Stage 1:

As a first step to understand the functionality of different components of the project without any information about detail of involved components (Blackbox view), I followed the following steps:

1. ClearCanvas was installed as a workstation that can interact with a PACS system through DICOM protocol. ClearCanvas has been selected because it is compatible with DCM4CHEE as a free PACS and DICOM server.

2. DCM4CHE, which is a collection of open source applications and utilities, was installed. In general, DCM4CHEE is comprised of two parts. The first part is "Dcm4che2 toolkit" which is an open source and robust implementation of the DICOM standard, and the second part is "Dcm4chee" which is an archive and image manager and has the potential to play the role of a comprehensive PACS system.

To see the functionality of the whole system, DCM4CHEE was configured to use the mySql database, and also was defined an Application Entity called DCM4CHEE in ClearCanvas along with its configuration. Then, I did some interactions with system like sending and receiving various type of images (MRI, CT,…). It was observed that the system works properly.

Stage2:

After setting up a proper configuration for whole system, the second step is concentration on the detail of each component. As can be seen from following figure (based on http://www.dcm4che.org), DCM4CHEE has different modules.

Figure1: DCM4CHEE architecture

For our project, developing an Adaptor to parse a DICOM message, it seems that the DICOM server is the core part. Therefore, I set up a Java project based on DCM4CHEE source code in Eclipse environment to analyse the code to discover the relation of packages and particularly the communication among classes. The last version of source code, which we are going to work on it, has 39 folders and each folder contains several packages. So, extracting manually the relation among classes is not easy and strait forward job.

To resolve this challenge, I installed plugins for Eclipse to analyse the Java code and extract class diagram and relation. Following figures show the class diagram for two packages of one folder (dcm4chee-core) which has totally four packages.

Figure2: Class diagram for package: org.dcm4che2.io
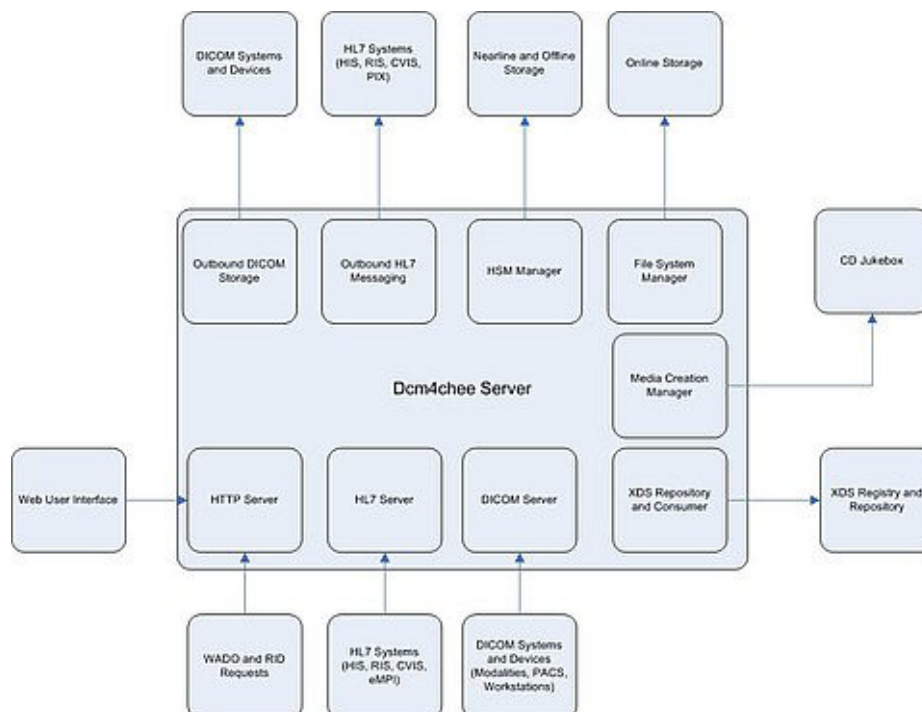
Figure3: Class diagram for package: package org.dcm4che2.media

Next Step:

I believe that after analysing the code and understanding the role of each package and its classes we are required to reuse the related part of this source code based on our design for Adaptor. By the way, we should provide a detail design for the Adaptor and a test bed for developing and debugging.