Overview of summer research on Emulation project

Contents:

Purpose of the document:

The purpose of the document is to give a detailed overview of the summer research project on Emulation. The summer project focused on creating hundreds of automation scripts to evaluate emulation assistant. Many interesting details were gathered during the process of scripting. This document serves as a complete reference material to create legacy executable.

I. Introduction

Digital objects are vulnerable to software or hardware obsolescence and physical degradation. These digital objects have huge amount of information and cultural heritage that will be lost if the objects are not preserved. Rework on all the available digital objects to make them run on present and future systems demands high costs and time. The extensive variety of digital objects makes preservation a challenging task. The digital objects vary in a) type of software that used to write the program to run it, b) type of hardware that runs the software, c) type of formatting, for example, the object can be a multimedia application or may contain simple text or PDF files. To open these objects, additional software is needed to be installed before we use the object, d) special characteristics like few objects are written in international language or made for a certain set of population etc.

While preserving the digital objects, we must also bear in mind that the future user should be able to use them and understand the information present on them. We ascertain that the future systems will be more sophisticated than the present ones. We no longer use the DOS mode of Microsoft Operating Systems and older versions of 16-bit Windows have become obsolete. We now use a 64-bit version of Windows which will make 32-bit obsolete in not much time. The future systems will have more enhanced software and hardware features and the existing mouse and key stoke interfaces may no longer be in use. Also the future user will be unaware of any technology used currently. It will be a difficult task for the user to learn the older technologies and access the digital objects.

During the summer, we worked on writing helper scripts that can automate the process of installing the software that will enable future user to access the digital object with minimum or no knowledge of existing systems. We used emulation, one of the two widely used strategies that can render the digital object in its original form on the emulated platform. The basic approach was to take bunch of ISOs, mount each of them on the virtual machine and follow the entire procedure of each installation manually. Finally, write scripts capturing the procedure and automating each installation. Every step had its own complications and handling them gave out some interesting results. The rest of the document talks about the research in more detail.

II. Pre-work/ Initial Setup:

Close to a thousand of digital objects were taken to serve the purpose of research. Most of these digital objects were available on CD-ROMs. They were either Government or Commercial, English or International. The information on the CD-ROMs was vast that contained studies on art, culture, demography etc. The objects were picked from the library reserves randomly. Not all the objects might be useful, or they might not run fine, or they might not have upgraded with latest information. We leave all these to the library administrators. The digital objects were converted to ISO image and reserved to begin experimentation. An ISO image is a CD-ROM or

DVD image saved in ISO-9660 format that can be used as a virtual copy of its original disk. The ISOs are not opened, but they are mounted.

VMware Server/Workstation was setup to virtually create the original environment in order to run the CD-ROM images. VMware simulates the set of hardware that is required by the guest operating system. It supports list of hardware devices like hard disk, network adapter, CD/DVD drive etc., which enables bridging with the host operating system on which VMware runs. ISOs can be mounted on CD/DVD drive and this will work similar as you try to run or open a CD-ROM on an original system. Different operating systems are required to support the software on the CD-ROMs. Few needed Windows or Linux and the other needed Mac. Also they operated on various versions of Windows Operating System from Windows 3.1 to Windows XP and these images were our prime focus.

Choosing a virtual machine that can support maximum digital objects was the next task. We came across few ISOs that ran on a 16- bit Windows machine and few on 32-bit Windows. Few needed DOS-mode of Windows to function properly, while some needed graphics and screen resolutions from earlier versions of Windows. Microsoft Windows XP Professional (32-bit) is a best-suited operating system that can meet the requirements of many digital images. We chose Windows XP for two reasons primarily. First, most of the ISOs used software and hardware that was supported by Windows XP. Second, Windows XP is backward compatible and it can run a program in compatibility mode of 16-bit Windows. It also provides settings to run the program in display with 256 colors or with the screen resolution of 640 x 480.

The helper scripts needed a scripting tool to automate each installation. AutoIt, a freeware GUI Scripting language for Microsoft Windows was used for this purpose. It is a BASIC-like scripting language that facilitates automating the user tasks by capturing the keystrokes, mouse movements or window controls. AutoIt was installed on the Windows XP Professional virtual machine on VMware Server/Workstation and stored on the baseline of the virtual machine by taking a snapshot. This gives a fully-fledged initial setup to perform the research.

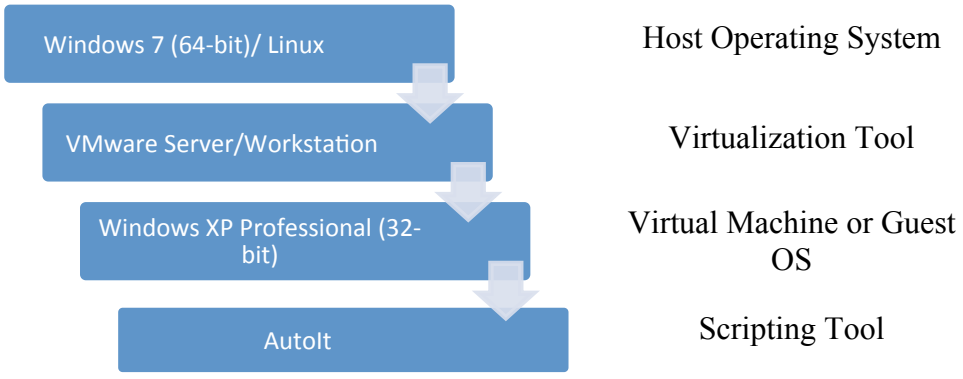

Figure 1: The initial setup to create automation scripts

III. Project Work:

As a start up for the project, a set of about 100 ISOs were taken from the reserve and then completed scripting for about 400 ISOs. Each ISO was mounted on the Windows XP virtual machine and a script was written to automate its installation. We divide our research into following stages:

1. Exploring the ISO image.
2. Creating Automation Scripts
3. Analysis – *new baseline*
4. Results

These stages will serve as a guide to write helper applications in evaluating the assisted emulation.

1. Exploring the ISO image

Initially, each mounted ISO image was examined to gather various details. The details include the type of data present on the ISO image, whether the software is an executable or some other application such as media or text file, the procedure to install the software, is the software dependent on any other application or ISO image(s), does it require any changes in system configuration settings etc. The README files on the image give a detailed summary on complete method of installation and list of directories created (if any) to access the installed product. But not all the ISOs had README files on them. In such case, we searched for the path for the right executable on the ISO and executed it manually by following the instructions that appeared on the screen. It was not very difficult to find the installed directory that had the final product as many installations either opened those directories or prompted to enter the directory name while installing or the common location was C:\(Program Files) or they opened the installed files automatically. After a successful installation, we checked if the executable rendered the complete information and that no components or data on the file were missing.

We found that many of the ISOs would not run properly due to many reasons. Dependency on additional software that was not installed on the Virtual Machine hindered the execution of the ISO. Few ISOs had the additional software included with the package and either installed them in the process of the execution or needed a separate installation. The ISOs that did not include the additional software raised a set of challenges in finding out the appropriate software needed to run the program. Neither had we found any README file or any other documentation that gave any related information. For International software, the README file was in a foreign language that needed to be translated in order to understand its content. We explored ways to deal with such exceptions.

Though few programs were installed successfully, but the data was still missing on them. For example, an ISO had PDF file that needed a QuickTime plugin to run its media files. If any of the two software, PDF or QuickTime, was found missing, then the installation became useless. For most cases, we found the software by looking at the extensions of the files. But finding the

name of the right software was not good enough, as the software came in various versions, a newer upgrading its older version. We tried multiple versions randomly that we thought would be best to run the program. Yet there were few ISOs that would not start installation unless they found the required software on the machine and threw an error message denoting the absence of the software. Few messages were hard to decipher and made it trying to understand if the message was about missing software or any other system configuration files. For example, we encountered a message "mtb30run.exe missing". Mtb30run is multimedia toolbox software and is very hard to find. We paired up the ISOs with required additional software with the ISO in the same directory and mounted each ISO on separate drives of the Virtual Machine and created a script accordingly that would install the additional software first followed by the actual software.

There were a bunch of programs in International languages. These programs would open files displaying contents that were not in a readable format. They needed Windows to run the particular Language option to make sense in the displayed content. There is a provision on Windows XP to select the Unicode of the required language. Follow the subsequent steps to operate Windows in other foreign languages:

a) Find the suitable language. This can be found in a separate documentation that is maintained with the ISO while converting the CD-ROM to its equivalent ISO image.
b) Install East Asian Languages to support programs in Chinese, Japanese, Korean and other Asian languages. Other languages like German, French did not need any additional package.
c) Go to Start→ Control Panel → Regional and Language options → Languages → Check "Install files for East Asian Languages" and click OK
d) It then prompts to insert the Windows XP Professional disk. Insert the follow the instructions.
e) Reboot the machine after you have finished installation.
f) After rebooting, go to Regional and Language options → Regional Options and you can select the required language and it will work.

Few programs needed 3MB of virtual memory on Windows XP. To achieve this follow the following procedure:

a) Control Panel → System → Advanced → (performance) Settings → Performance Options → Advanced → click Change → check No paging file → click Set → click OK
b) Restart the machine.

Few needed to set Display settings for 16 bit color quality. Below are the steps to do this:

a) Go to Control Panel → Display → Settings → Color Quality → select Medium (16 bit) → click Apply → Monitor Settings → Click Yes → click Ok.

Few programs were developed in DOS mode during the era of early Windows operating system. Windows XP support for compatibility mode helped us to run these programs. To do this,

a) Right Click on the file that you wish to open in compatibility mode and click Properties.
b) Go to Compatibility option and check "Run this Program in compatibility mode for:" and select the right option from the provided list of operating systems.
c) Click Apply and then OK.

In order to run in DOS mode, few ISOs needed to allocate some value to extended memory (XMS).

a) Go to C:\Windows\system32, right click on command.com and go to Properties→ Memory.
b) Set Expanded (EMS) memory and Extended (XMS) memory to 16384.
c) Go to Misc and uncheck Alt+Space, Alt+Esc and Alt+Enter and Click Ok
d) Run the program from command.com prompt.

Few software came in set of multiple discs. The set contained independent and dependent discs. The independent discs where easy to handle as they could be run individually. For the dependent scripts, the program on one disk required a file present on some other disc. We tested these programs by mounting all the ISOs in the set on multiple drives simultaneously so that the program on one ISO could access the required file from other ISO mounted on one of the other drives. Few ISOs were very friendly, they auto ran and did not need any kind of scripting. Yet we explored those ISOs to check if no component was missing.

Exploring the ISOs demonstrated the various requirements of the programs, special cases, challenges in handling these special cases and ways to resolve the cases and successfully install the programs. It is evident that the number of special cases will rise with more number of ISOs. Clearly, the variant nature of the software will demand for knowledge of these obsolete systems and it is important that we explore the ISOs to gather as much information as possible to preserve the information on the digital objects.

2. Creating Automation Scripts – the programming phase.

For each ISO, having known about its requirements, we then began writing scripts that could automate its installation. There are many Scripting tools available that can create automation scripts. We chose AutoIt for it is a freeware tool and has special control features that can make the scripts more reliable. Its BASIC-like structure is easy to learn and write scripts (*.au3 files) for complicated installations. Tutorial on AutoIt can be found at http://www.autoitscript.com/autoit3/docs/. An example of AutoIt script is as follows:

Example.au3

```
Run("D:\SETUP.EXE")
WinWait("Setup")
ControlClick("Setup", "", "Button1")
WinWait("", "successfully installed")
ControlClick("", "successfully installed", "Button1", "", 2)
WinWait("CD-ROM Delos")
ControlListView("CD-ROM Delos", "", "SysListView321", "Select", ControlListView
        ("CD-ROM Delos", "", "SysListView321", "FindItem", "Delos"))
ControlSend("CD-ROM Delos", "", "SysListView321", "!{ENTER}")
WinWait("Delos Properties", "Shortcut")
WinClose("CD-ROM Delos")
ControlCommand("Delos Properties", "Shortcut", "SysTabControl321", "TabRight")
WinWait("Delos Properties", "Compatibility")
SendKeepActive("Delos Properties", "Compatibility")
Send("{TAB}{SPACE}")
ControlClick("Delos Properties", "Compatibility", "Button11")
ControlClick("Delos Properties", "Compatibility", "Button10")
Run("D:\WIN\DELOS.EXE")
```

AutoIt has a special feature that can convert its au3 file to an executable (.exe) file that can run the script on any machine that does not have AutoIt installed on it. These executables were placed along with the respective ISOs in each ISO directory and the emulation assistant ran these executables to automatically install the software.

Our script catalogue also includes the scripts that automated all the special cases that are discussed in the Exploring the ISO section. With autorun software, we saw that the autorun.inf programs were not very consistent, at times they ran and at times they broke. Also the emulation assistant needed an executable for each ISO to install the program on the Virtual Machine or else it threw an error that it did not find the program that needs to be run. Therefore, we wrote small scripts that close all the windows which open up when the ISO is mounted and then run the actual executable on the ISO image instead of playing the autorun file.

An au3 script uses window name and the commands that are passed to the window to automate the process of installation. Most of the commands are passed through key strokes; the shortcut keys enable us move through the installation smoothly. But international software have the window names in their native language. These names are hard to capture and so are the commands that do not take the shortcut keys made for English Unicode. In this scenario, we used AutoIt Window Info feature that captured the entire window information. More sophisticated control commands like ControlSend are more reliable than rudimentary commands like Send.

Few installations took long time to complete, for example, in case of restarting the machine. In such case, the user may not be sure if the installation is complete or still in progress. The user intervention at this point can hinder the execution of au3 script. To avoid this confusion, we

came up with two ways. First, to freeze the Windows Desktop until the script finishes. Second, provide a status bar showing the progress of the script. We modified our scripts with the second option.

The scripting phase of the research dealt more with creating scripts that could not only automate the process of installation but also handle any type of exception that occurred while installing the software. Many of these complicated scripts that managed the intricate requirements of the software have been found to be very useful and re-used in several installation scripts. We started with simple scripts that made simple installations and then moved to writing erudite scripts that made complex installations. We also maintained a detailed documentation containing the installation notes for each ISO. The following section will deal with the detailed analysis of the research.

3. Detailed Analysis

Having created ample amount of scripts, we reviewed all the scripts to analyze what made the automation of the install procedure a hard task and what was done to make the process easy so that the user can read, understand and use the data in the future. We also tested each script on the Emulation Assistant and the evaluation is discussed in the Section IV. To address the concerns about the automation of the installation, we discuss the following aspects in this section:

a) Characteristics of ISOs
b) Software Dependence
c) Common Scripts and ISOs
d) Non-working/Bad ISOs
e) Cost in terms of Time
f) New Configuration
g) New Storage

a) Characteristics of ISOs

The information on the digital object is what that makes the digital object distinctive. The information can be in different languages, can belong to a different genre or can be government/commercial. A digital object that belong to a Cultural genre will have video, audio and text file, where as a digital object that has Geological Survey information will have graphics that show maps, charts and tables. An object written in particular language is more relevant to set of population that uses that language. An object published during early 1990s may have different software, hardware and storage media when compared to the objects published currently. Depending on its characteristics, each digital object will have different software that is developed to create and access the data and a different program to run this software. Thus, the installation of no two ISOs is same. We showcase different characteristics of ISOs in Table 1. We have tried to include all possible characteristics that we came across during the automation of the install procedure.

Category	Genre	Language	Publication date
Commercial Government	Academic Astronomical Biography Cultural Database Educational Entertainment Geological Historical Informational Periodical Recreational	Chinese (PRC) Chinese (Taiwan) Czech English German Hungarian Japanese Korean Polish Spanish	1990 -2008

Table 1: Characteristics of ISOs**

b) Additional Software Dependence

Since ISOs have media and text files and files in many other formats, these ISOs become dependent on the software that support these files. As discussed earlier in Exploring the ISO section, finding additional software has been the major challenge in the research, especially in cases where a proper documentation was not available. Few ISOs had the additional software included in the package. In certain cases, the additional software was quite evident from the extensions of the files. For some ISOs, the error messages gave the hint about the type of software required. At times, we substituted software with its equivalent software. For example, we used Adobe reader in place of Abapi reader. But not all the ISOs were of obvious case and not all the error messages could be easily deciphered. In cases where we found the software type, which version of software to be used was the next big question and we handled this by trial and error method. And without the additional software it would be difficult to render the digital data in its correct and authentic form. Table 2 includes various additional software that were installed as per the dependence of the ISO.

Additional Software Name	**Version**
Adobe Reader	3.0, 4.0, 5.0, 6.0, X 10.0
QuickTime	2.0.3, 3.0, 4.0, 5.0, 7.6.9, Browser Plug-In
Multimedia Tool Box	mtb30
Internet Explorer	8.0.6001.18702

Microsoft Office	97, 2000, 2010
Real Audio	5.0.0.97

Table 2: Additional Software details**

The most commonly used additional software were Adobe Reader and QuickTime. To choose a right version was not a difficult task as it is backward compatible. But to run the .mov (media) files, it was important that we choose the appropriate version of QuickTime. QuickTime is not backward compatible and a media file that ran on a particular version did not run on other versions. Multimedia Tool Box (mtb30) is a media tool that is rarely used and is very difficult to find. All the html files opened accurately on the default IE set up on Virtual Machine. Since Microsoft Office too is backward compatible, the latest version managed to open all the .doc(x) or .dot(x) files. But few programs were very dependent on the Office version like 97 and 2000, and actually checked if the version was available on the operating system. If they did not find the required version, they simply exited the installation.

c) Common Scripts and ISOs

Many ISOs require similar version of additional software, similar system settings such as freeing virtual memory, language change and so on. We created separate scripts that handled these special cases and embedded these scripts into the actual install scripts. We created ISO format of additional software and used them whenever they were needed. Re-use of these ISOs and scripts significantly reduced the time and load and increased the efficiency in scripting, we believe that it is necessary to maintain a repository that will store these scripts and ISOs and can be used anytime in future for creating automation scripts.

d) Non-Working/Bad ISOs

Bad ISOs were the one that did not work or had data missing on them. The ISOs were bad because they were truncated while converting the CD-ROM to ISO image or they had specific additional software requirements or they needed Windows operating system that operated in DOS mode or there is a possibility that the can be errors while creating the actual software and the software might have not run at all. The truncated ISOs have to be created again. A particular ISO needed MS Word 97 or 2000, the issue can be resolved by providing the MS Office 97 or 2000 but Office is not a freeware tool and providing this on the baseline will cause legal concerns. Also we have a default Office 2010 set up on our baseline; hence it will be difficult to have two versions of Office on Virtual Machine at the same time. This issue is yet to be addressed. Few error messages that still remain to be handled are:

1) Error while accessing the registry reinstall or repair Windows
2) NTVDM CPU error: illegal instruction

3) C:\Windows not found
4) Needs Windows 3.0 or 3.1
5) Error while parsing CGI arguments

Error message 1 can be due to password protection enabled on a .doc file and the ISO does not have any information on the password.

The bad ISOs took significant amount of time to analyze the cause behind their improper behavior. Despite every attempt to resolve the issues, we still have few cases that need to be worked upon.

e) Cost in Time

Time can be the best measurement for the effort and cost involved in scripting. Other factors like VMware, AutoIt and Virtual Machine set up are one-time cost. VMware Server and AutoIt are freeware tools and we need a Windows XP CD to create the Virtual Machine. Many of the additional software are available for free. We monitored the amount of time taken for each script and recorded the same. Initially, writing the scripts took some time and with more ease in using AutoIt the amount of time significantly reduced for subsequent scripts. The time taken to write a script ranged from less than 5 minutes to 3 hours. All the autorun scripts took less than 5 minutes; installations in International language took more time. On an average, most scripts were written in 20-30 minutes. If the script took more than 30 minutes of time, then this means that we must have encountered few issues while creating the scripts. The following are the possible reasons behind long time:

a) Installation needed specific environment settings to be changed. For example, freeing virtual memory, changing compatibility settings
b) To find out the right version of additional software and installing the software.
c) The software was developed in International language. This needed to change the Regional language option on the operating system and capture the window names that were written in a foreign language.
d) README or other available documentation was in International language. We translated the documents to English to understand the steps of installation.
e) Multiple re-starts required to complete the installations.
f) Multiple programs on the ISO needed to be installed simultaneously.
g) Installations those were lengthy and complicated.
h) Trying to decipher misleading error messages

Since there is a huge repository of digital objects, it is mandatory that the amount of time spent on each ISO be as minimal as possible. We employed the following strategies to reduce the amount of time taken to write the scripts and enhance the efficiency:

a) Maintain a repository of scripts and ISOs that are frequently used and re-use them whenever needed.
b) If two ISOs have similar steps for installation, then use the same script for all those ISOs.
c) Make maximum use of the VMware facility to take Snapshots. As discussed earlier, to install East-Asian language pack on the operating system, Windows XP CD is needed. So install the pack on the Virtual Machine and take a snapshot.
d) Maintain an updated documentation of the installations that can be referred to in future to handle similar issues that arise.

Results section discusses in detail about the varying time based on different criteria through graphical notations.

f) New Configuration

After the scripting phase, we removed the AutoIt tool from the Configuration (baseline). We had to modify the configuration to make it more suitable to meet all the requirements that we explored while creating automation scripts. For multiple ISO scenarios, we added more number of drives to the Virtual Machine hardware. Since few additional software, for example Adobe Reader X, consumed more time to install, to include them on the baseline significantly reduced the scripting and installation time. Few system settings that may not obstruct the installation of other ISOs were also included on baseline to save time. Adding many components to the configuration may slow down the Virtual Machine, hence we changed the configuration with components that were mandatory and their installation made a huge difference in the installation time. The new configuration includes:

a) Three CD drives
b) Adobe Reader X
c) International language package
d) Microsoft office (debatable)

VMware limit the number of CD drives to three. To add more than three drives remains to be part of further research. The automation scripts are highly dependent on the drive location. To run these scripts accurately, place the actual ISO on Drive 1 and primary additional software (if any) on Drive 2 and secondary additional software (if any) on Drive 3.

g) New Storage

To incorporate the additional software ISOs, we also changed the layout of existing AFS storage. We added the additional ISO to the directory of the actual ISO that needed it. The present layout will have the actual ISO, intall.exe and the additional ISOs. Figure 2 gives the modified Virtual archive:

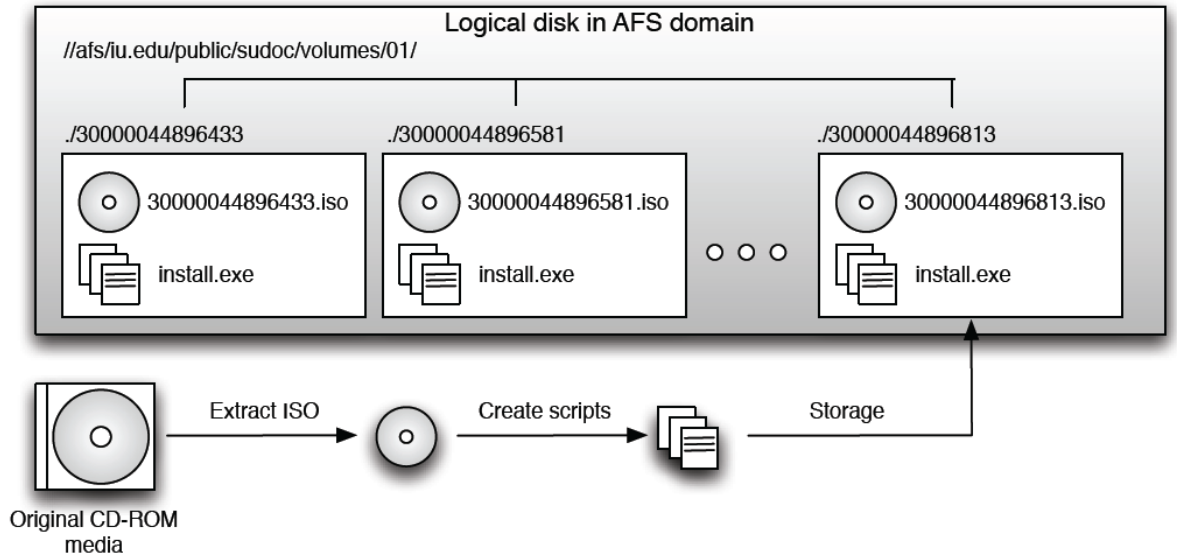

Figure 2: Modified virtual Archive (this figure needs to modified to include the ISOs)

4. Results

This section will include various graphs.

IV. Testing with Emulation Assistant

In the final stage of the research, we evaluated the emulation assistant for legacy executables by accessing the executables through emulation assistant. We had to modify emulation assistant to add multiple ISOs to multiple drives and to mount the ISOs from the virtual archive in the required order on the separate drives. The maximum number of executables ran perfectly fine on the emulation assistant automating the entire process of accessing the digital objects. Only few international ISOs were troublesome but were eventually handled.

(probably more notes on this section to be followed once we have final data ready)

*** tables may need updating*