is a complex process.

Though using a text file as storage buffer promises 66% less time, incorporating the load of calling DLL functions would further slow down the process.

Moreover, aside from loading, another time-consuming process is involved w/c is checking first the validity of the data to be loaded.



How can we
counter this?



Must pass through the data 2X w/c means processing time doubles

The ICIS Workbook: What can still be done?

Let's tak

```
Public Sub ReadTextfile()  
    Dim intFileNo As Integer  
    Dim strRowValue As String  
    Dim lngDelimPos As Long, lngStartPos As Long  
    Dim varStart As Variant, varEnd As Variant  
    Dim lngMinutes As Long, lngSeconds As Long  
  
    'read the textfile  
    varStart = Time  
    intFileNo = FreeFile  
    Open CurDir & "\" & TESTFILE & ".txt" For Input As #intFileNo  
    Do While Not EOF(intFileNo)  
        Line Input #intFileNo, strRowValue  
        lngStartPos = 1  
        Do  
            lngDelimPos = InStr(lngStartPos, strRowValue, Chr(9), vbTextCompare)  
            If lngDelimPos = 0 Then Exit Do  
            lngStartPos = lngDelimPos + 1  
        Loop  
        If lngDelimPos = 0 Then  
            lngStartPos = lngDelimPos + 1  
        End If  
    Loop  
    Close #intFileNo  
    varEnd = Time  
    lngMinutes = DateDiff("n", varStart, varEnd)  
    lngSeconds = DateDiff("s", varStart, varEnd)  
    MsgBox "Time Started: " & varStart & Chr(13) & Chr(10) & _  
        "Time Ended: " & varEnd & Chr(13) & Chr(10) & _  
        "Interval(M): " & lngMinutes & Chr(13) & Chr(10) & _  
        "Interval(S): " & lngSeconds, vbInformation, "Reading Textfile"  
  
End Sub
```

MS Excel can save a workbook as a
TAB-delimited text file w/c is
basically a sequential file..ed?

This means that to access the contents of
the file, we must read it line-by-line from
top to bottom...

TAB

In order to distinguish between
cells in one row, we must look
for the TAB characters as
delimiters...

This definitely
consumes time...

The ICIS Workbook: What can still be done?

Recomm

Use a

```
Public Sub ReadTextfile()  
    Dim intFileNo As Integer  
    Dim strRowValue As String  
    Dim lngDelimPos As Long, lngStartPos As Long  
    Dim varStart As Variant, varEnd As Variant  
    Dim lngMinutes As Long, lngSeconds As Long  
  
    'read the textfile  
    varStart = Time  
    intFileNo = FreeFile  
    Open CurDir & "\" & TESTFILE & ".txt" For Input As #intFileNo  
    Do While Not EOF(intFileNo)  
        Line Input #intFileNo, strRowValue  
        lngStartPos = 1  
        Do  
            lngDelimPos = InStr(lngStartPos, strRowValue, Chr(9), vbTextCompare)  
            If lngDelimPos = 0 Then Exit Do  
            lngStartPos = lngDelimPos + 1  
        Loop  
        If lngDelimPos = 0 Then  
            lngStartPos = lngDelimPos + 1  
        End If  
    Loop  
    Close #intFileNo  
    varEnd = Time  
    lngMinutes = DateDiff("n", varStart, varEnd)  
    lngSeconds = DateDiff("s", varStart, varEnd)  
    MsgBox "Time Started: " & varStart & Chr(13) & Chr(10) & _  
        "Time Ended: " & varEnd & Chr(13) & Chr(10) & _  
        "Interval(M): " & lngMinutes & Chr(13) & Chr(10) & _  
        "Interval(S): " & lngSeconds, vbInformation, "Reading Textfile"  
  
End Sub
```

The ICIS Workbook: What can still be done?

Using a Random Access File...

```
Type Person
  LName as String*12
  FName as String*8
  Age as Integer
End Type
```

It is necessary to define a fixed-length record type & assign the length of this record as the size of the basic unit of the random-access file...

```
Sub ReadRandom()
```

```
  Dim P as Person
```

Random
Access
Text file

```
  Open "People.txt" For Random As #1 Len=Len(P)
```

```
  For i = 1 To Int( LOF(1) / Len(P) )
```

```
    Get #1, i, P
```

```
    Debug.Print P.LName, P.FName, P.Age
```

```
  Next
```

```
  Close #1
```

```
End Sub
```

With this, it is so easy to jump to any record anywhere in the file by simply specifying the record number...

The ICIS Workbook: What can still be done?

But in conjunction with changing the source file format, another improvement is also being considered...

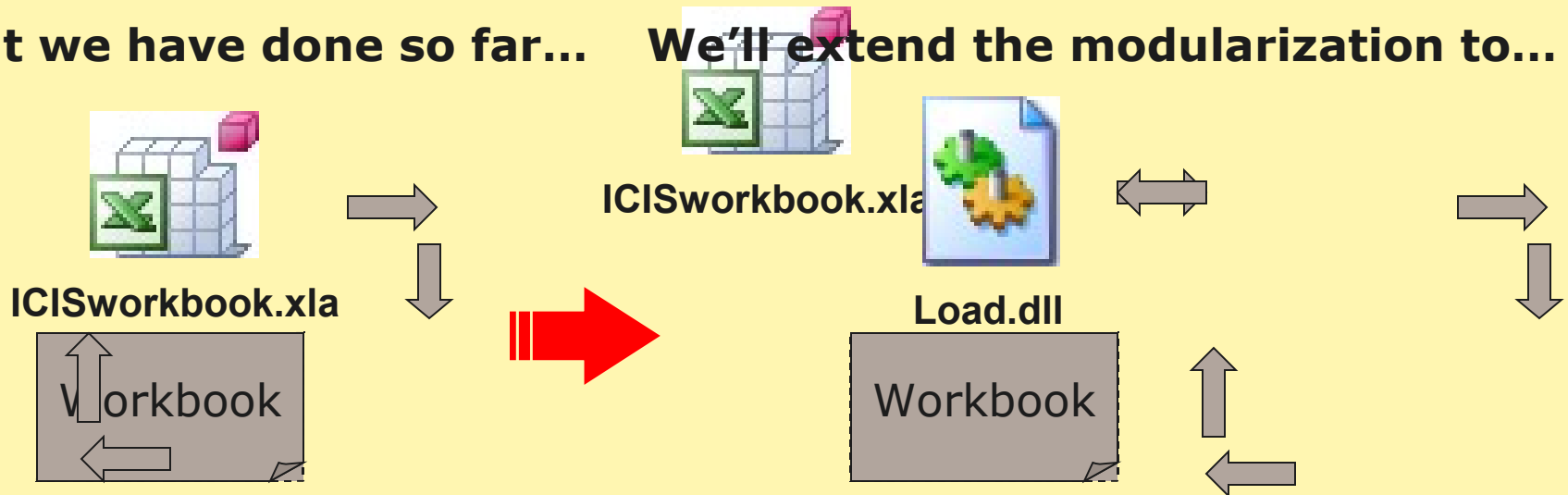
That is improving the speed of execution by using a generally-known fast programming language which is C.

The idea is to rewrite the Loading and Retrieving functions by changing from:

- 1 Using Sequential File → Using Random-Access File
- 2 VBA accessing DLL & text file → C accessing DLL & text file

What we have done so far...

We'll extend the modularization to...



The ICIS Workbook: What is the future?

The single application will be broken down into modules...



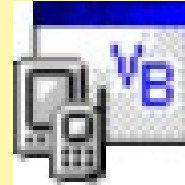
ICISworkbook.xla

VBA handling
the MS Excel
Components



Load.dll

C handling the
manipulation
of Text files



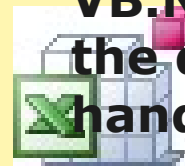
Device.exe

VB.NET handling
the data entry in
hand-held devices



Web.jsp

Java handling
the web study
retrieval



ICISworkbook.xla

And continuous improvement will be done on...



ICIS32.dll



File Conversion Wizard



Ontology Finder



Data Validation

The ICIS Workbook: What is the future?

The single application will be broken down into modules...



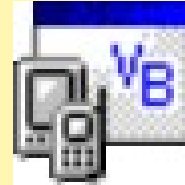
ICISworkbook.xla

VBA handling
the MS Excel
Components



Load.dll

C handling the
manipulation
of Text files



Device.exe

VB.NET handling
the data entry in
hand-held devices



Web.jsp

Java handling
the web study
retrieval

And continuous improvement will be done on...



ICIS32.dll



File Conversion Wizard

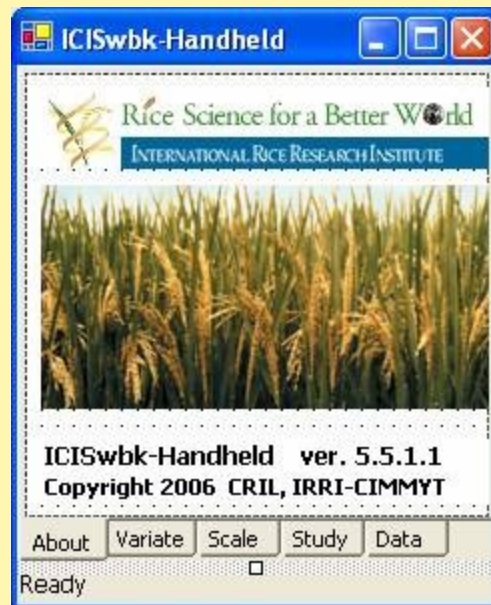


Ontology Finder

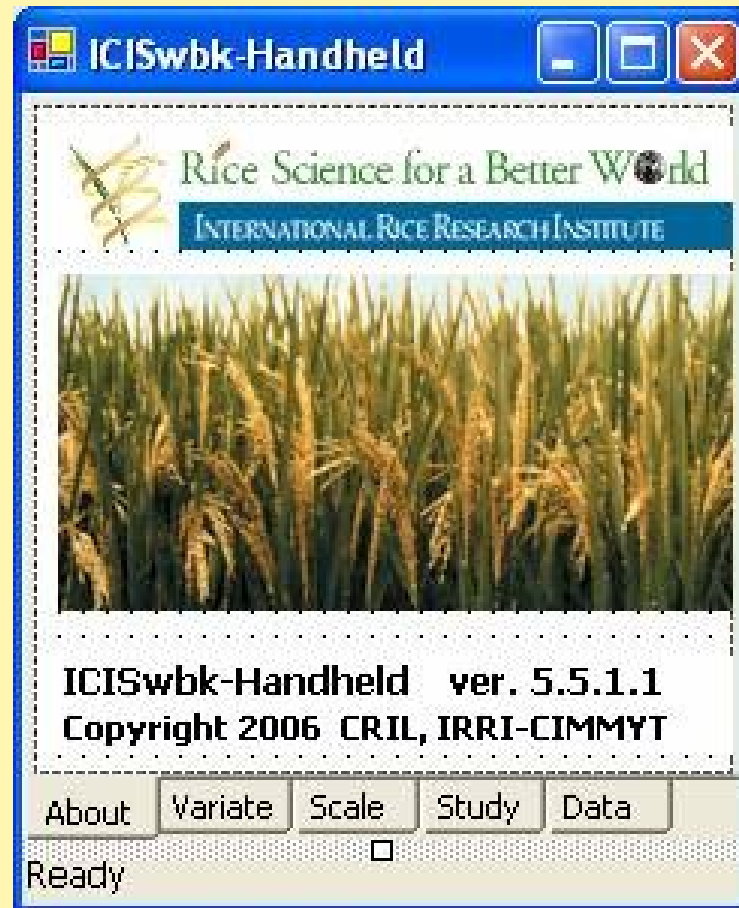


Data Validation

The ICIS Workbook: ICISwbk-Handheld



The ICIS Workbook: ICISwbk-Handheld



The ICIS Workbook: ICISwbk-Handheld

Code: Name: + - =

BPH Brown Planthopper

Scale: 1 to 9 step 2 includ 0

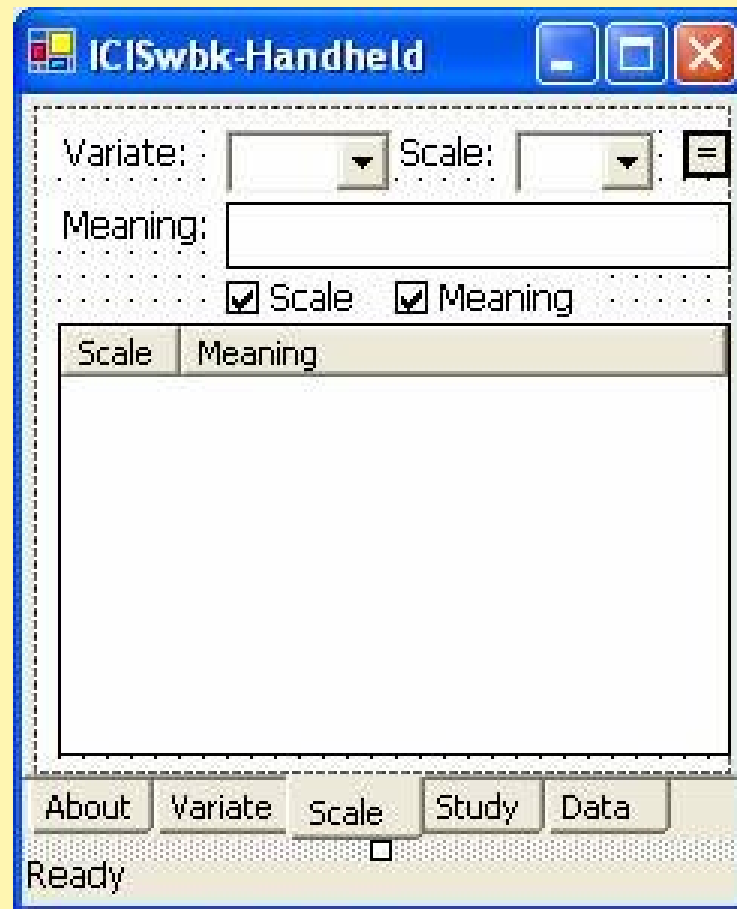
Code Name Scale

Code	Name	Scale
------	------	-------

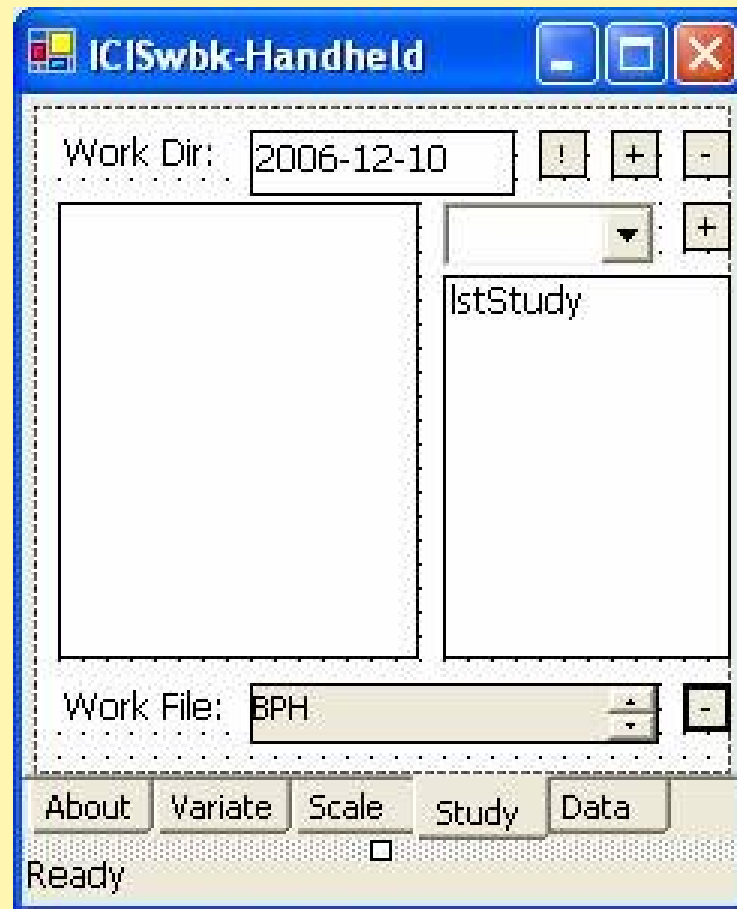
About Variate Scale Study Data

Ready

The ICIS Workbook: ICISwbk-Handheld



The ICIS Workbook: ICISwbk-Handheld



The ICIS Workbook: ICISwbk-Handheld

