Application for magnetic tapes duplication and diagnostic

Software Design Specification

Submitted by Oleksandr Shashkov

CS 445, Fall 2008

Revision History

Version	Date	Comment	By
1.0	10/01/2008	Draft Version	Oleksandr Shashkov
2.0	10/09/2008	Reviewed usage scenario	Oleksandr Shashkov
3.0	10/10/2008	Added introduction to tape layout	Oleksandr Shashkov
4.0	10/20/2008	Complete Class Diagrams	Oleksandr Shashkov
5.0	10/28/2008	Final review	Oleksandr Shashkov

1. Introduction

The purpose of this document is to present software design specification in Object Oriented concept for an Application for magnetic tapes duplication and diagnostic. This document describes functional, behavioral and data models for software.

1.1 Core Requirements

Electronic Data Discovery (EDD) department is in need of Application for magnetic tapes duplication and diagnostic. The application should be designed for use in MS Windows environment (Windows XP, Server 2003). It should provide abilities to duplicate (clone) magnetic tapes of recent and most common technologies: LTO, DLT and SDLT. The application should be implemented in OO concept utilizing C++ or C# programming language for reuse as a core tape manipulating engine for future projects.

1.1.1 Main functionality:

1.1.1.1 Read and save (image) tape contents into tape image files for further analysis and manipulations.
1.1.1.2 Create tape layout map file in text format during the read procedure.
1.1.1.3 Write image files onto tape with optional block offset utilizing tape layout map file.
1.1.1.4 Duplicate tapes (transfer entire contents from one tape to another without modifying data layout).
1.1.1.5 Operator (user) should be able to interrupt read/write/duplicate process at any time.
1.1.1.6 Provide progress indication and I/O speed reporting

1.1.2 Provide means for tape manipulating procedures which could be utilized in data recovery, research and reverse engineering of new backup formats:

1.1.2.1 Load/Unload tape cartridges into/from drive
1.1.2.2 Lock/Unlock cartridge in the device
1.1.2.3 Eject tape cartridges
1.1.2.4 Rewind tape
1.1.2.5 Adjust tension
1.1.2.6 Detect tape and tape drive parameters
1.1.2.7 Set tape and tape drive parameters
1.1.2.8 Write file marks
1.1.2.9 Position to particular file mark, physical block, Beginning Of Data (BOD), End Of Data (EOD)
1.1.2.10 Read and display tape block contents in HEX View.

2. Usage scenario

This section provides a usage scenario for the software. It translates core requirements into use-cases.

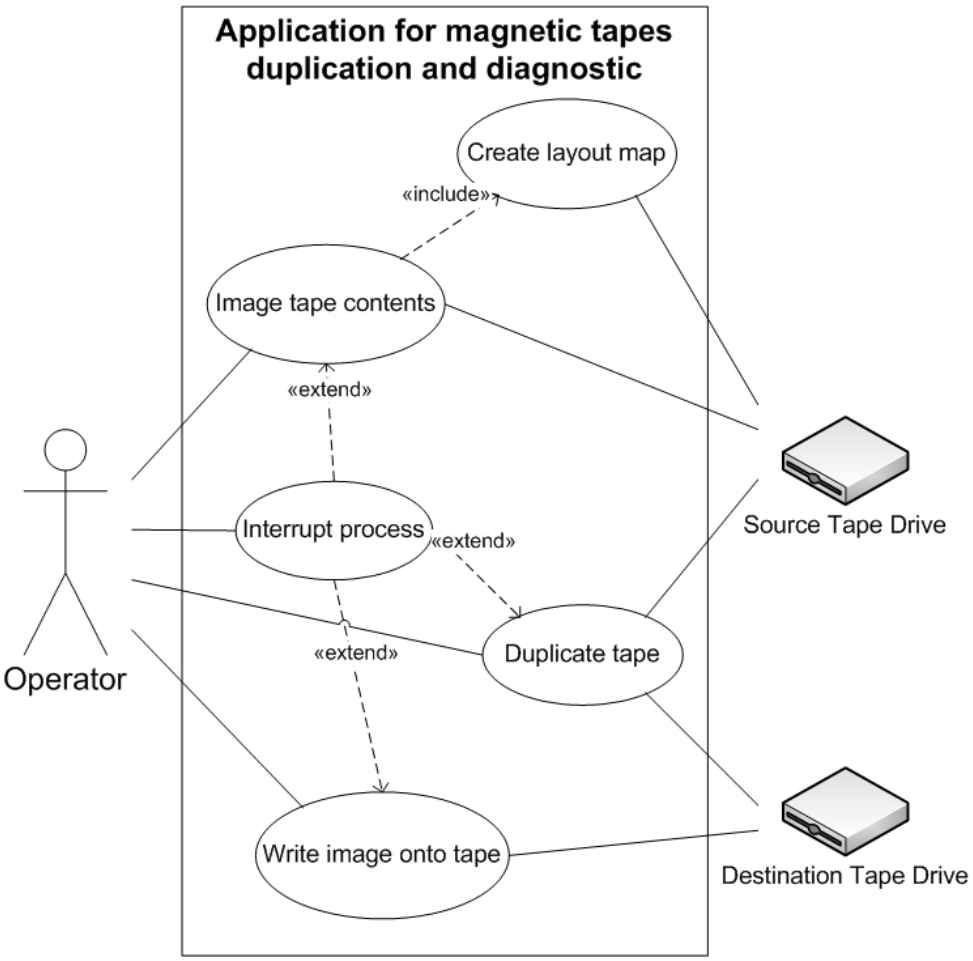
2.1 User profile

The analysis of core requirements reveals only one type of users of the system – Operator. Operator is a regular user, who will be lunching application, loading/unloading tape cartridges to/from tape drive devices, scheduling imaging and copying of the tapes and performing other diagnostic and auxiliary procedures with tape devices and cartridges. Ultimately for use case modeling system has only one primary Actor – the Operator. Two secondary actors will be considered for usage model – Source and Destination tape drives.

2.2 Use-cases

All use-cases for the software are presented below.

2.2.1 Use-Case Diagram



2.2.2 Use-Case Descriptions

Use-case: **Image tape contents**

Primary Actor:
Operator
Secondary Actor:
Source Tape Drive
Preconditions:
System is equipped with appropriate tape drive; Application is lunched.
Trigger:
Operator is in need to obtain image of a tape media.
Scenario:
1. Operator loads tape cartridge into appropriate Source Tape Drive.
2. Operator selects Source Tape Drive and destination folder to image tape to in the GUI form and then clicks “Image” button.
3. Application starts reading physical blocks from the tape and saving contents into files in destination folder.
4. Once EOD is reached on the tape, Application indicates to operator that operation is completed successfully.
5. Operator unloads tape from the Source Tape Drive.

Exceptions:
1. Input-Output errors (read from tape or write to file) during the process – stop, display error message.

Use-case: **Write image onto tape**

Primary Actor:
Operator
Secondary Actor:
Destination Tape Drive
Preconditions:
System is equipped with appropriate tape drive; Application is lunched; the source folder with tape image files and tape’s layout map file exists; Operator has a blank tape of the same type as the original tape.
Trigger:
Operator is in need to create tape duplicate from previously obtained images.
Scenario:
1. Operator loads blank tape into Destination Tape Drive.
2. Operator selects Destination Tape Drive, source folder with image files and layout map in Application’s GUI form and then clicks “Write Image” button.

3. Application starts writing image files onto destination tape in accordance with the tape layout map.
4. Once all the image files are transferred to the tape, Application indicates to operator that operation is completed successfully.
5. Operator unloads newly written tape from the Destination Tape Drive.

Exceptions:

1. The contents of source folder are inconsistent – do not start writing, display error message.
2. Input-Output errors (read from file or write to tape) during the process – stop, display error message.

Use-case: **Duplicate tape**

Primary Actor:

Operator

Secondary Actors

1. Source Tape Drive
2. Destination Tape Drive

Preconditions:

System is equipped with appropriate Source and Destination tape drives of the same type; Application is lunched; Operator has a blank tape of the same type as the original tape.

Trigger:

Operator is in need to create tape duplicate “on the fly” – from original tape.

Scenario:

1. Operator loads original tape cartridge into Source Tape Drive.
2. Operator loads blank tape into Destination Tape Drive.
3. Operator clicks “Duplicate” button.
4. Application starts reading physical blocks from the original tape and writing them to blank destination tape.
5. Once EOD is reached on the original tape, Application indicates to operator that operation is completed successfully.
6. Operator unloads original tape and its duplicate from the tape drives.

Exceptions:

1. Original tape and blank one are of different types – do not start, display error message.
2. Input-Output errors (read-write) on any tapes during the process – stop, display error message.

***Use-case:* Interrupt process (extends "Image tape contents", "Write image onto tape" and "Duplicate tape" use-cases)**

Primary Actor:
Operator
Preconditions:
Application is lunched; one of the processes extended by this use-case is running.
Trigger:
Operator decides to interrupt currently running process.
Scenario:
1. Operator clicks "Stop" button.
2. Application displays confirmation dialog box.
3. Once interruption is confirmed by Operator, Application stops currently running process. Otherwise the process continues to run.

Exceptions:
None

***Use-case:* Create layout map (included in use-case "Image tape contents")**

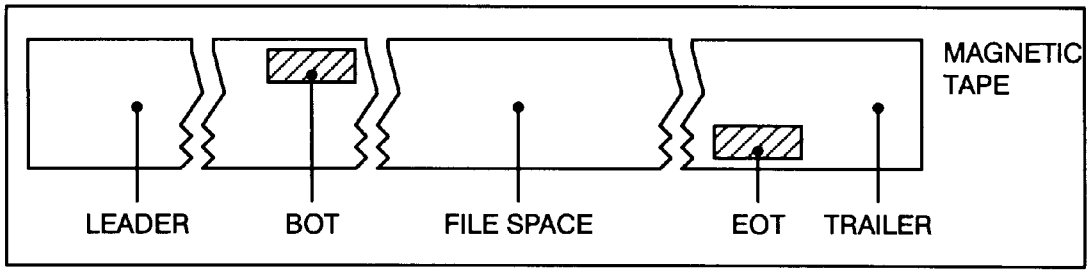
Primary Actor:
Operator
Secondary Actor:
Source Tape Drive
Preconditions:
System is equipped with appropriate tape drive; Application is lunched; Tape of imaging interest is loaded into Source Tape drive and this Drive is selected in Application GUI from; Destination folder for image files is selected in Application GUI form.
Trigger:
Operator clicks "Image" button.
Scenario:
1. Application creates empty tape's layout map text file and starts reading physical blocks from the tape and saving contents into files in destination folder.
2. While Application reads through the tape it fills in tape's layout map file with information on tape file marks locations, physical block size, its change and image file names.
3. Once EOD is reached on the tape, Application indicates to operator that operation is completed successfully.
4. Operator unloads tape from the Source Tape Drive.

Exceptions:
1. Input-Output errors (read from tape or write to file) during the process – stop, display error message.

2.3 Special usage considerations

2.3.1 Introduction to Tape Data Layout

Following diagram explains how the space is allocated for the generic tape.



LEADER – part of the tape that reading-writing device uses for pulling tape from a cartridge and hooking inside its tape-handling mechanism.

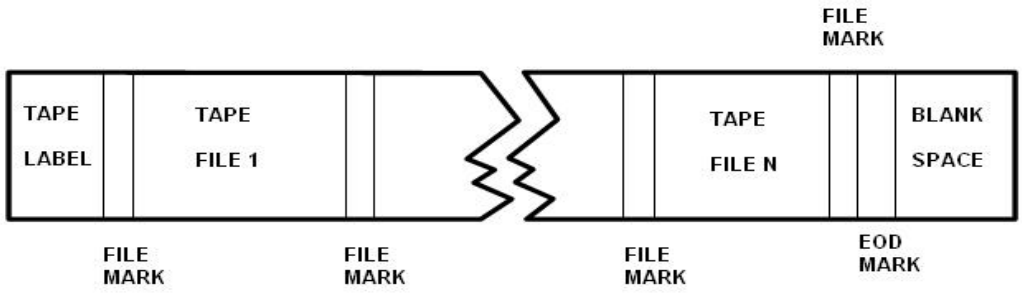
BOT – Beginning Of Tape marker indicates that this is a begginign of tape. Could be implemented as an optical or magnetic mark depending on tape storage technology.

FILE SPACE – the area in which actual data gets read and written. This area may contain backup sessions (tape files), file marks and blank areas. For more detailed explanation see diagram below.

EOT – End Of Tape mark indicates that this is the end of the tape. Could be implemented as an optical or magnetic mark depending on tape storage technology.

TRAILER – part of the tape that holds the tape inside the cartridge and serves as part of tape-handling mechanism providing it's movement.

Typical layout of the File Space:



The typical tape written in one of the common backup formats (TAR, CPIO, MTF, ARCSERVE, LEGATO ,etc) would have following structures in the file space area:

TAPE LABEL – serves as a tape ID, usually indicates what kind of backup format was used to create a tape, contains serial number, unigue label, date, physical block size, etc. From the physical layout point of view TAPE LABEL could be considered as another TAPE FILE (tape file #0).

TAPE FILE - Tape files are the results of backup sessions run by backup software, where the device is opened, written to, and closed. This is not the same as disk files since every backup session may contain many disk files embedded. The layout of data inside the backup session (inside TAPE FILE accordingly) is format dependent and is out of scope for this project. Tape may contain multiple TAPE FILES.

FILE MARK – serves as a separator between tape files. File marks are used by the device driver to indicate where one tape file ends and another begins.

EOD MARK – indicates that there is no more data on the tape after this mark. If tape capacity is not exceeded, file space may still contain blank space after the EOD mark. In order to continue writing to the tape, backup software must set tape position to EOD and continue writing new TAPE FILE from there. In this case EOD get's moved closer to the EOT.

2.3.2 Important concept - BLOCK SIZE

When data is written to tape it is written in blocks. It means that all the tape files are aligned by block size boundaries. Block size may be:

1) Fixed(constant) for the entire tape;
2) Fixed for particular tape file but different for different tape files on the same tape;
3) Variable – changes across the tape files.

Accordingly, all read and write transactions to the tape drive must be done with careful consideration of block size.

3. Static analysis and modeling: Object and Classes identification, Class Diagrams

This section identifies objects, classes and their features.

The following table contains identified objects and their brief description.

Object	Description
Application	The software that allows operator to manipulate with tapes
GuiForm	The set of Windows Forms through which Operator controls the software, schedules tasks and manipulates tapes.
Operator	The user of the Application
TapeDrive	Electronic device which allows to read and write tape media
TapeMedia	Tape cartridge containing data; it could blank or prewritten
Image	The data contents of the tape, organized in files in one destination folder and containing media layout map
MediaLayoutMap	The text file representing actual layout of the file space for particular tape (i.e. image files names, file marks locations, block size; block size change occurrences, etc.). This file gets created and filled in sequentially while original tape is imaged into disk files (use-case Image tape contents)

At this point “Application” and “Operator” objects do not seem to have any distinguishing features important for OO analysis for this particular application. “GuiForm” is the subject for GUI Design topic and will be elaborated later. Thus we will leave these 3 objects out of the scope of this section.

The following tables describe each remaining object by its attributes.

TapeMedia

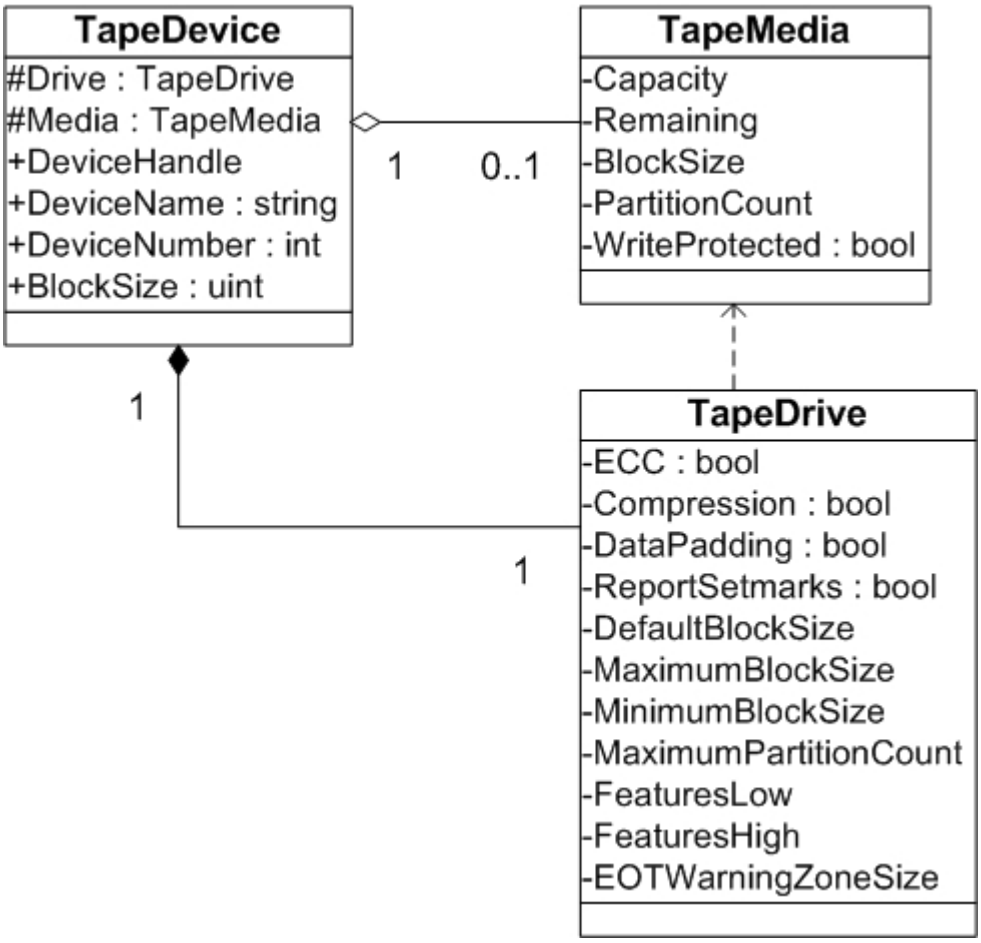
Attribute	Description
Capacity	Total number of bytes on the current tape partition
Remaining	Number of bytes between the current position and the end of the current tape partition
BlockSize	Number of bytes per block
PartitionCount	Number of partitions on the tape
WriteProtected	If this member is TRUE, the tape is write-protected. Otherwise, it is not

TapeDrive

Attribute	Description
ECC	If this member is TRUE, the device supports hardware error correction. Otherwise, it does not.
Compression	If this member is TRUE, hardware data compression is enabled. Otherwise, it is disabled.
DataPadding	If this member is TRUE, data padding is enabled. Otherwise, it is disabled. Data padding keeps the tape streaming at a constant speed.
ReportSetmarks	If this member is TRUE, setmark reporting is enabled. Otherwise, it is disabled.
DefaultBlockSize	Device's default fixed block size, in bytes.
MaximumBlockSize	Device's maximum block size, in bytes.
MinimumBlockSize	Device's minimum block size, in bytes.
MaximumPartitionCount	Maximum number of partitions that can be created on the device.
FeaturesLow	Low-order bits of the device features flag. For more details see Microsoft MSDN.
FeaturesHigh	High-order bits of the device features flag. For more details see Microsoft MSDN.
EOTWarningZoneSize	Indicates the number of bytes between the end-of-tape warning and the physical end of the tape.

Further analysis shows that we can derive class, called **TapeDevice** which includes TapeDrive and depends on TapeMedia. This class envelopes features of Drive and Media and introduces methods that could be applied to the tape storage unit. Following diagram explains relationship and features of mentioned classes:

Class Diagram: Relationship for tape storage related classes



Image

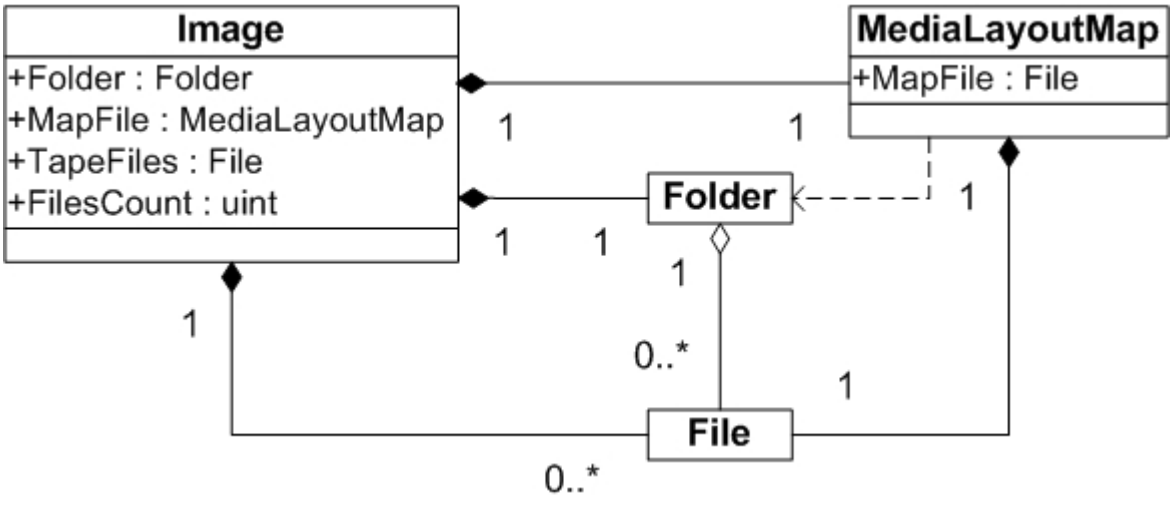
Attribute	Description
Folder	The folder containing tape files. It could be identified by full path name (i.e. C:\TapeImages\Image1)
MapFile	Text file which describes layout of file space for the tape. It represents the set of Tape Files, their names and locations of the filemarks, block size and the locations at which block size changes.
Files Count	Number of files belonging to image
TapeFiles	The set of disk files located in the Folder and containing actual tape files.

MediaLayoutMap

Attribute	Description
MapFile	The text file which describes layout of file space for the tape. It represents the set of Tape Files, their names and locations of the filemarks, block size and the locations at which block size changes.

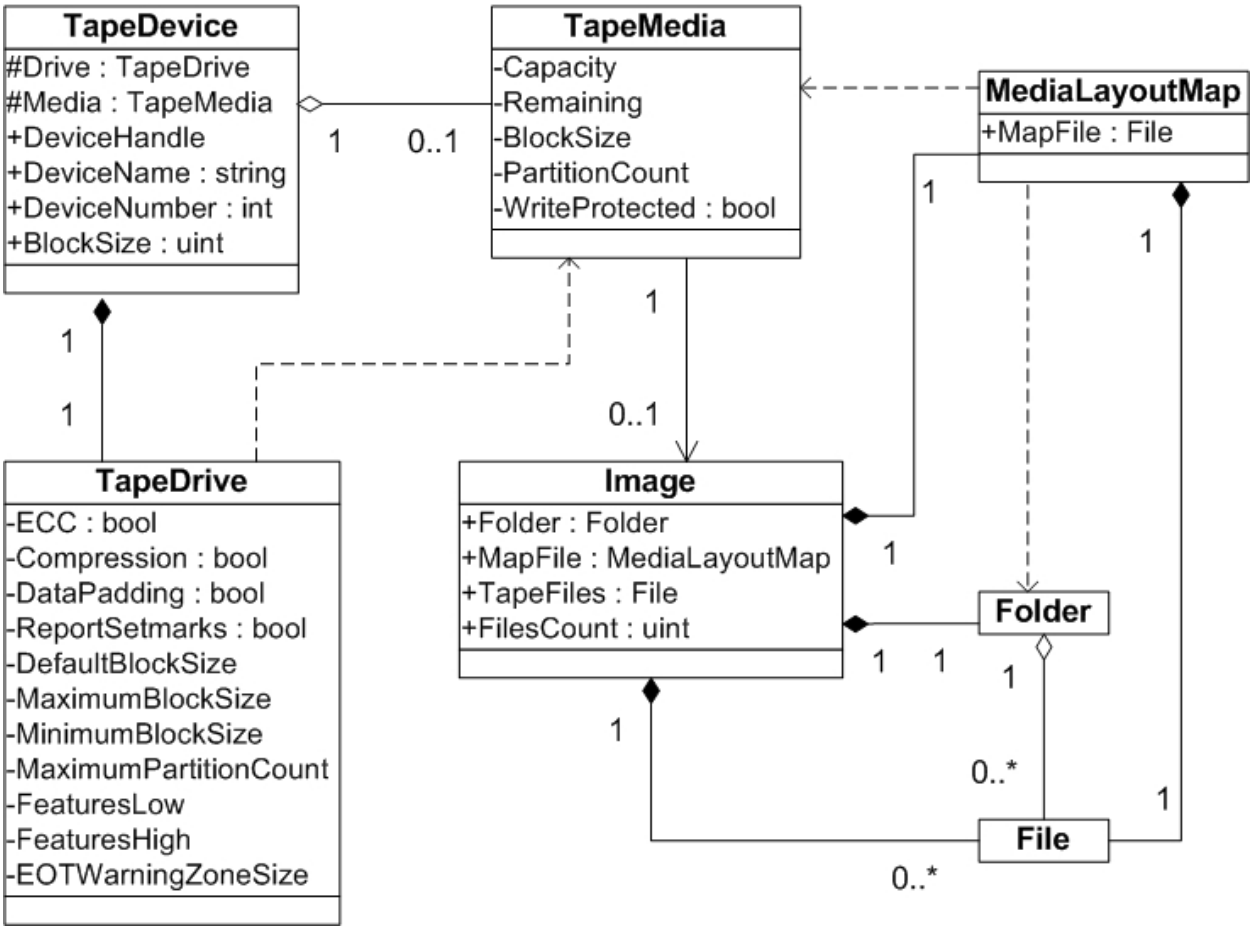
Following diagram explains relationship between Image and MediaLayoutMap classes. For better understanding model includes generic classes Folder and File which actually represent regular file system object for Windows operating system. They are not the subject of the design.

Class Diagram: relationship for Image related classes



Since “Image” class represents “TapeMedia” contents, we can complete Class Diagram as follow:

Complete Class Diagram



Medial Layout Map file format

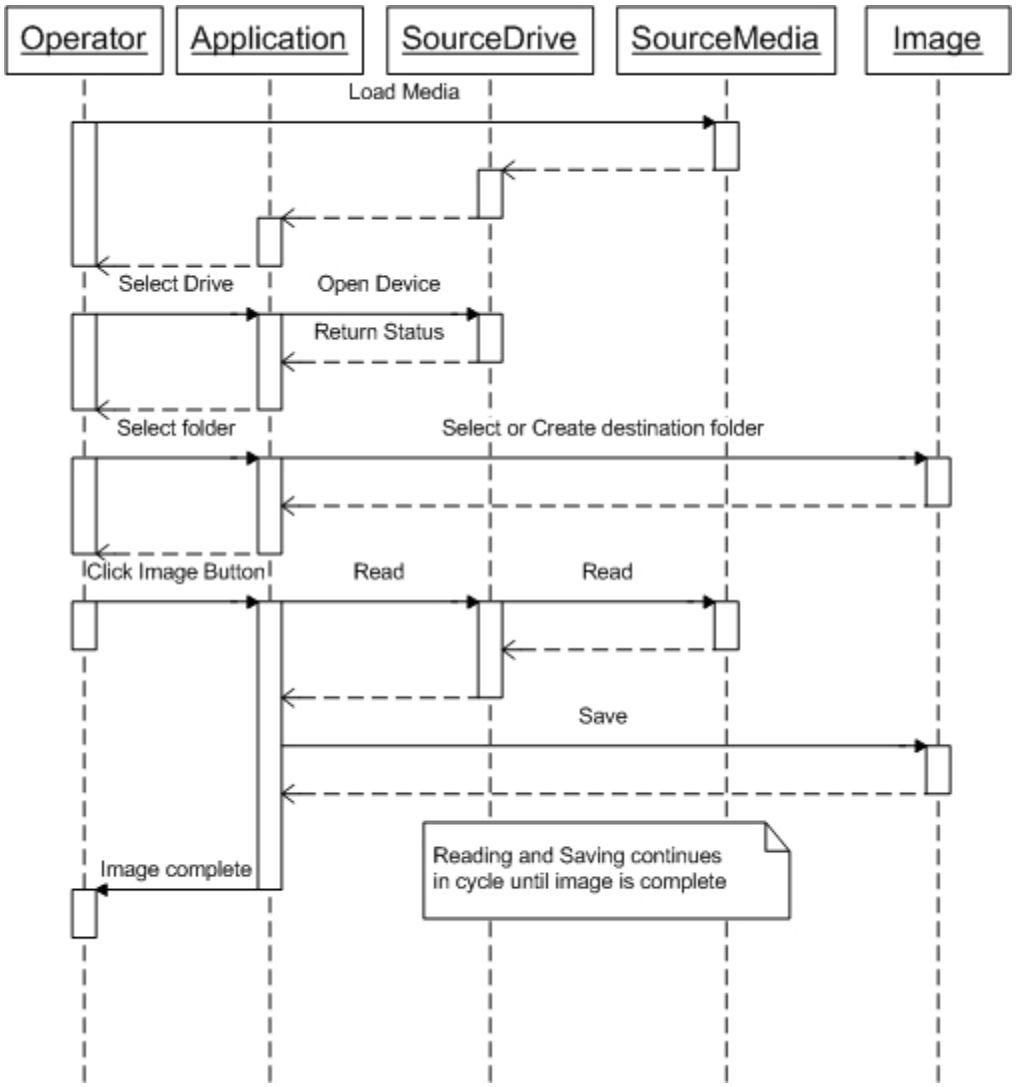
Map will be represented by simple text file with sequential records. Each record is represented by string. String could be a comment line (starts with “;”), a new tape file record, a new block size record or a filemark record. Here is the sample:

; ";" - comment line
; filename=xxxxxxx.yyy - image file name
; OfsPBA:xxxxxx BSize:yyyyyy - position where tape block size has changed
filename=file000000.dat
OfsPBA:0 BSize:524288
filename=file000001.dat
OfsPBA:0 BSize:524288
filename=file000002.dat
filemark

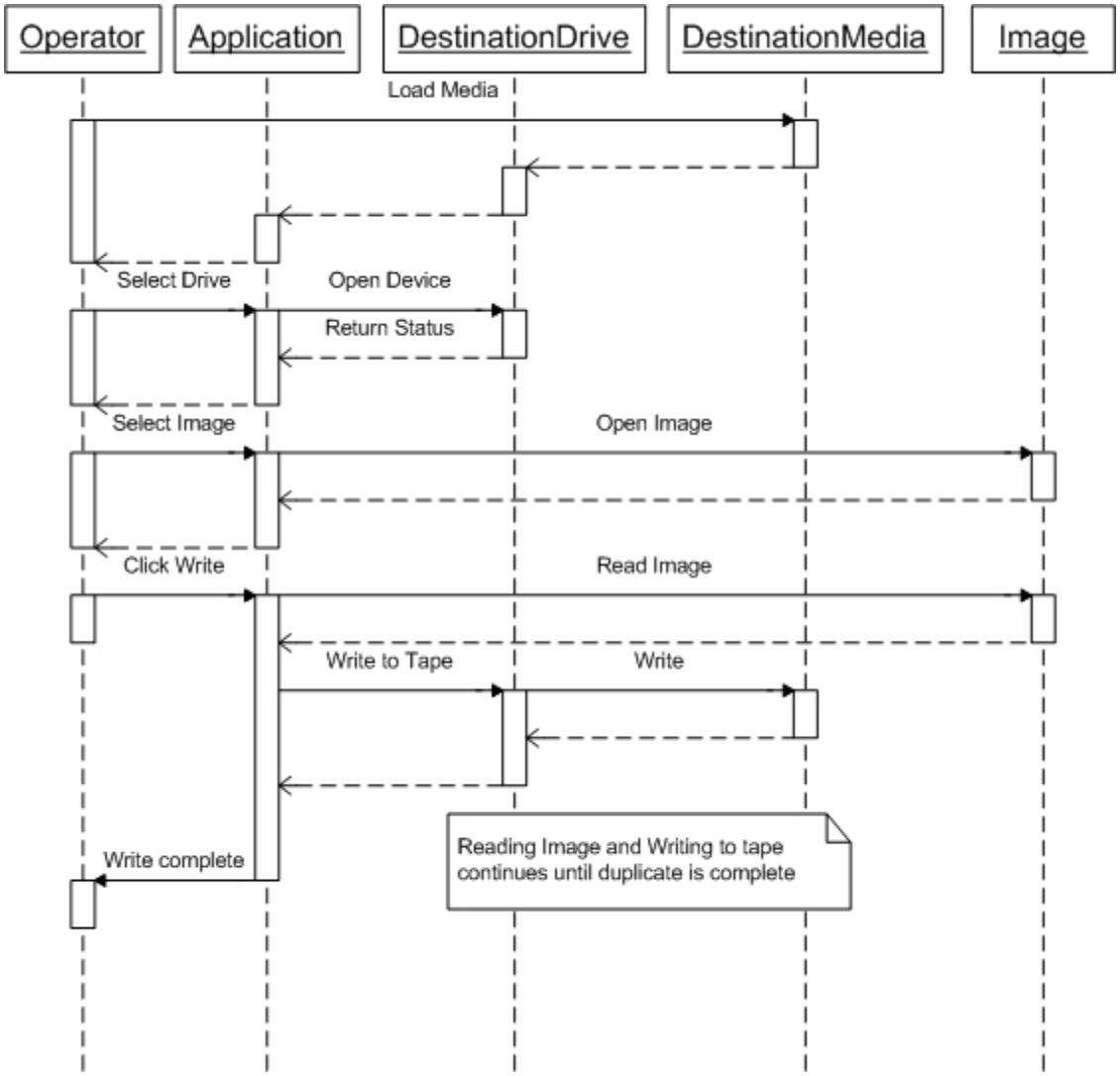
4. Behavioral analysis and modeling – sequence and state chart diagrams

Behavioral Model of the application is represented by the sequence and statechart diagrams depicting sets of major events, states and transition routes.

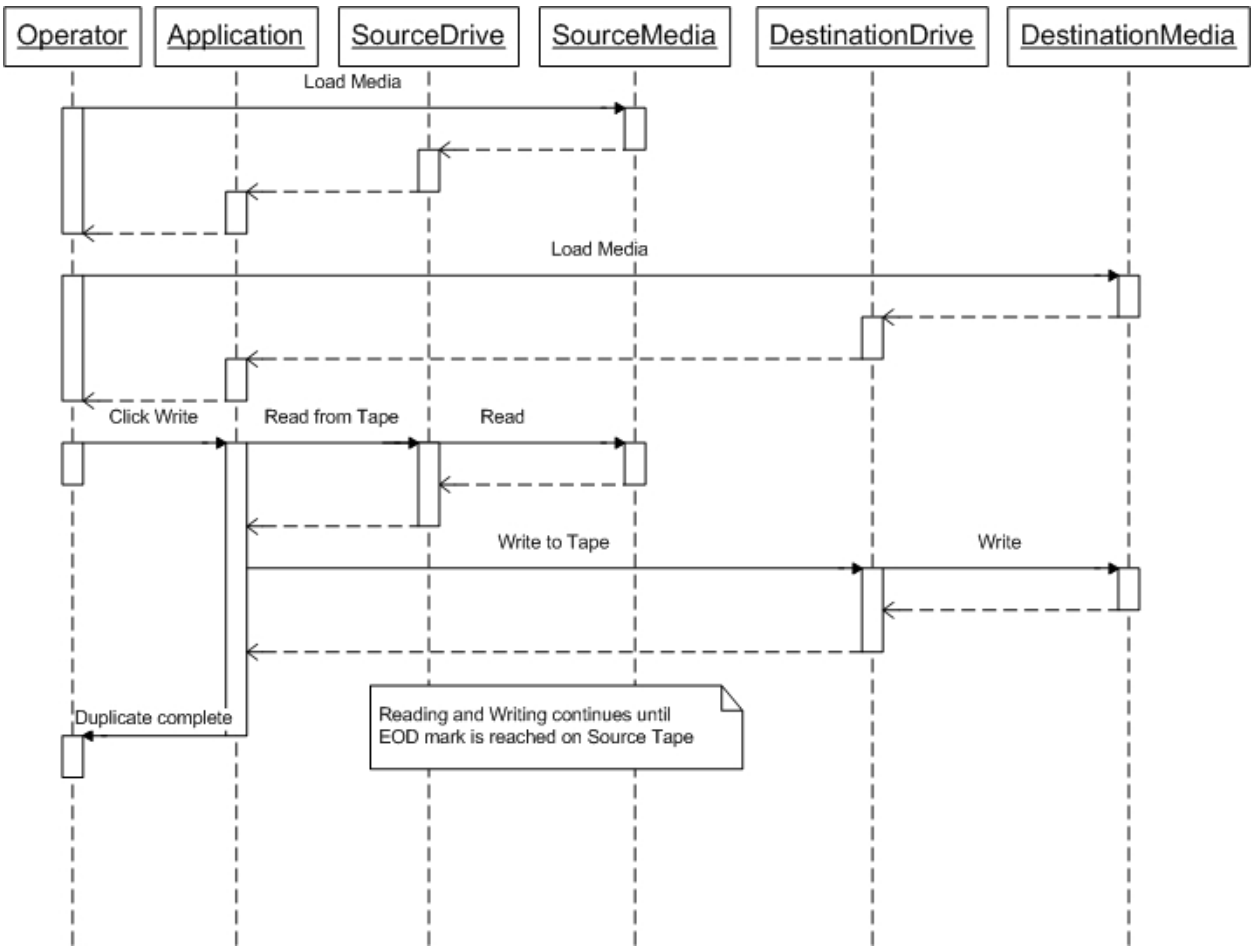
Sequence diagram: Image tape contents



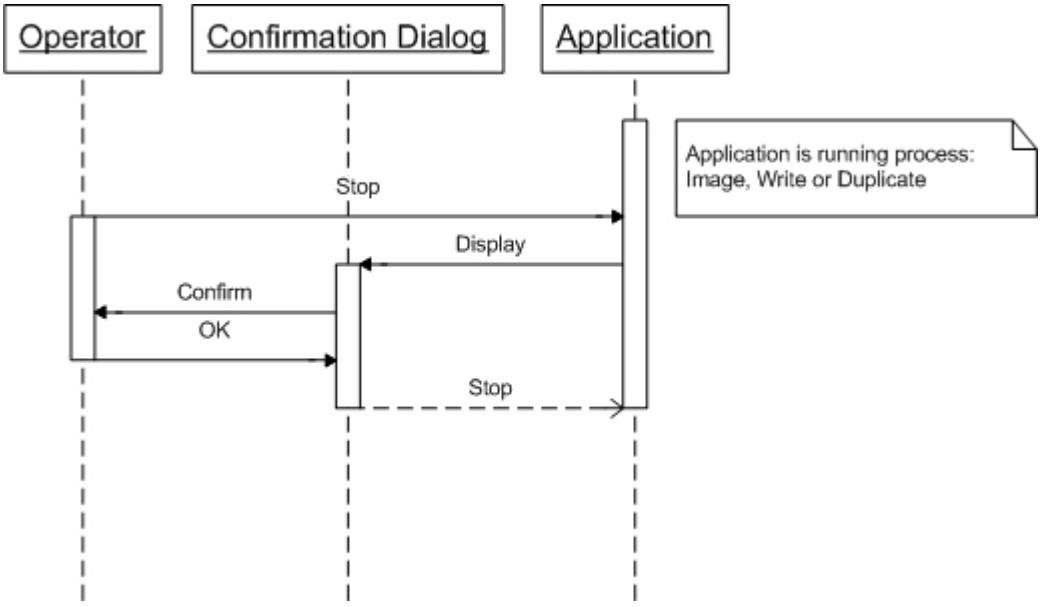
Sequence diagram: Write image onto tape



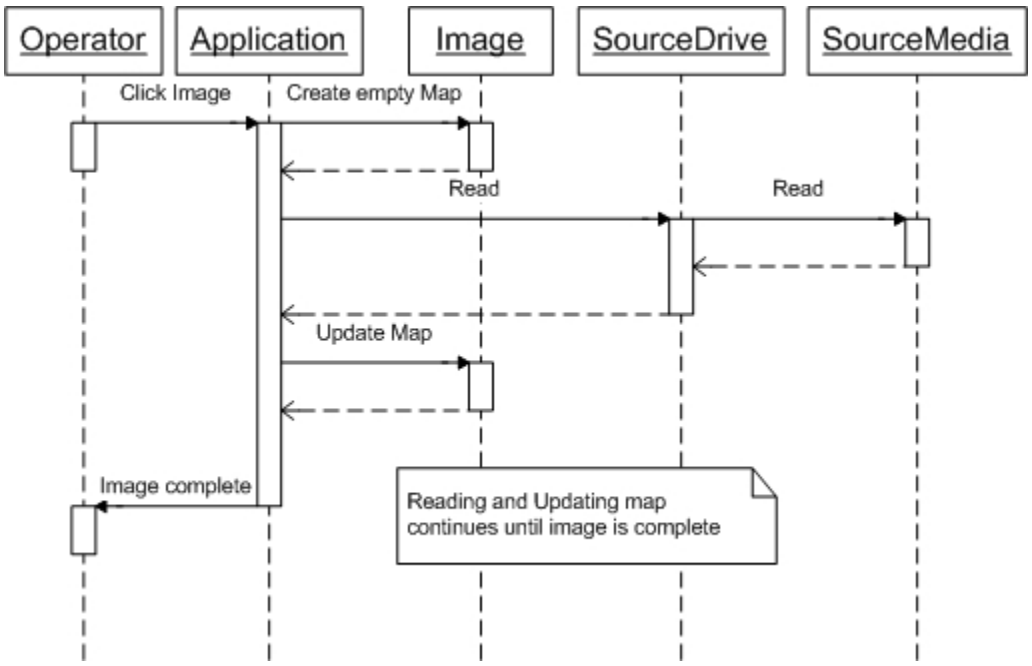
Sequence diagram: Duplicate tape



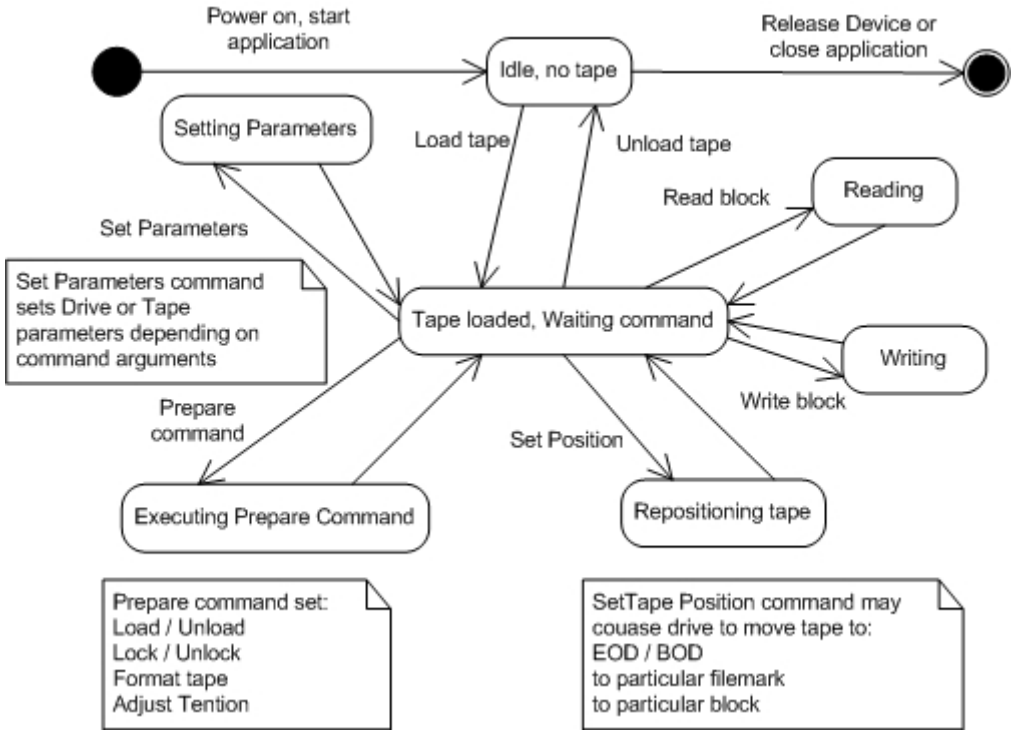
Sequence diagram: Interrupt process



Sequence diagram: Create layout map



State diagram: TapeDevice object



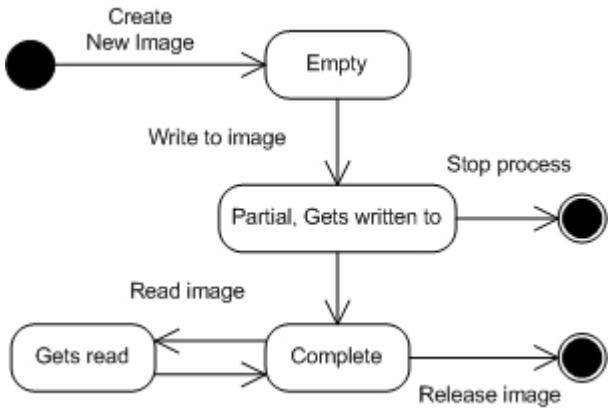
States:

1. Idle, no tape – Tape drive is powered on, connected to the system and configured properly; application started; the slot for tape is empty
2. Tape loaded, waiting command – the tape is in the drive's slot; tape drive is ready and waiting for command
3. Executing Prepare command – Tape drive is executing of the commands belonging to the set of PrepareTape commands.
4. Repositioning tape – Tape drive is moving tape using method specified in the command
5. Reading – drive is reading physical block from the tape starting from the current position.
6. Writing – drive is writing to the tape starting from the current position.
7. Setting parameters – drive is executing set parameters command modifying drive configuration or tape parameters.

Events:

1. Load tape – operator manually loads tape in the drive's slot
2. Unload tape – operator manually unloads tape from the slot
3. Read block – drive receives read block command
4. Write block – drive receives write block command
5. Set Position – drive receives Set Position command
6. Prepare – drive receives Prepare tape command
7. Set Parameters – drive receives set parameters command

State diagram: Image object



States:

1. Empty – a new Image just got created, it does have folder component but it is empty (does not have any files in it).
2. Partial, Gets written to – Image object has been created and application is in process of saving data into files
3. Complete – Image is complete.

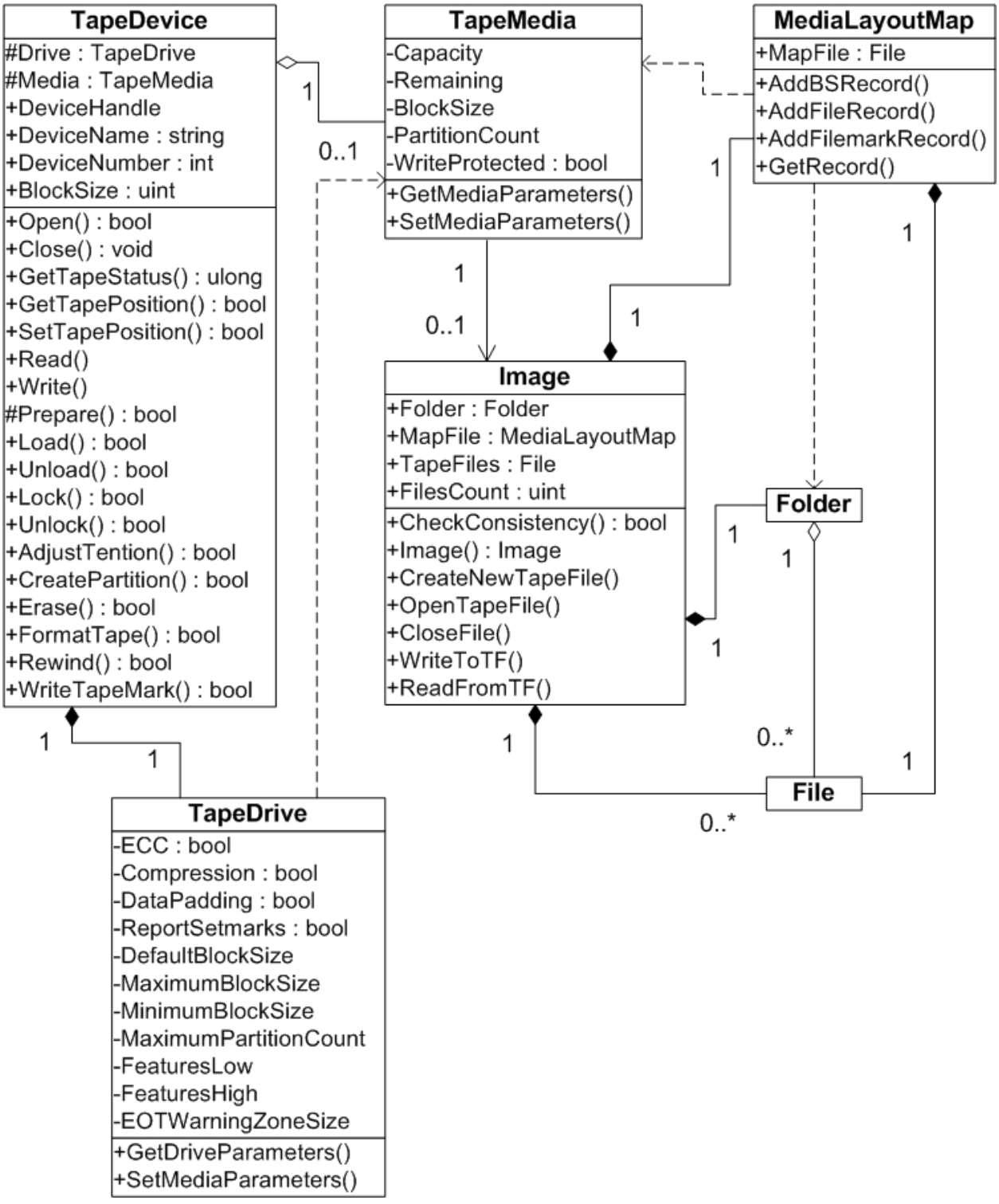
4. Gets read – Image is complete; application is accessing it (read) while creating tape duplicate.

Events:

1. Create New Image – instantiation of the Image class; occurs when operator starts imaging tape.
2. Write to image – application writes data from the tape into Image files
3. Read Image – application reads from existing image while creating tape duplicate
4. Release image – application releases all the handles and does not need particular Image any more.
5. Stop process – operator cancels currently running process

5. Elaborate Class Diagram

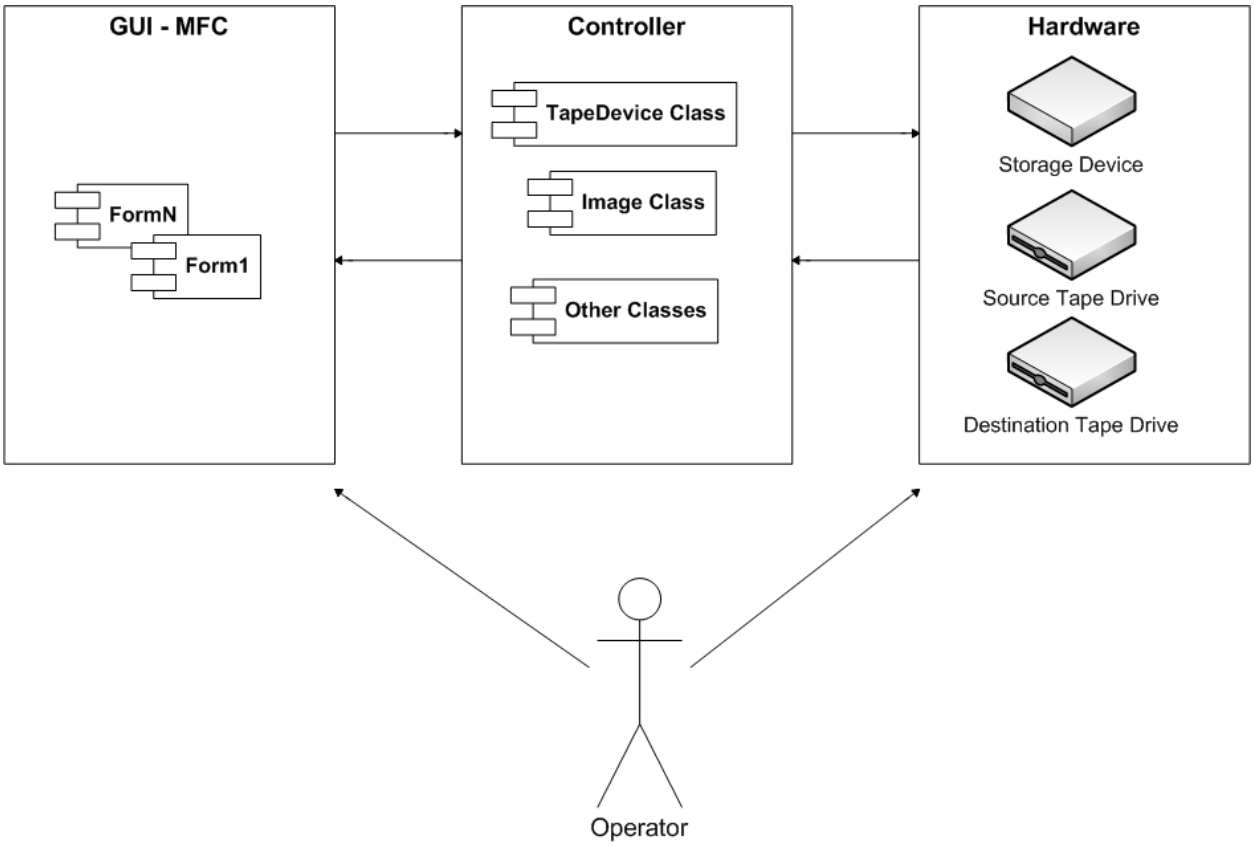
The diagram showed below represents elaborated classes, their relationship and detailed sets of identified features



6. Architectural consideration

The program is a Windows forms type or MFC application utilizing simple graphical interface (dialog based control) and using Windows API functions in order to control tape drive devices.

Architecture diagram:



The architecture utilizes simplified MVC pattern, replacing Model part for information domain with the hardware to be controlled (tape devices).

7. Validation Criteria

Designed application shall be validated. The main criterion is to satisfy scoped requirements listed in this specification. Functional testing approach (black box testing) will be used for validation and verification (V&V). Series of test-cases will be engineered and conducted for V&V purposes. At the end, validated demo release of Application will be presented for final approval.

Classes of tests:

Representative set of tape samples will be created for validation purposes. The set will have several tapes of different type (i.e. LTO, DLT, DDS, etc.) created by different backup software (BackupExec, ArcServe, NetBackup, etc.). The functionality testing will be performed in the next areas (classes of tests):

1. Tape operations (TapeDevice class)
2. Image operations (Image class)
3. The validity and consistency of created images and duplicates
4. GUI functionality