FileMan to Caché SQL
Data Extraction and Mapping Tool

VERSION 1.0

Installation Guide

June 15, 2006

Veterans Health Administration

Office of Information

Health Data & Informatics

Revision History

Date	Revision	Description	Author(s)
6/15/2006	1.0	Documentation developed to support initial software release.	Standard Terminology Services (STS)

Contents

Revision History ii

Contents ii

Using this Guide iii

Assumptions about the Reader iii

Reference Materials iii

1. Introduction 1

1.1 Data Mapping 1

1.2 Patch and Package Installation 1

2. Installing FileMan to Caché SQL 1.0 2

2.1 General Pre-installation Information 2

2.2 M-Specific Pre-Installation Information 3

2.3 Installation Information 5

3. Glossary 13

Using this Guide

The following conventions are used in this document to indicate special information to the reader.

Symbol	Description
	Used to inform the reader of general information including references to additional reading material.
	Used to caution the reader to take special notice of critical information.

- Descriptive text is presented in a proportional font (as represented by this font).
- "Snapshots" of computer online displays (i.e., roll-and-scroll screen captures/dialogs) and computer source code are shown in a *non*-proportional font and enclosed within a box.

  User responses to online prompts will be in boldface type.

  The word "**Enter**" in snapshots further prompts the user to press the **Enter** or **Return** key on their keyboard.

  Author comments are displayed in italics or as "callout" boxes.

Assumptions about the Reader

This guide is written with the assumption that readers have experience with the following:

- VistA computing environment
- Kernel Installation and Distribution System [KIDS]
- VA FileMan data structures and terminology

This guide makes no attempt to explain how the overall VistA programming system is integrated and maintained. Such methods and procedures are documented elsewhere. We suggest you look at the various VA home pages on the World Wide Web (WWW) for a general orientation to VistA. For example, go to the Health System Design & Development (HSD&D) Home Page at the following web address: [redacted]

Reference Materials

Readers who wish to learn more should consult the following:

- Data Standardization Project Intranet Website: [redacted]
- The VistA documentation library has more detailed information about all aspects of VistA. Readers may be especially interested in documentation about the MFS, Kernel and Kernel Toolkit patches, which are involved in the Data Standardization process: [redacted]
- Caché is a product of InterSystems Corporation. More information about their product is available on their Website, http://www.intersystems.com/.

- Data Standardization is supported by VHA Directive 2005-044. A copy of this directive is available from this Website, http://www1.va.gov/vhapublications/.

Documentation is made available online, on paper and in Adobe Acrobat Portable Document Format (.PDF). A .PDF must be read using the Adobe Acrobat Reader (i.e., ACROREAD.EXE), which is freely distributed by Adobe Systems Incorporated at the following URL or Web address: http://www.adobe.com/

For more information on the use of the Adobe Acrobat Reader, please refer to the "Adobe Acrobat Quick Guide" also available at the Adobe URL above.

DISCLAIMER: The appearance of external hyperlink references in this manual does not constitute endorsement by the Department of Veterans Health Administration (VHA) of this Web site or the information, products or services contained therein. The VHA does not exercise any editorial control over the information you may find at these locations. Such links are provided and are consistent with the stated purpose of this VHA Intranet Service.

1. Introduction

This guide offers advice and instructions regarding the installation of the FileMan to Caché SQL Mapping Tool. This document is intended to assist Information Resources Management (IRM) and Enterprise VistA Support (EVS).

This document provides a general overview of the data mapping process. This document provides a general overview of the data mapping process. This document does not attempt to provide a general overview of functionality in either FileMan or Caché, and expects that the reader has a working knowledge of these programs.

This FileMan to Caché SQL Mapping Tool has been designed to allow developers to quickly and easily create custom Caché SQL mappings for their FileMan Files. The tool relies on existing functionality available in both FileMan and Caché, and automates the creation of Caché classes that are mapped to target FileMan files. The data remains unaltered in physical storage on the FileMan system. The tool makes this data available via Caché classes, which provide support for ODBC calls and SQL queries, among other options.

The tool has been designed to be lightweight and flexible and to have few dependencies outside of Caché and FileMan. It has been developed as a standard FileMan patch, and is released as a normal KIDS build.

This tool is being released in support of Data Standardization efforts. Data Standardization is supported by VHA Directive 2005-044, and will enable extraction of existing VistA reference data so that it can be thoroughly analyzed.

1.1 Data Mapping

The data mapping process involves two steps: FileMan file discovery, and Caché Class creation. File discovery will read a target FileMan file and determine the file's name, fields, and datatypes. The discovery information recorded for these files includes enough information to create a corresponding Caché Class, its Methods, Properties and Triggers. This discovery record, or "map," identifies the number and structure of the Caché Classes.

When all target files are mapped, Caché classes are created.

The user can control the level of detail added to the Caché classes with a series of flags; for example, the user can control whether Sub-Classes are created for each Multiple or Word-Processing Field. These flags are described in more detail elsewhere in this document and in the Developer Manual.

This tool can map any FileMan file and it's sub-files. The user can choose to map just one, many, or all FileMan files in a specified namespace.

1.2 Patch and Package Installation

Installation will install one package: "FM TO CACHE SQL" (CASH) Version 1.0.

2. Installing FileMan to Caché SQL 1.0

2.1 General Pre-installation Information

Installation is expected to take 5 minutes or less. This estimate includes time to load the software, and to run the post-installation procedure, POST^CASHI, to completion.

The Maximum Memory Per Process (KB) setting should be at least 1024 (1MB) for this tool.

The minimum recommended requirement for Caché SQL is at least 2048 (2MB), but I have found 1MB to be enough for most purposes. If you plan to use Web Services in Caché 5.0+, then you will almost certainly need 2MB.

2.1.1 Required Packages/Patches

The following patches must be installed prior to installation of the data mapping toolset.

Software	Version	Patch Information
Caché	4.1 or higher	Standard installation.
VA FileMan	22.0	Fully patched.
Kernel	8.0	Fully patched.
Kernel Toolkit	7.3	Fully patched.
MailMan	8.0	Fully patched.

2.1.2 Documentation Retrieval

Download the documentation from an FTP Server. The preferred method is to "FTP" the files from ftp://download.vista.med.va.gov (Internal) or ftp://ftp.va.gov/vista (FOIA).

1. Download the documentation files:
   - CASH_v_1_Installation.pdf
   - CASH_v_1_Technical.pdf
   - CASH_v_1_Developer.pdf
   - CASH_v_1_User.pdf.

   These are all BINARY files.

2.1.3 Software Retrieval

Download the software from an FTP Server. The preferred method is to "FTP" the files from ftp://download.vista.med.va.gov (Internal) or ftp://ftp.va.gov/vista (FOIA).

1. Download the Host File CASH1_0.KID.
   (As an alternative you may prefer to download CASH1_0.RO and CASH1_0.GOGEN to do a manual installation)
   These are all ASCII files.
2. Move the files to the appropriate directory on your system.

2.1.4 Hardware and Operating System Requirements

The Maximum Memory Per Process (KB) setting should be at least 1024 (1MB).

As discussed in section 2.1 above, you may wish to increase this to 2048 (2MB), the minimum recommended requirement for Caché SQL, especially if planning to use Web Services.

The amount of disk space required will vary based on the number of FileMan files that are mapped to Caché Classes, and also the Caché version. The following table shows disk space usage for a test system containing 7400 FileMan files, which you can use as a rough guide.

Table 1: Disk Usage for 7400 FileMan Files Mapped to Caché Classes

	Cache.dat Size Before		**Cache.dat Size After**	
Caché Version	**Size (MB)**	**Percent Full**	**Size (MB)**	**Percent Full**
4.1.16	900	85%	5,000	98%
5.0.12	900	85%	1,700	92%

2.1.5 System Performance Capacity

Once installation is complete, there should be no significant changes in the performance capacity of the operating system. Installation of the tool is quick and should have little impact on the system. Performance may be impacted when the mapping tool is mapping files and compiling Caché classes though, especially if the ALL^CASH() API is used. It is recommended that mapping is done outside of peak business hours.

Use of the resulting Caché Classes and Tables should be carefully planned and monitored. Large, unregulated SQL queries will affect system performance and network bandwidth. Some of the APIs in this tool will delete and re-create Caché Class objects, which will affect users relying on those objects to complete queries.

2.2 M-Specific Pre-Installation Information

2.2.1 IRM Staff

Programmer access is required for installation.

2.2.2 Software Installation Time

Installation of the FileMan to Caché SQL tool will take approximately 5 minutes, or less.

This estimate includes time to load the installation distribution and to run post-installation routines to completion.

There is no need to start or stop any services in preparation for installation.

2.2.3 Users on the System

Users may remain on the system during installation.

2.2.4 New Namespaces

The FileMan to Caché SQL tool has been assigned the CASH namespace.

2.2.5 Routine List

The following routines are included in the CASH Package.

Routine Name	Routine Name	Routine Name	Routine Name
CASH*	CASHCN	CASHFN12	CASHUT02
CASH0	CASHCU	CASHFN13	CASHV4C
CASH1	CASHD	CASHFN14	CASHV4C0
CASH2	CASHDT01	CASHFN18	CASHV4C1
CASH3	CASHDT02	CASHFN19	CASHV4C2
CASHC	CASHDT03	CASHI	CASHV4C3
CASHC0	CASHDT04	CASHN	CASHV4C4
CASHC1	CASHDT05	CASHR	CASHV4D
CASHC2	CASHDT06	CASHR0	CASHV4UT
CASHC3	CASHDT07	CASHU	
CASHC4	CASHDT08	CASHUT	
CASHC5	CASHFN11	CASHUT01	

2.2.6 File and Global Information

The following table shows information about files and globals for the CASH package:

Number	Name	DD	RD	WR	DEL	LAYGO	AUDIT
15050.11	CASH FM CLASS MAP FILE	@	@	@	@	@	@
15050.12	CASH CUSTOM DATATYPES FILE	@	@	@	@	@	@
15050.13	CASH ERRORS FILE	@	@	@	@	@	@
15050.14	CASH PARAMETER FILE	@	@	@	@	@	@
15050.18	CASH LOCKDOWN FILE	@	@	@	@	@	@
15050.19	CASH SQL RESERVED WORDS FILE	@	@	@	@	@	@

2.2.7 Translation

There are no translation requirements.

Some sites may have a spurious translation for "%*" globals to "%CACHESYS", which MUST be removed. This translation was correct, though unnecessary for Caché 3.1 (as Caché handles "%*" globals internally), but is definitely wrong for Caché 4.1 or higher. With the introduction of the "%CACHELIB" database, Caché MUST be allowed to place the "%*" globals itself, or Caché SQL will not function, and this patch will error.

2.2.8 Journaling

No file requires journaling.

Use of this tool can have an impact on journaling. Creating 5315 Classes using this tool on a test system created about 3 gigabytes of journal files.

You may want to stop journaling ^oddDEF and ^oddCOM if this is a problem, and perhaps ^CASH.

2.2.9 Protection

The following table details the global protection that should be set on the M side for the CASH package:

Global Name	**Protection**	
	DSM for Open VMS	**Caché**
^CASH	System: RWP World: RW Group: RW User: RW	System: RWD World: N Group: N User: RWD

2.2.10 Mail Groups

MailMan mail groups are not created as part of the installation.

2.3 Installation Information

There is no need to start/stop any services prior to beginning this portion of the installation.

If you are re-installing this product on any Caché Version 4.1 system, and you have thousands of Classes which were already been built using this tool, the CASH.FileMan.* Datatypes can take a long time to compile! This is probably because Caché is building a list of dependencies. See Appendix A of the Developer Guide for more details.

1. Locate the Kernel Installation and Distribution System (KIDS) host file: CASH1_0.KID.
2. Transfer the KIDS host file to the appropriate M system(s) for installation. The .KID file is an ASCII file.
3. From the KIDS Menu, select the "Installation" option.
4. Select the KIDS Installation Menu option, "Load a Distribution," entering **CASH1_0.KID** as the name of the host file.
5. Use the KIDS Installation Menu option, "Install Package" to install the CASH package. Use **FM TO CACHE SQL 1.0** as the name to install.
6. Answer the following prompts as shown:

```
Want KIDS to INHIBIT LOGONs during the install? YES// NO
Want to DISABLE Scheduled Options, Menu Options, and Protocols? YES// NO
Enter the Device you want to print the Install messages.
You can queue the install by enter a 'Q' at the device prompt.
Enter a '^' to abort the install.
DEVICE: HOME// <Enter>  CONSOLE
```

7. Once the installation completes, a post installation routine, POST^CASHI will run automatically to do the following:
   a. Initialize file #15050.19 (CASH SQL RESERVED WORDS) with a list of reserved words taken from the Caché documentation.
   b. Remove any old error log data from file #15050.13 (CASH ERRORS)
   c. Create and Compile the Utility Classes: CASH.FMFile, CASH.FMField and CASH.Utility.
   d. Initialize file #15050.12 (CASH CUSTOM DATATYPES).
   e. Create and Compile the Custom Datatype Classes (CASH.FileMan.*) .
   f. Run ALL^CASH("",HDIS","","HDISUser") to create all Caché Classes for files in the Data Standardization controlled list. "HDIS" generates the Classes in the HDIS Package, and generates tables in the HDIS schema (e.g. HDIS.GmrAllergies). "HDISUser" sets the Package owner to this SQL group. See the Technical Manual for more information about ALL^CASH.

      Note: If for some reason the Custom Datatypes were not created and/or file 15150.19 SQL RESERVED WORD FILE was not created, you should re-run POST^CASHI manually.
8. Enable the Caché account, HDISUser, and set a new password for it.

2.3.1 KIDS Installation Example

Following is a capture of a KIDS installation:

```
D ^XUP

Setting up programmer environment
This is a TEST account.

Terminal Type set to: C-VT320

Select OPTION NAME: XPD MAIN       Kernel Installation & Distribution System

          Edits and Distribution ...
          Utilities ...
          Installation ...
          Patch Monitor Main Menu ...

Select Kernel Installation & Distribution System Option: Installation

   1      Load a Distribution
   2      Verify Checksums in Transport Global
   3      Print Transport Global
   4      Compare Transport Global to Current System
   5      Backup a Transport Global
```

```
   6       Install Package(s)
           Restart Install of Package(s)
           Unload a Distribution

Select Installation Option: 1  Load a Distribution
Enter a Host File: CASH1_0.KID

KIDS Distribution saved on May 15, 2006@22:50:46
Comment: FM To Cache SQL Mapping Tool - Version 1.0 Build 8

This Distribution contains Transport Globals for the following Package(s):
   FM TO CACHE SQL 1.0
Distribution OK!

Want to Continue with Load? YES// <Enter>
Loading Distribution...

   FM TO CACHE SQL 1.0
Use INSTALL NAME: FM TO CACHE SQL 1.0 to install this Distribution.


   1       Load a Distribution
   2       Verify Checksums in Transport Global
   3       Print Transport Global
   4       Compare Transport Global to Current System
   5       Backup a Transport Global
   6       Install Package(s)
           Restart Install of Package(s)
           Unload a Distribution

Select Installation Option: 6  Install Package(s)
Select INSTALL NAME: FM TO CACHE SQL 1.0       Loaded from Distribution     Loaded
from Distribution  5/18/06@15:36:57
     => FM To Cache SQL Mapping Tool - Version 1.0 Build 8  ;Created on May 15

This Distribution was loaded on May 18, 2006@15:36:57 with header of
   FM To Cache SQL Mapping Tool - Version 1.0 Build 8  ;Created on May 15,
2006@22:50:46
   It consisted of the following Install(s):
FM TO CACHE SQL 1.0
Checking Install for Package FM TO CACHE SQL 1.0

Install Questions for FM TO CACHE SQL 1.0

Incoming Files:

   15050.11  CASH FM CLASS MAP

   15050.12  CASH CUSTOM DATATYPES

   15050.13  CASH ERRORS

   15050.14  CASH PARAMETER FILE  (including data)

   15050.19  CASH SQL RESERVED WORDS

Want KIDS to INHIBIT LOGONs during the install? YES// NO
Want to DISABLE Scheduled Options, Menu Options, and Protocols? YES// NO

Enter the Device you want to print the Install messages.
You can queue the install by enter a 'Q' at the device prompt.
Enter a '^' to abort the install.
```

```
DEVICE: HOME// <Enter>  CONSOLE
Complete                              FM TO CACHE SQL 1.0

 Install Started for FM TO CACHE SQL 1.0 :
               May 18, 2006@15:37:09

Build Distribution Date: May 15, 2006

 Installing Routines: 100%
               May 18, 2006@15:37:09

 Running Pre-Install Routine: CLEAN^CASHI

 Installing Data Dictionaries: 100%
               May 18, 2006@15:37:09

 Installing Data:
               May 18, 2006@15:37:09

 Running Post-Install Routine: POST^CASHI


Initializing SQL Reserved Words File (#15050.19), if necessary.

%AFTERHAVING added
%ALPHAUP added
```

Note: 269 total terms will be added. Each term added to file 15050.19 will be listed for the user, but they have not been listed here for space considerations.

```
WRITE added
YEAR added

Removing old Error Log data (#15050.13)

Create and compile the CASH Utility Classes.

Deleting class CASH.FMField                                          done.
Deleting class CASH.FMFile                                           done.
Compilation started on 05/18/2006 15:37:10
Compiling class CASH.FMFile ..................
Compiling class CASH.FMField ..................
Compiling table CASH.FMField ...
Compiling table CASH.FMFile ...
Compiling routine CASH.FMFile.1
Compiling routine CASH.FMField.1
Compilation finished successfully.

Deleting class CASH.Utility                                          done.
Compilation started on 05/18/2006 15:37:11
Compiling class CASH.Utility ..................
Compiling routine CASH.Utility.1
Compilation finished successfully.


Create and compile the latest CASH.FileMan Datatypes.

Deleting class CASH.FileMan.Date                                     done.
Compilation started on 05/18/2006 15:37:11
Compiling class CASH.FileMan.Date .....
Compiling routine: CASH.FileMan.Date.G1.MAC.............
Compiling routine CASH.FileMan.Date.1
Compilation finished successfully.
```

```
Deleting class CASH.FileMan.DateTime                                    done.
Compilation started on 05/18/2006 15:37:11
Compiling class CASH.FileMan.DateTime .....
Compiling routine: CASH.FileMan.DateTime.G1.MAC............
Compiling routine CASH.FileMan.DateTime.1
Compilation finished successfully.

Deleting class CASH.FileMan.Numeric                                     done.
Compilation started on 05/18/2006 15:37:11
Compiling class CASH.FileMan.Numeric .....
Compiling routine: CASH.FileMan.Numeric.G1.MAC............
Compiling routine CASH.FileMan.Numeric.1
Compilation finished successfully.

Deleting class CASH.FileMan.Pointer                                     done.
Compilation started on 05/18/2006 15:37:11
Compiling class CASH.FileMan.Pointer .....
Compiling routine: CASH.FileMan.Pointer.G1.MAC............
Compiling routine CASH.FileMan.Pointer.1
Compilation finished successfully.

Deleting class CASH.FileMan.SetOfCodes                                  done.
Compilation started on 05/18/2006 15:37:11
Compiling class CASH.FileMan.SetOfCodes .....
Compiling routine: CASH.FileMan.SetOfCodes.G1.MAC............
Compiling routine CASH.FileMan.SetOfCodes.1
Compilation finished successfully.

Deleting class CASH.FileMan.String                                      done.
Compilation started on 05/18/2006 15:37:11
Compiling class CASH.FileMan.String .....
Compiling routine: CASH.FileMan.String.G1.MAC............
Compiling routine CASH.FileMan.String.1
Compilation finished successfully.

Deleting class CASH.FileMan.StringDateTime                              done.
Compilation started on 05/18/2006 15:37:11
Compiling class CASH.FileMan.StringDateTime .....
Compiling routine: CASH.FileMan.StringDateTime.G1.MAC............
Compiling routine CASH.FileMan.StringDateTime.1
Compilation finished successfully.

Deleting class CASH.FileMan.VariablePointer                             done.
Compilation started on 05/18/2006 15:37:11
Compiling class CASH.FileMan.VariablePointer .....
Compiling routine: CASH.FileMan.VariablePointer.G1.MAC............
Compiling routine CASH.FileMan.VariablePointer.1
Compilation finished successfully.


 Updating Routine file... 100%

 Updating KIDS files... 100%

 FM TO CACHE SQL 1.0 Installed.
              May 18, 2006@15:37:11

 Install Message sent #3021

Install Completed
```

You are now ready to create Classes!

2.3.2 Manual Installation Example

Alternatively you can install this tool without KIDS, by using %RI to restore the routines in the CASH1_0.RO file, and %GIGEN to restore the globals in CASH1_0.GOGEN.

You MUST then run POST^CASHI to complete the post-installation tasks.

Following is a capture of a manual installation:

2.3.2.1 Install the routines

```
D ^%RI

Input routines from Sequential
Device: CASH1_0.RO   Parameters: "RS"=>

File written by Cache for Windows NT using %RO on 15 May 2006  10:50 PM
with extension INT and with description:
FM To Cache SQL Mapping Tool - Version 1.0 Build 8

( All Select Enter List Quit )

Routine Input Option: All Routines

If a selected routine has the same name as one already on file,
shall it replace the one on file? No => Yes
Recompile? Yes => Yes
Display Syntax Errors? Yes => Yes

^ indicates routines which will replace those now on file.
@ indicates routines which have been [re]compiled.
- indicates routines which have not been filed.


CASH.INT^@       CASH0.INT^@      CASH1.INT^@      CASH2.INT^@      CASH3.INT^@
CASHC.INT^@      CASHC0.INT^@     CASHC1.INT^@     CASHC2.INT^@     CASHC3.INT^@
CASHC4.INT^@     CASHC5.INT^@     CASHCN.INT^@     CASHCU.INT^@     CASHD.INT^@
CASHDT01.INT^@   CASHDT02.INT^@   CASHDT03.INT^@   CASHDT04.INT^@   CASHDT05.INT^@
CASHDT06.INT^@   CASHDT07.INT^@   CASHDT08.INT^@   CASHF.INT^@      CASHFN11.INT^@
CASHFN12.INT^@   CASHFN13.INT^@   CASHFN14.INT^@   CASHFN19.INT^@   CASHI.INT^@
CASHN.INT^@      CASHR.INT^@      CASHR0.INT^@     CASHU.INT^@      CASHUT.INT^@
CASHUT01.INT^@   CASHUT02.INT^@   CASHV4C.INT^@    CASHV4C0.INT^@   CASHV4C1.INT^@
CASHV4C2.INT^@   CASHV4C3.INT^@   CASHV4C4.INT^@   CASHV4D.INT^@    CASHV4UT.INT^@


45 routines processed.
```

2.3.2.2 Install the globals

```
D ^%GIGEN

Device: CASH1_0.GOGEN   Parameters: "R"=>
Transfer entire set of files? Yes
Transferring files on May 15 2006 at 10:55 PM
From global ^CASH(15050.11,0
To global ^CASH(15050.11,0

OK to transfer? Yes

Transfer completed
Transferring files on May 15 2006 at 10:55 PM
From global ^CASH(15050.12,0
```

```
To global ^CASH(15050.12,0

OK to transfer? Yes

Transfer completed
```

Note: There are about 36 partial global references, only the first and last few are listed here. In Caché 5.0 you may need to confirm each reference!

```
Transferring files on May 15 2006 at 10:55 PM
From global ^DD(15050.141
To global ^DD(15050.141

OK to transfer? Yes

Transfer completed
Transferring files on May 15 2006 at 10:55 PM
From global ^DD(15050.19
To global ^DD(15050.19

OK to transfer? Yes

Transfer completed
Done for this set of files.
```

2.3.2.3 Run the post install tasks

```
D POST^CASHI

Initializing SQL Reserved Words File (#15050.19), if necessary.

%AFTERHAVING added
%ALPHAUP added
```

Note: 269 total terms will be added. Each term added to file 15050.19 will be listed for the user, but they have not been listed here for space considerations.

```
WRITE added
YEAR added

Removing old Error Log data (#15050.13)

Create and compile the CASH Utility Classes.

Deleting class CASH.FMField                                          done.
Deleting class CASH.FMFile                                           done.
Compilation started on 05/18/2006 15:19:05
Compiling class CASH.FMFile ..................
Compiling class CASH.FMField ..................
Compiling table CASH.FMField ...
Compiling table CASH.FMFile ...
Compiling routine CASH.FMFile.1
Compiling routine CASH.FMField.1
Compilation finished successfully.

Deleting class CASH.Utility                                          done.
Compilation started on 05/18/2006 15:19:06
Compiling class CASH.Utility ..................
Compiling routine CASH.Utility.1
Compilation finished successfully.


Create and compile the latest CASH.FileMan Datatypes.
```

```
Deleting class CASH.FileMan.Date                                        done.
Compilation started on 05/18/2006 15:19:06
Compiling class CASH.FileMan.Date .....
Compiling routine: CASH.FileMan.Date.G1.MAC............
Compiling routine CASH.FileMan.Date.1
Compilation finished successfully.

Deleting class CASH.FileMan.DateTime                                    done.
Compilation started on 05/18/2006 15:19:06
Compiling class CASH.FileMan.DateTime .....
Compiling routine: CASH.FileMan.DateTime.G1.MAC............
Compiling routine CASH.FileMan.DateTime.1
Compilation finished successfully.

Deleting class CASH.FileMan.Numeric                                     done.
Compilation started on 05/18/2006 15:19:06
Compiling class CASH.FileMan.Numeric .....
Compiling routine: CASH.FileMan.Numeric.G1.MAC............
Compiling routine CASH.FileMan.Numeric.1
Compilation finished successfully.

Deleting class CASH.FileMan.Pointer                                     done.
Compilation started on 05/18/2006 15:19:06
Compiling class CASH.FileMan.Pointer .....
Compiling routine: CASH.FileMan.Pointer.G1.MAC............
Compiling routine CASH.FileMan.Pointer.1
Compilation finished successfully.

Deleting class CASH.FileMan.SetOfCodes                                  done.
Compilation started on 05/18/2006 15:19:06
Compiling class CASH.FileMan.SetOfCodes .....
Compiling routine: CASH.FileMan.SetOfCodes.G1.MAC............
Compiling routine CASH.FileMan.SetOfCodes.1
Compilation finished successfully.

Deleting class CASH.FileMan.String                                      done.
Compilation started on 05/18/2006 15:19:06
Compiling class CASH.FileMan.String .....
Compiling routine: CASH.FileMan.String.G1.MAC............
Compiling routine CASH.FileMan.String.1
Compilation finished successfully.

Deleting class CASH.FileMan.StringDateTime                              done.
Compilation started on 05/18/2006 15:19:06
Compiling class CASH.FileMan.StringDateTime .....
Compiling routine: CASH.FileMan.StringDateTime.G1.MAC............
Compiling routine CASH.FileMan.StringDateTime.1
Compilation finished successfully.

Deleting class CASH.FileMan.VariablePointer                             done.
Compilation started on 05/18/2006 15:19:06
Compiling class CASH.FileMan.VariablePointer .....
Compiling routine: CASH.FileMan.VariablePointer.G1.MAC............
Compiling routine CASH.FileMan.VariablePointer.1
Compilation finished successfully.
```

You are now ready to create Classes!

3. Glossary

API	Application Programming Interface. This is the definition (calling conventions) by which one application can get services from another application.
Class **(also Sub-Class)**	A Class is a collection of Properties that describes a set of objects and the Methods (or Behaviors) that can be applied to these objects. In this context, a class is a Caché database object, analogous to a table in a relational database. A Sub-Class is a child object, related to its the parent Class through a Foreign Key. In Caché, database objects can be defined as Classes, though these objects can also be exposed as Relational Tables (accessible via standard SQL), using Caché's Unified Data Architecture.
Domain	A subset of medicine, a natural grouping of clinical acts (e.g., demographics, vital signs, laboratory, pharmacy)
DS	Data Standardization
GB	Gigabyte. Equal to 1024 megabytes.
HDI	Health Data and Informatics
HDR	Health Data Repository
IEN	Internal Entry Number, a number assigned to each entry in FileMan files.
JDBC	Java Database Connectivity. A standard API for Java programs to use to access databases.
Mapping	Mappings are a set of relationships established between two formats for data. In this context, the Mapping Tool uses the metadata provided by the FileMan data dictionary to generate Caché Classes (and their implied Relational Tables) that can access the FileMan data. Each map is a template for the Caché Class or datatype generated from it.
MB	Megabyte. Equal to 1024 kilobytes.
Method	A procedure or sub-routine associated with a Class. Can also be referred to as a Behavior. In Caché, methods can be exposed as SQL Stored Procedures as well.
Multiple	A Multiple is a FileMan Field that is stored as an array. The field is hierarchically related to a parent file, allowing storage of zero, one or many values. A Multiple is equivalent to a child table in a relational database.
ODBC	Open Database Connectivity. A standard API for accessing different database systems. An application can submit standard statements to ODBC, and ODBC will translate these to a language the database understands.
Package	For the purposes of this document, package always refers to a collection of Caché Classes.

Pointer	A Pointer is a FileMan field that links to another FileMan file. A Pointer is equivalent to a Foreign Key in a relational database.
Property	In Caché, a Property is a field or attribute in a class/table.
Standardization	The process of defining, creating, deploying, and maintaining a common terminology resource (i.e., content and services) to all current and future VHA applications.
SQL	Structured Query Language. An industry-standard language for creating, updating and querying relational database management systems. SQL is also and ISO and ANSI standard.
Terminology	Set of terms, definitions, relationships of a specialized subject area. The terms which are characterized by special reference within a discipline are called the 'terms' of the discipline, and collectively, they form the terminology, those which function in general reference over a variety of languages are simply 'words', and their totality 'the vocabulary' [Sager]. See also vocabulary.
Trigger	An action causing the automatic invocation of another action (such as a routine or procedure). A trigger "fires" when data is modified. Triggers are typically used to ensure that an update to one piece of data also updates all related data.
URL	Uniform Resource Locator - a Web page address on the World Wide Web.
VUID	VHA Unique Identifier - these are meaningless numbers that are automatically assigned to concepts, properties, and relationships in a terminology to facilitate their access and manipulation by computers.
Word Processing Field **(also WP Field)**	A Word Processing Field is a special multi-line field that can be used to store a large amount of text. WP Fields are analagous to a Memo field in a relational database management system. BMS. This data is stored in FileMan as an array (sub-file), very similar in structure to a Multiple. The main difference being that a WP Field has only one field in its sub-file, whereas a Multiple can have many.
XML	Extensible Markup Language – a text format used extensively for information exchange.