ingenieur
wissenschaften
htw saar

Hochschule für
Technik und Wirtschaft
des Saarlandes
University of
Applied Sciences

Master-Thesis

zur Erlangung des akademischen Grades
Master of Science (M. Sc.)
an der Hochschule für Technik und Wirtschaft des Saarlandes
im Studiengang Praktische Informatik
der Fakultät für Ingenieurwissenschaften

Analyse, Vergleich und prototypische Implementierung verschiedener Möglichkeiten zur multimodalen Datenfusionierung in das DICONDE Format

vorgelegt von
Christian Wirth

betreut und begutachtet von
Prof. Dr. Helmut G. Folz
Prof. Dr.-Ing. Klaus Berberich

Saarbrücken, 28. 02 2021

Selbstständigkeitserklärung

Ich versichere, dass ich die vorliegende Arbeit (bei einer Gruppenarbeit: den entsprechend gekennzeichneten Anteil der Arbeit) selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

Ich erkläre hiermit weiterhin, dass die vorgelegte Arbeit zuvor weder von mir noch von einer anderen Person an dieser oder einer anderen Hochschule eingereicht wurde.

Darüber hinaus ist mir bekannt, dass die Unrichtigkeit dieser Erklärung eine Benotung der Arbeit mit der Note „nicht ausreichend“ zur Folge hat und einen Ausschluss von der Erbringung weiterer Prüfungsleistungen zur Folge haben kann.

Saarbrücken, 28. 02 2021

Christian Wirth

Zusammenfassung

Diese Arbeit beschäftigt sich mit der Umsetzung von Teilen des Digital Imaging and Communication in Non Destructive Evaluation (DICONDE) Standards als Softwarelösung. Der Fokus liegt dabei auf den im Bereich der Werkstoffprüfung mit Ultraschall anfallenden Daten. Hierzu zählen unter anderem Bilder und Ultraschall Rohdaten. Außerdem werden verschiedene Netzwerkfunktionalitäten des Digital Imaging and Communications in Medicine (DICOM) bzw. DICONDE Standards bereitgestellt. Dazu werden zum einen die im betrieblichen Umfeld benötigten Anforderungen und Erfordernisse mittels einer Stakeholderanalyse und einer Anforderungsanalyse ermittelt.
Des Weiteren werden verschiedene Bibliotheken, die zur Implementierung des Projektes infrage kommen, miteinander verglichen. Dies geschieht unter anderem mithilfe von exemplarischen Implementierungen eines kleinen Teiles des DICONDE Standards. Anschließend wird die Implementierung der Softwarelösung beschrieben. Dabei wird der Ansatz verfolgt, die benötigten Funktionalitäten in einer zentralen Softwarebibliothek zu implementieren, die dann in verschiedene, bereits existierende oder neue Softwarelösungen integriert werden kann. Außerdem werden Vorschläge zur Erweiterung des DICONDE Standards, mit dem Ziel der besseren Abbildung von Ultraschalldaten, erarbeitet.
Zum Schluss wird der Erfolg dieser Umsetzung anhand zuvor ermittelten Anforderungen überprüft. Dies geschieht unter anderem, indem die implementierten Funktionalitäten in bereits bestehende Softwarelösungen integriert werden.

We have seen that computer programming is an art,
because it applies accumulated knowledge to the world,
because it requires skill and ingenuity, and especially
because it produces objects of beauty.

— Donald E. Knuth [24]

Danksagung

An dieser Stelle möchte ich den Personen danken, die mich während dieser Arbeit unterstützt haben. An erster Stelle möchte ich hier meinem Firmenbetreuer Bernd Sprau danken. Außerdem möchte ich Prof. Dr. Helmut Folz und Prof. Dr.-Ing. Klaus Berberich für die Betreuung dieser Arbeit danken. Nicht zuletzt gebührt ein ganz spezieller Dank meiner Familie für die bedingungslose Unterstützung während meines ganzen Studiums. Sie war immer für mich da und die ganzen Jahre in vielerlei Hinsicht eine große Stütze. Auch möchte ich meinen Freunden Mesut Oezcan und Cedric Schreiner danken, mit denen zusammen ich mein Studium bestritten habe und die mir währenddessen immer mit Rat und Tat zur Seite standen.

Inhaltsverzeichnis

1 Einleitung

In diesem Kapitel wird das Thema der Arbeit beschrieben. Dabei wird auch auf die Motivation und die Umgebung, in der die Arbeit erstellt wurde, eingegangen.

1.1 Motivation

Im Bereich der Zerstörungsfreien Prüfung (ZfP) fallen unterschiedlich strukturierte Daten aus verschiedenen Messverfahren an. Dies führt dazu, dass die Wissensgewinnung aus diesen Daten zusätzlich erschwert wird, da die verschieden strukturierten Daten nur schwer miteinander verglichen werden / in Relation gesetzt werden können. Deshalb ist es wünschenswert, die Daten in ein einheitliches Format zu bringen. Ein solches, für die ZfP entwickeltes, Format ist das DICONDE-Format. Im Fraunhofer-Institut für Zerstörungsfreie Prüfverfahren (Fraunhofer IZFP) soll genau dies, als Teil des DiNA 4.0 (Data for iNovative Applications) Projektes umgesetzt werden.

1.2 Thema

Diese Arbeit beschäftigt sich mit der Verarbeitung, Speicherung und Übertragung der im Rahmen von ZfP mit Ultraschall anfallenden Daten. Der Fokus liegt dabei auf der Überführung dieser Daten in das DICONDE-Format. Dies soll realisiert werden, indem eine Softwarebibliothek erstellt wird, die unter anderem, Funktionalitäten zum Konvertieren von Messdaten in das DICONDE-Format, das Laden von Daten aus dem DICONDE-Format sowie verschiedene DICONDE Netzwerkfunktionalitäten bereitstellt.

1.3 Umgebung der Arbeit

Im Folgenden wird das betriebliche Umfeld dieser Arbeit beschrieben.

1.3.1 Fraunhofer-Gesellschaft

„Die Fraunhofer-Gesellschaft mit Sitz in Deutschland ist die weltweit führende Organisation für anwendungsorientierte Forschung. Mit ihrer Fokussierung auf zukunftsrelevante Schlüsseltechnologien sowie auf die Verwertung der Ergebnisse in Wirtschaft und Industrie spielt sie eine zentrale Rolle im Innovationsprozess. Sie ist Wegweiser und Impulsgeber für innovative Entwicklungen und wissenschaftliche Exzellenz. Mit inspirierenden Ideen und nachhaltigen wissenschaftlich-technologischen Lösungen fördert die Fraunhofer-Gesellschaft Wissenschaft und Wirtschaft und wirkt mit an der Gestaltung unserer Gesellschaft und unserer Zukunft.“[22]

1.3.2 Fraunhofer IZFP

Das Fraunhofer-Institut für Zerstörungsfreie Prüfverfahren (Fraunhofer IZFP) beschäftigt sich als Teil der Fraunhofer-Gesellschaft mit anwendungsorientierter Forschung im Bereich der Zerstörungsfreien Prüfverfahren. [21]

1.3.3 DiNA 4.0

Bei Data for iNnovative Applications 4.0 (DiNA 4.0) handelt es sich um ein Strategisches Investitionsprojekt des Fraunhofer IZFP, welches auf 3 Jahre ausgelegt ist und sich in die beiden Hauptbereiche Digitalisierung und Projektmanagement gliedert. Der Bereich Digitalisierung spaltet sich weiterhin in die Unterbereiche Open Platform Communications Unified Architecture (OPC-UA) und DICONDE auf. Das Gesamtziel des Projektes ist dabei, die Umsetzung der internen Daten-Digitalisierungsstrategie auf Sensor- und Technologie-Ebene (Mess-, Prüf-, und Materialdaten) in einer modernen Plattformlösung und Erprobung in und mit Kundenprojekten. Diese Arbeit wurde im Kontext des Unterbereichs DICONDE durchgeführt. [22]

2 Grundlagen

In diesem Kapitel werden verwendete Technologien und Grundlagen von für die Arbeit relevanten Techniken beschrieben. Dazu zählen zum Beispiel verwendete Frameworks und Ultraschall-Grundlagen. Außerdem werden die für diese Arbeit relevanten Aspekte von DICOM und DICONDE beschrieben.

2.1 DICOM

DICOM ist ein Standard mit dem Zweck zu definieren, wie medizinisches Bildmaterial gehandhabt, übermittelt und mit zusätzlichen Informationen verknüpft wird. [2, S. 11] Der Standard wird und wurde mit einem Schwerpunkt auf bildgebende Diagnoseverfahren in der Medizin entwickelt, lässt sich aber auch auf eine Vielzahl an sonstigen zu übertragenden Informationen im medizinischen Umfeld anwenden. [2, S. 11]

2.1.1 Historisches

Der Standard wird von einem Komitee entwickelt, das 1983 vom American College of Radiology (ACR) und der National Electrical Manufacturers Association (NEMA) gegründet wurde. [2, S. 11–12] Im Jahr 1995 wurde das Komitee in „DICOM Standards Committee“ umbenannt und für eine breit angelegte Zusammenarbeit von verschiedenen Interessengruppen aus dem Fachgebiet der medizinischen Bildgebung umorganisiert. [2, S. 12]

2.1.2 Prinzipien

Ein Prinzip des DICOM Standards ist, das er weltweit angewendet werden kann. Um dies zu gewährleisten, sind unter anderem verschiedene Schriftsysteme, Zeichensätze, Sprachen enthalten. Ein weiteres Prinzip des DICOM Standards ist, dass jede Instanz global eindeutig identifizierbar ist. Dies wird erreicht, indem jeder Instanz eine global eindeutige ID zugewiesen wird. [2, S. 12–13]

2.1.3 Model of the Real World

Die genaue Darstellung von Objekten aus der realen Welt im DICOM Standard ist unter [5, S. 111–114] zu finden. Da für diese Arbeit allerdings eine vereinfachte Betrachtung des „Model of the Real World“ ausreicht, werden in diesem Abschnitt nur die relevanten Module betrachtet.

In Abbildung 2.1 ist dieses vereinfachte Modell zu sehen.

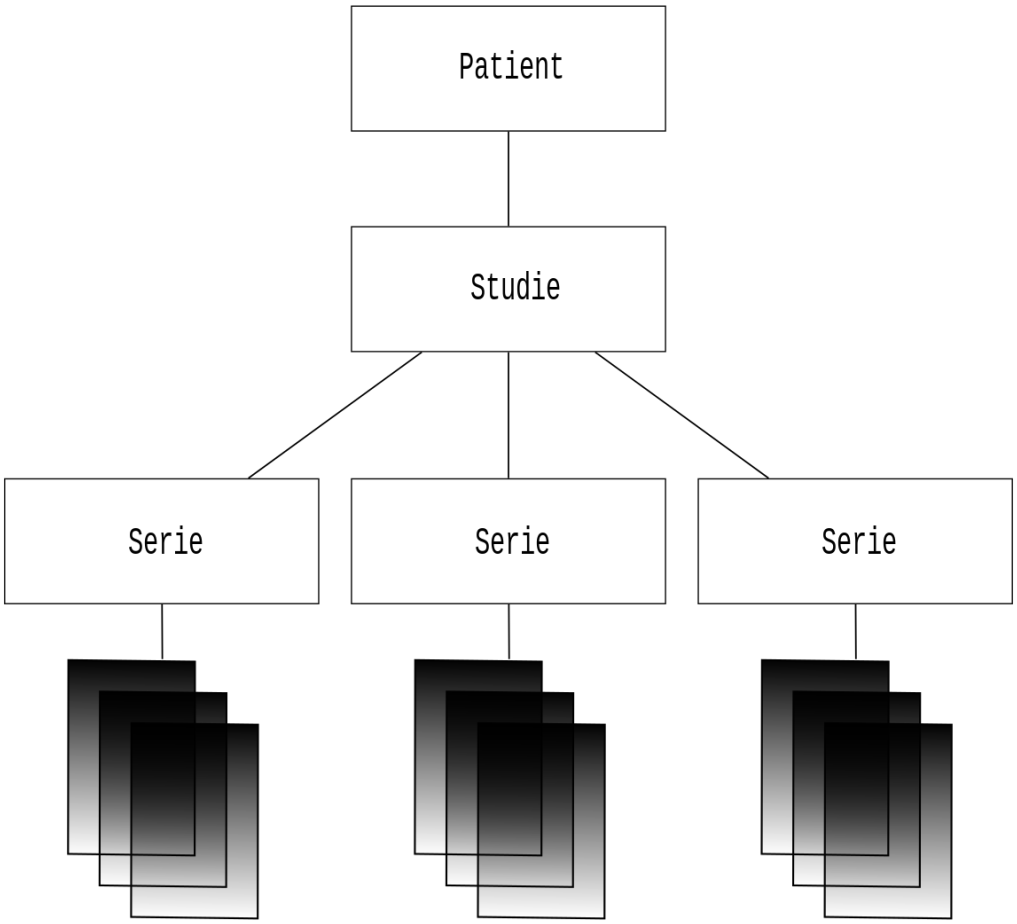

Abbildung 2.1: Vereinfachte Darstellung des DICOM „Model of the Real World“. [18]

Die in Abbildung 2.1 zu sehenden Module sind dabei wie folgt zu verstehen.

Patient Das Patienten Modul stellt die zu einem Patienten gehörenden Daten wie Name, Geburtsdatum, Geschlecht und Wohnort dar. Außerdem können hier Informationen über den Gesundheitsstatus hinterlegt werden, zum Beispiel ob der Patient Raucher ist oder welche Allergien er hat. [5, S. 115, 446–448]

Studie Im Studien Modul werden Informationen hinterlegt, die für den gesamten Verlauf einer Untersuchung, Behandlung etc. relevant sind. Beispiele hierfür sind, die behandelnden Ärzte oder bisher durchgeführten Prozeduren. [5, S. 448]

Serie In einer Serie sind Informationen, die sich direkt auf eine durchgeführte Prozedur oder ähnliches beziehen. Beispiele hierfür sind der Zeitpunk an dem die Prozedur durchgeführt wurde, sowie welche Prozedur durchgeführt wurde. [5, S. 497]

Bilder Die unter den Serien zu sehenden Bilder, beinhalten die eigentlichen Bildinformationen.

Diese Komponenten stehen dabei wie folgt in Verbindung. Eine Studie ist immer einem Patienten zugeordnet und beinhaltet eine oder mehrere Serien. Die eigentlichen Bildinformationen werden dann innerhalb der Serie abgelegt. [5, S. 111]

2.1.4 Information Object Definition

Im DICOM Standard wird eine Anzahl an Information Object Definitions (IODs) definiert. Es handelt sich dabei um eine abstrakte Abbildung eines Objektes aus der echten Welt. Vergleichbar mit einer Klasse aus dem Bereich der objektorientierten Programmiersprachen. [5, S. 107]

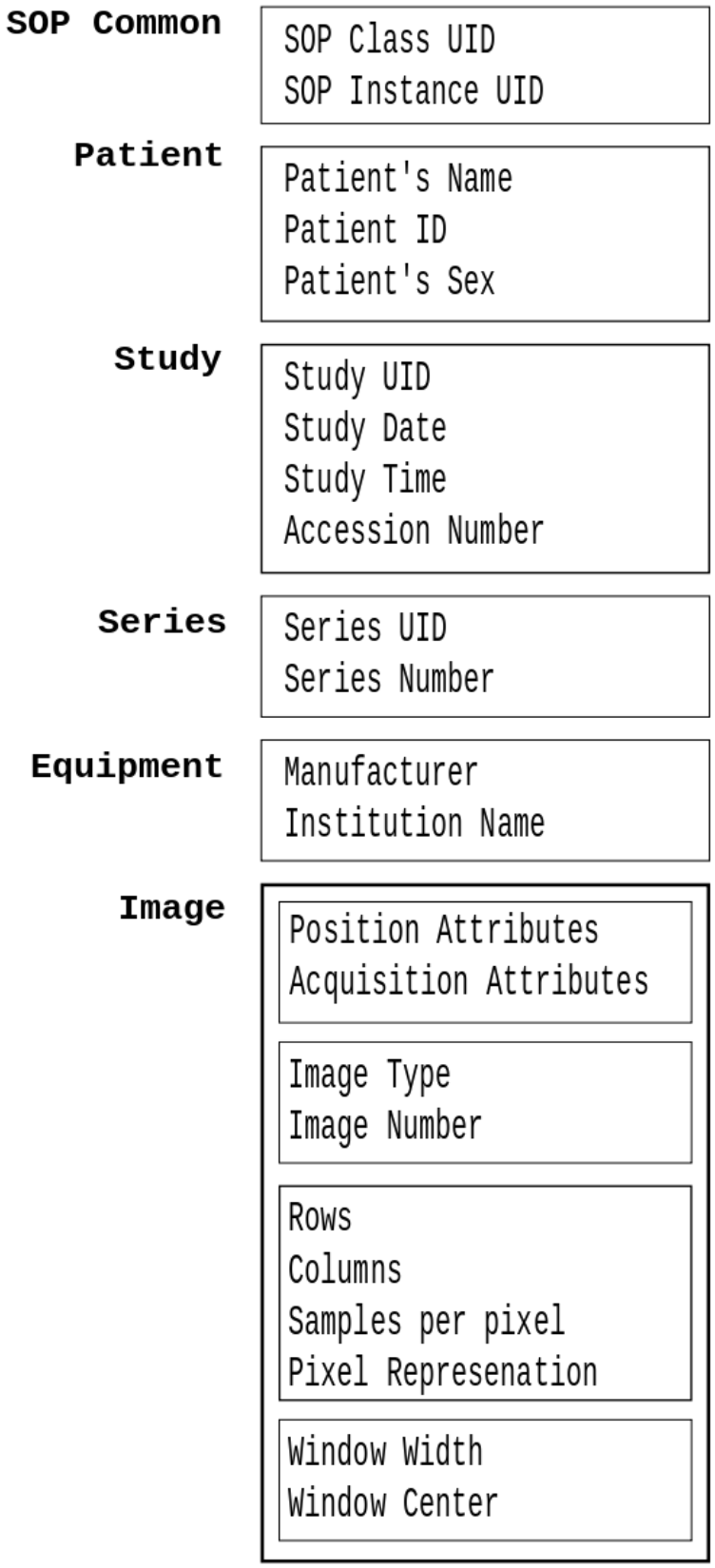

Abbildung 2.2: Beispielhafter Aufbau einer DICOM Composite IOD. [39]

In der Abbildung 2.2 ist der Aufbau eine IOD zu sehen. Anzumerken ist dabei, dass die Information Object Definition (IOD) durch eine Menge von Attributen definiert ist, die miteinander verwandten Attribute werden dabei wieder in Modulen gruppiert. Die Attribute werden dabei als Data Elements encodiert. (Siehe Abschnitt 2.1.10) [5, S. 108]

Man unterscheidet zwischen zwei Arten von IODs. [5, S. 107]

Composite IOD Bei einer „Composite IOD“ handelt es sich um eine IOD, die zu dem Objekt, das in der IOD abgebildet wird, im Bezug stehen. Sie setzt sich dabei aus mehreren Entitäten aus dem in [5, 111 ff] beschriebenen „Model of the Real World“ zusammen. In Abbildung 2.2 handelt es sich um eine solche IOD. [5, S. 107] [5, 201ff]

Normalized IOD Bei einer „Normalized IOD“ handelt es sich um eine IOD, die nur eine einzige Entität aus dem „Model of the Real World“ darstellt. [5, S. 107] [5, 437ff]

2.1.5 Services

Im DICOM Standard werden zum Austausch von Informationen bestimmte Services definiert. Bevor die für diese Arbeit relevanten Services aufgezählt und kurz erläutert werden, müssen noch die folgenden Begriffe eingeführt werden.

DICOM message service element (DIMSE) Protocol Legt fest, wie Nachrichten beim DICOM Datenaustausch aufgebaut sein müssen. [9, S. 17]

DIMSE Service User Kann zwei Rollen einnehmen, entweder er ist der Aufrufende oder der Ausführende DIMSE Service User. [9, S. 33]

DIMSE Service Provider Ist eine Entität, die DIMSE Services bereitstellt. [9, S. 33]

Folgende Services werden im Zuge dieser Arbeit genutzt: [9, S. 33]

C-Store Dienst zum speichern einer SOP Instance von einem DIMSE Service User auf einem anderen DIMSE Service User. [9, S. 43]

C-Find Dienst um von einem DIMSE Service User, anhand eines Satzes von Attributen mit Attributen von SOP Instancen auf einem anderen DIMSE Service User abzugleichen. [9, S. 45]

C-Get Wie bei C-Find kann von einem DIMSE Service User ein Satz von Attributen mit SOP Instances auf einem anderen DIMSE Service User abgeglichen werden. Außerdem werden mittels mehrerer C-STORE Unteroperationen alle SOP Instances zurückgegeben die passen. [9, S. 47]

2.1.6 WADO-RS

Bei Web Access to DICOM Objects by RESTful Services (WADO-RS) handelt es sich um einen im DICOM-Standard spezifizierten RESTful Web Service, der unter andrem [4, 101 ff] das Abrufen von Daten von einem Picture Archiving and Communication System (PACS) Server ermöglicht.

2.1.7 Service-Object Pair (SOP) Class

Eine SOP Class ist eine Kombination aus einem Service und einer IOD. Der Zweck ist dabei, die Nutzung der im Abschnitt 2.1.5 erläuterten Services für eine Bestimmte im Abschnitt 2.1.4 erläuterten IOD einzuschränken. [6, S. 50]

2.1.8 File Format

Das DICOM File Format beschreibt, wie DICOM Informationen in einer Datei abgelegt werden. In diesem Kapitel werden die für diese Arbeit relevanten Aspekte des DICOM File Format erläutert. Innerhalb einer DICOM Datei wird dabei immer eine Instanz, einer im Abschnitt 2.1.7 erläuterten Service-Object Pair (SOP) Class, gespeichert.

Datei-Header Der Datei-Header besteht aus einer 128-Byte Preamble, gefolgt von einem 4-Byte Präfix, der fest auf den Wert „DICM“ gesetzt ist. In diesen 128-Byte können Informationen gespeichert werden, um die gekapselten Datasets zu identifizieren. [3, S. 31]

Metainformationen-Header Im Metainformationen-Header sind Informationen zur Codierung der Datei abgelegt. Insbesondere der Transfer-Syntax, welcher unter anderem aus der Art der Value Representation (VR) Codierung, der Byte-Reihenfolge und der verwendete Kompression besteht. Dies ist wichtig, da nur so sichergestellt werden kann, dass zwischen verschiedenen Systemen übertragene Daten richtig interpretiert werden können.

2.1.9 Dataset

Ein DICOM-Dataset stellt ein Informationsobjekt aus der echten Welt dar. Es setzt sich dabei aus den im Kapitel 2.1.10 beschriebenen DICOM Data Elements zusammen. In der Abbildung 2.3 ist ein solches DICOM Dataset beispielhaft dargestellt. Dazu ist anzumerken, dass ein DICOM Dataset in der Regel deutlich mehr DICOM Data Elements enthält und es sich bei der Abbildung nur um ein Beispiel zur Veranschaulichung des Prinzips handelt.

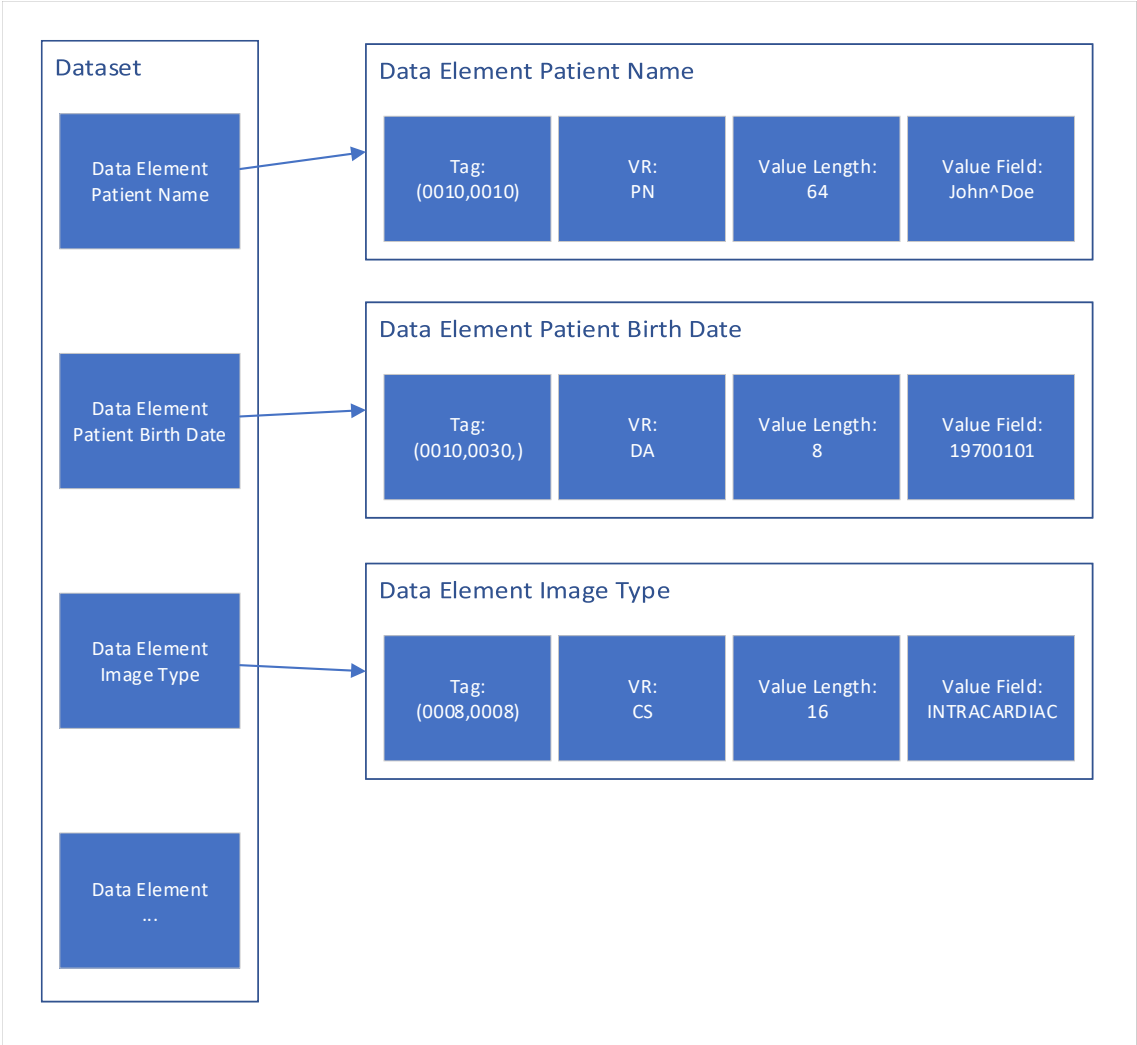

Abbildung 2.3: Beispiel für ein DICOM Dataset.

Wie in Abbildung 2.3 zu sehen ist, werden für jedes Data Element die im Folgenden beschriebenen Felder gespeichert.

Tag Dient zur eindeutigen Identifizierung des Elements. Wird im Abschnitt 2.1.10 genauer beschrieben.

VR Definiert die Struktur der Daten. Wird im Abschnitt 2.1.10 genauer beschrieben.

Value Length Dieses Feld gibt die Länge des Value Fieldes in Bytes an.

Value Field Dieses Feld enthält die eigentlichen Werte.

Außerdem ist anzumerken, dass es sich in den in Abbildung 2.3 dargestellten DICOM Data Elements um solche mit „Explicit VR“ handelt. Die Alternative dazu wäre eine „Implicit VR“. Wobei das VR Feld wegfallen würde und die Struktur des Value Feldes durch die in [8] für das entsprechende Tag definierte VR festgelegt wäre. [7, S. 51]

2.1.10 Data Elements

Ein DICOM Data Element wird durch die folgenden Bestandteile definiert.

Tag Es handelt sich dabei um ein vorzeichenloses 16-Bit Integer. Die ersten 8 Bit werden dabei als Group Number und die letzten 8 Bit als Element Number bezeichnet.

Value Representation (VR) Die VR gibt an, wie die Daten des Data Elements strukturiert sein müssen bzw. in welchem Datentyp sie gespeichert werden. Sie besteht aus zwei 8-Bit Chars [7, S. 37]. Mögliche VRs sind im DICOM-Standard unter [7, S. 37–50] definiert.

Value Multiplicity (VM) Gibt an, wie viele Werte in diesem Feld gespeichert werden können. [7, S. 50]

Type Gibt an, ob das Attribut zwingend erforderlich oder optional ist. Beziehungsweise, unter welchen Bedingungen das Data Element erforderlich oder optional ist. Dabei kann der Type die folgenden Werte enthalten.

1 Muss enthalten sein.

1C Muss unter bestimmten Bedingungen enthalten sein, hängt meist von anderen Attributen ab.

2 Muss enthalten sein, kann aber, falls der Wert unbekannt ist, ohne Wert hinterlegt werden.

2C Muss unter bestimmten Bedingungen enthalten sein, kann aber, falls die Bedingungen erfüllt sind und der Wert unbekannt ist, ohne Wert hinterlegt werden.

3 Dieser Wert bedeutet, dass das Attribut optional ist.

[7, S. 55–56]

Dabei wird zwischen Standard Data Elements und Private Data Elements unterschieden. Standard Data Elements erkennt man dabei an einer geraden „Group Number“ es handelt sich hierbei um bereits im DICOM-Standard definierte Data Elements. Private Data Elements sind hingegen an einer ungeraden Group Number zu erkennen. Sie können für Anwendungsfälle definiert werden, die im DICOM-Standard, in dieser Form, nicht vorgesehen sind. [7, S. 60–62] Für beide Arten von Data Elements gibt es Sonderfälle, in denen eine gerade oder ungerade Group Number nicht wie beschrieben für die entsprechende Art Data Element steht. Da diese Sonderfälle aber für diese Arbeit nicht relevant sind, werden sie hier nicht näher erläutert.

2.1.11 Sequences

Bei der Speicherung von DICOM Data Elements in einem Dataset stellt das Speichern von DICOM Sequences einen für diese Arbeit relevanten Sonderfall dar.

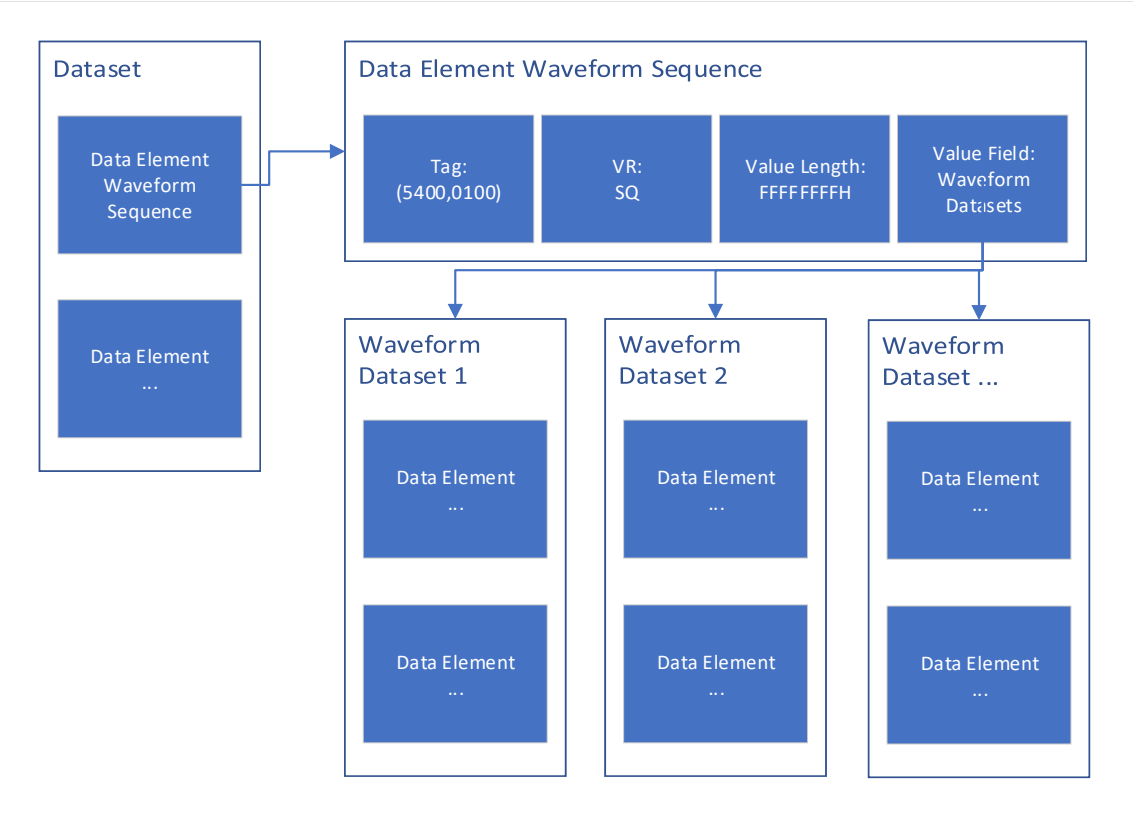

Abbildung 2.4: Beispiel für eine DICOM Sequence.

Wie in der Abbildung 2.4 exemplarisch zu sehen ist, werden innerhalb einer DICOM Sequence wiederum mehrere DICOM Datasets abgelegt. Diese enthalten dann wieder eigene DICOM Data Elements. Es ist auch möglich, dass die inneren DICOM Datasets weitere DICOM Sequences beinhalten. Ebenfalls ist zu sehen, dass die VR einer DICOM Sequence immer „SQ“ lautet. Außerdem fällt auf, dass im Feld Value Length der Wert „FFFFFFFF“ eingetragen ist. Dieser Wert steht für eine undefinierte Länge und wird bei DICOM Sequences standardmäßig verwendet. [7, S. 52]

2.2 DICONDE

DICONDE ist eine Erweiterung des DICOM-Standards, die dazu dient, den Standard auf Verfahren aus der ZfP anwendbar zu machen. [33, S. 1] Unter andem wir dies erreicht, indem DICOM konforme IODs für bildgebende Verfahren aus der ZfP definiert werden. [33, S. 1]

Im Folgenden werden die für diese Arbeit relevanten Unterschiede bzw. Erweiterungen von DICONDE gegenüber DICOM dargestellt.

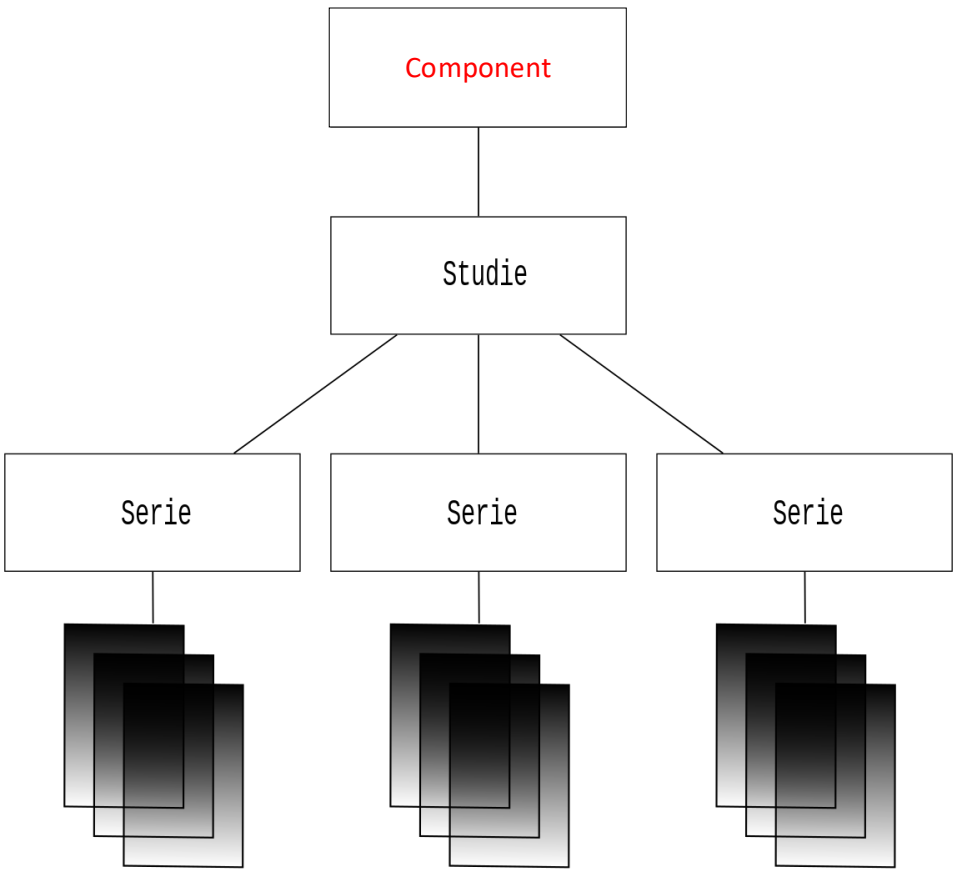

Abbildung 2.5: Unterschied Real World Model DICOM/DICONDE (Bearbeitete Version von [18]).

Wie in Abbildung 2.5 zu sehen, besteht der Unterschied vom DICONDE „Model of the Real World“ zum DICOM „Model of the Real World“ hauptsächlich darin, dass der Patient durch „Component“ ersetzt wird.

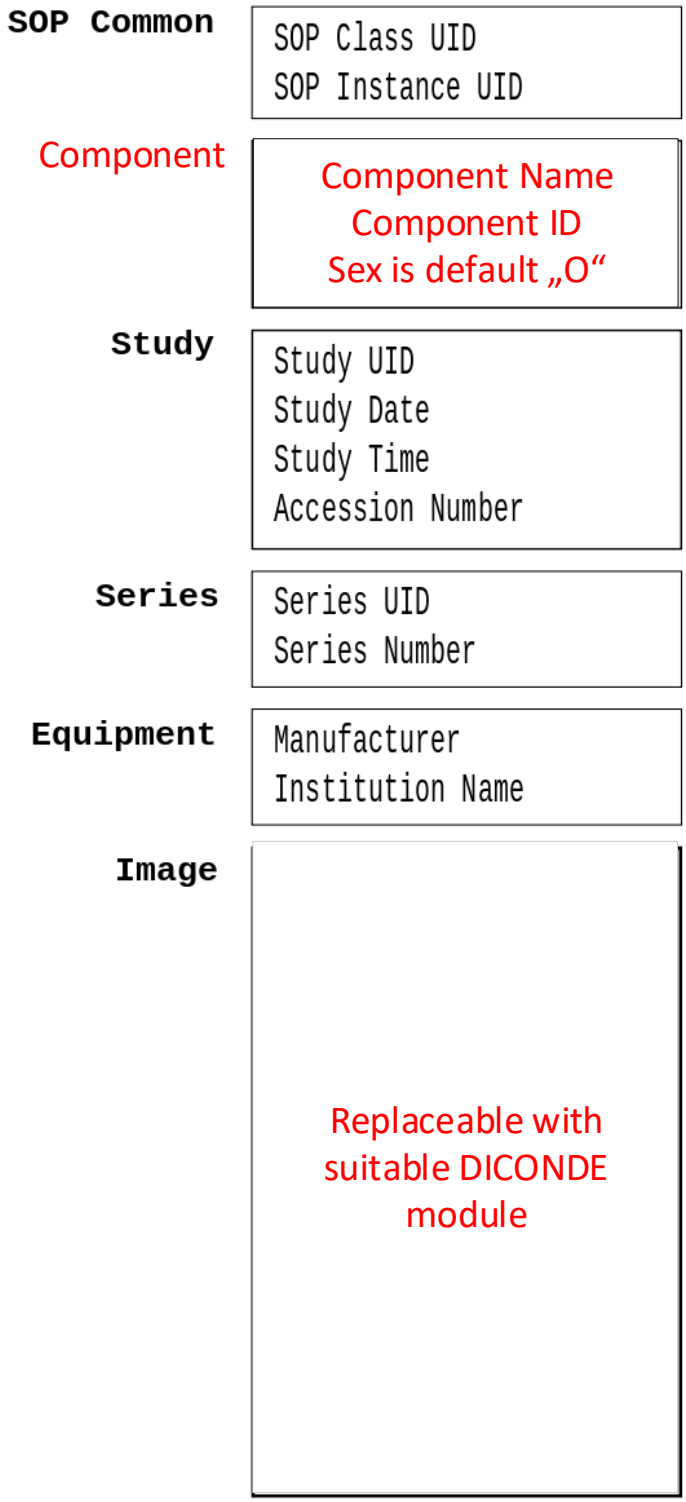

Abbildung 2.6: Unterschied IODs DICOM/DICONDE (Bearbeitete Version von [39]).

Wie in Abbildung 2.6 zu sehen ist, liegt ein weiterer Unterschied in dem Aufbau der IODs. Zum einen, wirkt sich hier aus, dass Patient mit „Component“ ersetzt wird. Zum

anderen wird das Modul zum Speichern der Bilder mit dem entsprechenden, je nachdem um welche IOD es sich handelt, DICONDE Modul ersetzt. [33]

2.3 Werkstoffprüfung mit Ultraschall

Bei Ultraschall im Allgemeinen handelt es sich um Schall mit einer Frequenz von über 20000 Hz. Im Bereich der Werkstoffprüfung wird allerdings nur Schall ab einer Frequenz von 100000 Hz (100 kHz, Kilohertz) bis über 10000000 Hz (10 MHz, Megahertz) eingesetzt. [23, S. 1] Das Ziel ist es dabei, die Schallwellen als Informationsträger über den Zustand eines zu prüfenden Objektes zu nutzen. Bei den in dieser Arbeit behandelten Verfahren wird dabei grundsätzlich so vorgegangen, dass eine Schallwelle im Werkstück erzeugt wird und die rückläufige Welle wieder empfangen wird. [23, S. 2]

Anwendungsbereiche für diese Verfahren sind dabei unter anderem die Prüfung von Verbindungen und Material in Flugzeugen, Zügen, Fahrzeugen sowie im Bereich des Maschinenbaus, von Kraftwerke und der metallerzeugeneden Industrie. [23, S. 3] Zusammengefasst also alle Prüfungen von Schweißnähten, die Prüfung auf Risse oder die Prüfung auf unerwünschte Einschlüsse.

Folgende, für diese Arbeit relevanten Begriffe aus dem Ultraschall-Bereich, sollen hierfür kurz eingeführt werden.

Phased Array Bei Phased Array wird im Gegensatz zum konventionellen Ultraschall mit mehren Sendern / Empfängern gearbeitet. Durch zeitlich verzögerte Anregung der Einzelelemente kann ein Beamforming hervorgerufen werden, welches u. a. einen programmatisch einstellbaren Fokuspunkt ermöglicht. [1]

A-Scan Bei A-Scan handelt es sich um einen digitalisierten Amplitudenverlauf entlang der Zeitachse. Dargestellt meist durch ein eindimensionales Short-Array, wobei die Werte dem Aplitudenwert und der Index einem Zeitpunkt entspricht.

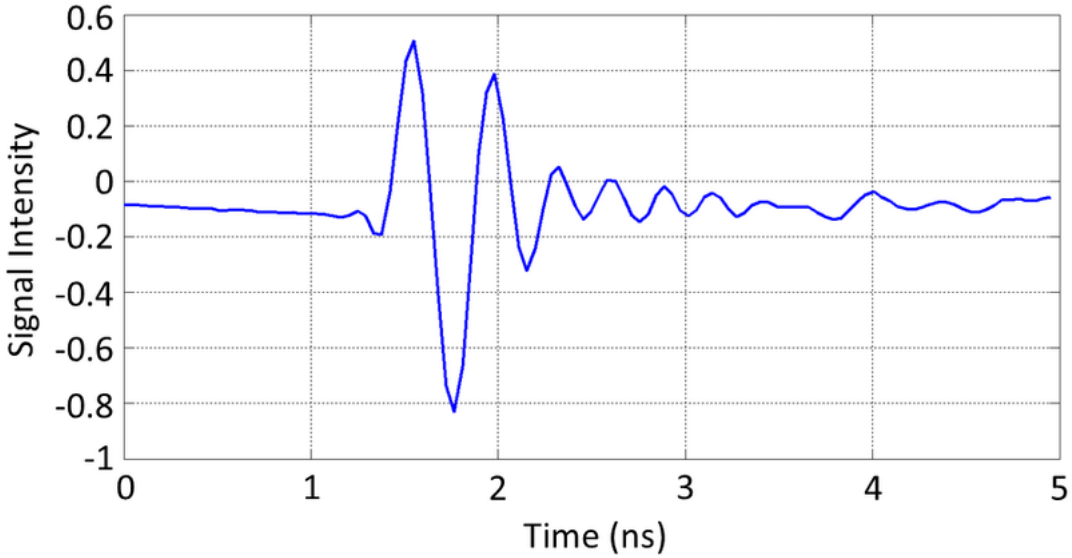

Abbildung 2.7: Beispielhafter A-Scan [28]

Single Scan Dieser Scan liefert einen A-Scan zurück, unabhängig davon, ob er von einem Einzelschwinger oder von einem Phased-Array aufgenommen wurde.

Linear Scan Durch Schieben einer virtuellen Apparatur durch einen Phased-Array-Kopf, wird die Scanposition elektronisch verschoben. Beispiel: Zunächst schießen die Elemente 1-5 dann die Elemente 2-6 und so weiter. Hierdurch ergibt sich ein rechteckiges Abtastbild.

Sector Scan Hier sind die genutzten Elemente konstant, dafür werden diese zeitlich versetzt angeregt, um über ein Beamforming einen Sektor-Abschnitt zu scannen. Hierbei erhalten wir eine Liste von A-Scans, welche zur korrekten Darstellung noch mit ihrem jeweiligen Winkel projiziert werden müssen.

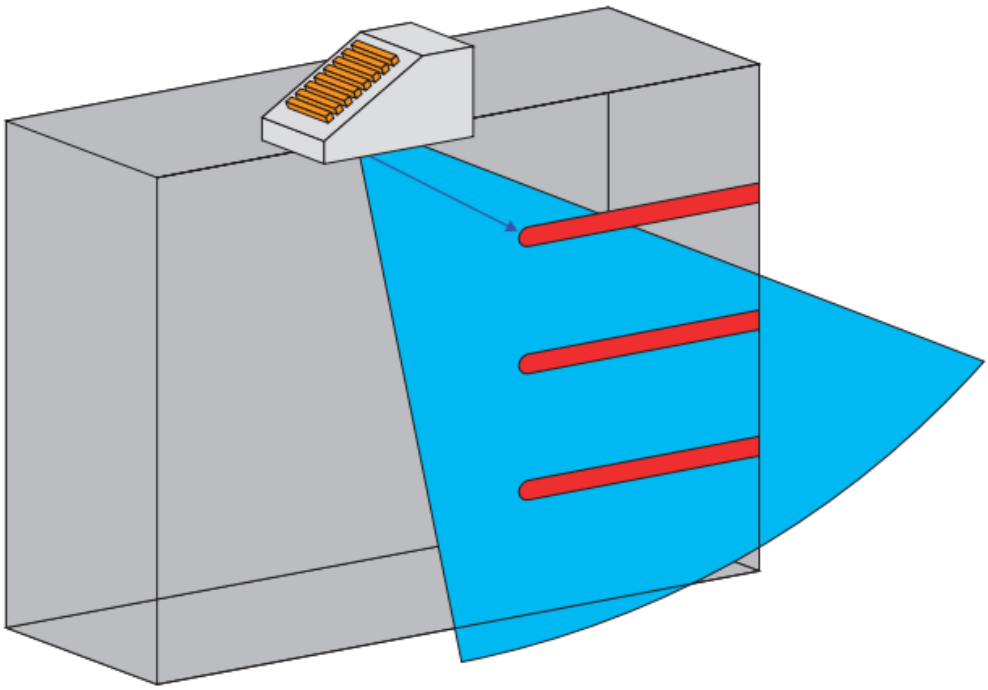

Abbildung 2.8: Beispielhafter Darstellung eines Sektorscans [19, S. 180].

2.4 Existierende Softwarelösungen

In diesem Abschnitt werden die bereits vorhanden Softwarelösungen, die im Zusammenhang mit dieser Arbeit stehen, aufgezählt und kurz erläutert.

2.4.1 DIMATE (Ehemals Visus) JiveX NDT

JiveX NDT Communication Server Hierbei handelt es sich um einen Picture Archiving and Communication System (PACS)-Server, der speziell für die zerstörungsfreie Prüfung erweitert wurde.

JiveX NDT Viewer Pro Standalone Anzeigelösung auf Java Basis, welche die Visualisierung von DICONDE Daten zulässt. Außerdem sind einfache Rekonstruktionen möglich, zum Beispiel CT-Schichtbilder 3-Dimensional zu visualisieren.

2.4.2 Ultraschall-Software-Plattform

Bei der Ultraschall-Software-Plattform (USP) handelt es sich um eine vom Fraunhofer IZFP entwickelte Software zur Durchführung und Auswertung von Ultraschallmessungen. Als Programmiersprache wird für die USP C# verwendet. Außerdem wird zur Erstellung der Benutzeroberfläche das Framework Windows Presentation Foundation (WPF) verwendet. Die Software ist für diese Arbeit relevant, da sie institutsweit eingesetzt wird, um Werkstoffprüfungen mittels Ultraschall zu erstellen, zu speichern und zu analysieren. Da die Software dabei noch nicht die Möglichkeit bietet, diese Daten im DICONDE-Format abzulegen bzw. diese wieder daraus zu lesen, werden im Rahmen dieser Arbeit Funktionalitäten zu der Software hinzugefügt bzw. bestehende Funktionen so abgeändert, dass das DICONDE-Format unterstützt wird.

2.4.3 USLab

USLab ist eine auf Einzellschwinger spezialisierte Datenaufnahme- und Auswerte-Software, die am IZFP weit verbreitet ist. Unter anderem zur Durchführung von akkreditierten Prüfungen. In abgewandelter Form findet diese Software auch Anwendung bei verschiedensten Industrieanlagen. In diese auf LabView basierenden Software sollen im Laufe dieser Arbeit in der DICONDE-Bibliothek 5.1 implementierte Funktionalitäten integriert werden.

2.5 Verwendete Technologien

In diesem Abschnitt wird aufgeführt, welche Technologien während der Erstellung dieser Arbeit verwendet wurden.

2.5.1 LabView

Bei LabView handelt es sich um eine visuelle Systementwicklungssoftware. Der Fokus liegt dabei auf der einfachen Erstellung von Anwendungen im industriellen Bereich. [38]

2.5.2 .NET

Bei .NET handelt es sich um eine von Microsoft entwickelte Software-Entwicklungsplattform, die verschiedene Programmiersprachen wie C# und Visual Basic unterstützt. [27]

2.5.3 .NET Framework

.Net Framework ist ein Teil der im Abschnitt 2.5.2 beschriebenen Entwicklungsplattform. Es handelt sich dabei um die erste Implementierung der .NET-Api, die insbesondere den Nachteil hat, dass sie keine plattformübergreifend Entwicklung ermöglicht. [27]

2.5.4 .NET Core

.Net Core ist ebenfalls ein Teil der im Abschnitt 2.5.2 beschriebenen Entwicklungsplattform. Es handelt sich dabei um eine Implementierung der .NET-API die eine plattformübergreifend Entwicklung ermöglicht. [27]

2.5.5 .NET Standard

Bei .NET Standard handelt es sich um eine Spezifikation der .NET-APIs, mit dem Ziel, eine Einheitlichkeit im .NET Ökosystem zu erreichen. [26]

2.5.6 fo-DICOM

Bei Fellow Oak DICOM (fo-DICOM) handel es sich um ein C#-Framework zum Arbeiten mit dem DICOM Standard. [42]

2.5.7 Tabula-Py

Bei Tabula-Py handelt es sich um einen Python Bibliothek. Zweck der Bibliothek ist es, Inhalte aus in PDFs gespeicherten Tabellen auszulesen. [35]

2.6 Model View ViewModel (MVVM)

Bei MVVM handelt es sich um ein Pattern zum Erstellen von Software mit Grafischer Benutzeroberfläche. Das Pattern besteht, wie schon am Namen zu erkennen, aus den folgenden drei Komponenten.

View Definiert die Struktur sowie die Darstellung.

Model Das Model enthält das Datenmodell, die Businesslogik sowie die Validierungslogik.

View Model Das View Model vermittelt zwischen View und Model und regelt die Logik der View.

Die einzelnen Komponenten sind dabei voneinander entkoppelt, was den Vorteil hat, dass die Komponenten einfach ausgetauscht werden können. In Abbildung 2.9 ist zu sehen, wie die Komponenten miteinander in Beziehung stehen. [25]

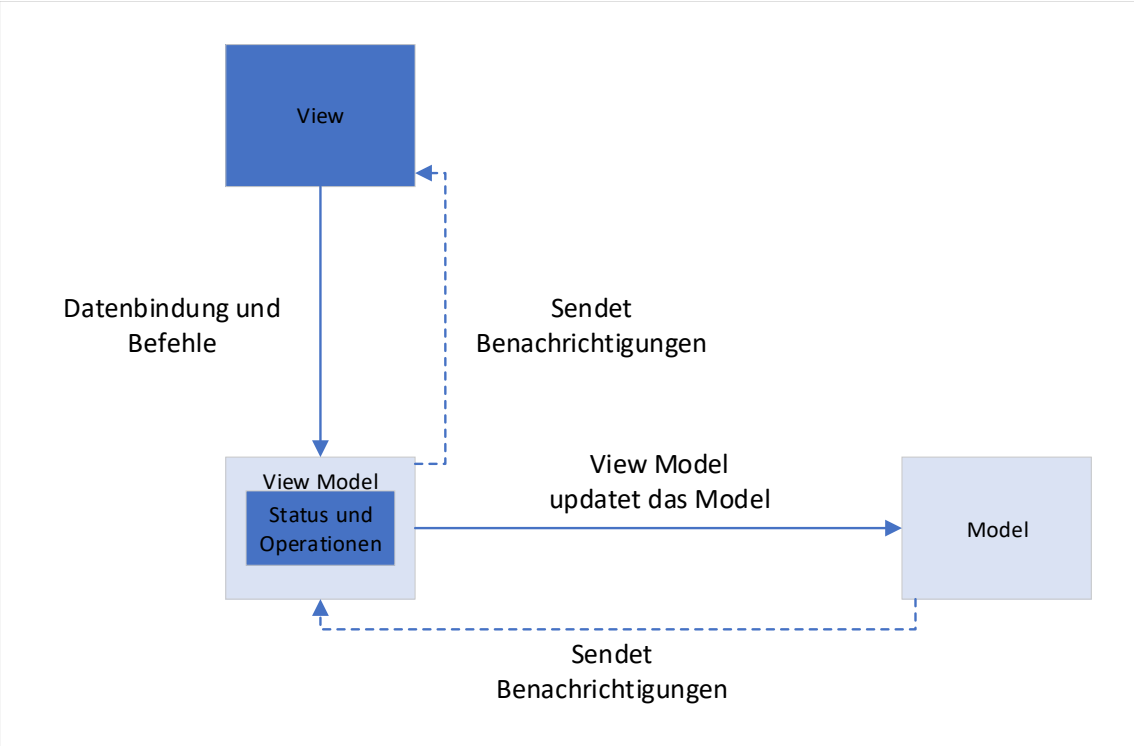

Abbildung 2.9: Beziehung der Komponenten [25]

3 Anforderungsanalyse und Konzept

In diesem Kapitel wird ein Überblick über die verwendeten Projektmanagement-Methoden gegeben. Des Weiteren wird die Ermittlung der Anforderungen beschrieben. Außerdem wird eine Marktanalyse durchgeführt und der Entscheidungsprozess zur Auswahl des Frameworks dargestellt.

3.1 Vorgehen

Im ersten Schritt wird eine Stakeholderanalyse erstellt. Der Zweck dieser ist zum einen geeignete Kommunikationsstrategien mit den Stakeholdern zu ermitteln, aber auch um eventuelle, aus den Interessen der Stakeholder resultierenden Anforderungen zu erkennen.

Im nächsten Schritt soll unter Berücksichtigung der Zielgruppen, so wie der Produktumgebung, festgelegt werden, welche Anforderungen an das Produkt gestellt werden. Außerdem soll die Kompatibilität mit den bisher vorhandenen Softwarelösungen bei der Ermittlung der Anforderungen berücksichtigt werden.

Im letzten Schritt wird dann festgelegt, wie die Anforderungen überprüft werden sollen.

3.2 Stakeholderanalyse

Mit Hilfe der Stakeholderanalyse soll ermittelt werden, wer im Unternehmen in irgendeiner Form Interesse an der im Rahmen dieser Arbeit zu erstellenden Software hat, beziehungsweise wessen Arbeit damit im Zusammenhang steht. Im konkreten Fall sind dies insbesondere Personen, die im Unternehmen Daten, die durch bildgebende Messverfahren entstanden sind, verarbeiten, speichern oder übertragen müssen. Auch soll ermittelt werden, wer bereits auf dem DICONDE-Standard basierende Verfahren oder Software einsetzt. Außerdem soll analysiert werden, in welcher Art und Weise die jeweiligen Steakholder betroffen sind, wie ihre Einstellung zu dem Projekt ist und welchen Einfluss sie darauf nehmen könnten.

Die Stakeholderanalyse besteht dabei aus den folgenden 3 Phasen

Stakeholder identifizieren In der ersten Phase wird ermittelt, welche Stakeholder es für das Projekt gibt.

Einfluss der Steakholder analysieren In dieser Phase wird analysiert, welcher der Stakeholder welche Bedeutung für das Projekt hat. Dies dient dazu, eine Grundlage für die Verteilung von Zeit und Mitteln zu bilden.

Maßnahmen ableiten In der letzten Phase wird auf Basis der bisher gesammelten Erkenntnisse abgleitet, welche Maßnahmen zum Umgang mit den Stakeholdern zu treffen sind.

3.2.1 Stakeholder identifizieren

Um die Stakeholder des Projektes zu ermitteln, werden strukturierte Interviews mit Projektmitgliedern des DiNA 4.0-Projektes durchgeführt. Die Interviews drehen sich dabei um die Frage, wie dessen Bezug zu dem Projekt ist und ob noch weitere Personen bekannt sind, die befragt werden müssen.

3.2.2 Einfluss der Stakeholder analysieren

Nachdem die potenziellen Stakeholder ermittelt sind, wird jedem möglichen Stakeholder ein Fragebogen zur Bearbeitung übermittelt. Dieser Fragebogen enthält Fragen zur Tätigkeit des potenziellen Stakeholders und seinem/ihrem Bezug zum DiNA 4.0-Projekt. Außerdem wir noch einmal spezifischer darauf eingegangen, inwiefern der potenzielle Stakeholder mit bildgebenden Verfahren arbeitet und ob er/sie dabei bereits DICONDE einsetzt, oder dieses plant. Außerdem wurden die potentiellen Stakeholder gebeten, ihre sonstigen Interessen und Anmerkungen bezüglich des Projektes darzulegen.

3.2.2.1 Auswertung der Fragebögen

In diesem Abschnitt werden die Ergebnisse der Auswertungen dargestellt, dabei handelt es sich bei den Angaben um eine Kombination aus der Auswertung der Fragebögen und der Interviews.

Die zusammengefasste Auswertung der Ja-Nein-Fragen ist in Tabelle 3.1 zu sehen.

Frage	Ja	Nein
Erstellen, verarbeiten oder speichern Sie im Rahmen Ihrer Tätigkeit Daten, die im Zusammenhang mit bildgebenden Prüfverfahren entstanden sind?	9	0
Arbeiten Sie zur Zeit im Rahmen des DiNA 4.0-Projektes?	8	1
Falls Sie selbst Software entwickeln, haben sie den DICONDE-Standard bereits in Ihrer Software implementiert, beziehungsweise planen diesen zu implementieren?	3	6
Verwenden Sie bereits Software die den DICONDE-Standard implementieren oder planen Sie eine solche Software zu verwenden?	8	1

Tabelle 3.1: Auswertung der Ja-Nein-Fragen.

Wie aus Tabelle 3.1 abzulesen, handelt es sich bei allen Personen, um Stakeholder, da jeder davon in Verbindung zu bildgebenden Prüfverfahren steht und somit auch eine mögliche Verbindung zum DICONDE-Standard besteht.

Die Rollen der Stakeholder im DiNA 4.0-Projekt lassen sich dabei, wie in Tabelle 3.2 zu sehen, kategorisieren.

Projektleitung	2
Projektmanagement	1
Softwareentwicklung	3
Geschäftsmodellentwicklung	2

Tabelle 3.2: Anzahl der Personen je Gruppe.

Außerdem lassen sich aus den Fragebögen sowie den persönlichen Anmerkungen einige Interessen der verschiedenen Gruppen im Projekt ableiten. In der Tabelle 3.3 werden diese Interessen dieser Gruppen den Interessen des Autors im Rahmen dieser Arbeit gegenübergestellt.

Rolle	Interessen der Stakeholder	Interessen des Autors
Projektleitung	Umsetzung der Projektziele innerhalb des zeitlich vorgesehenen Rahmens. Unter anderem die Umsetzung der Speicherung von Rohdaten im DICONDE-Format.	Befürwortung beziehungsweise Unterstützung des im Rahmen dieser Arbeit erstellten Projektes.
Projektmanagement	Das reibungslose Zusammenspiel der verschiedenen im Projekt durchgeführten Aktivitäten, mit dem Ziel, die von der Projektleitung festgelegten Ziele zu erreichen.	Befürwortung beziehungsweise Unterstützung des im Rahmen dieser Arbeit erstellten Projektes.
Softwareentwicklung	Technische Kompatibilität der im Rahmen dieser Arbeit erstellten Softwarelösungen, insbesondere im Hinblick auf die verwendeten Programmiersprachen.	Zusammenarbeit bei der Erstellung dieser Arbeit, insbesondere im Bezug auf Informationen über bereits bestehende Lösungen, beziehungsweise bei der Integration in bereits bestehende Lösungen.
Geschäftsmodellentwicklung	Monetarisierung der im Projekt erstellten Produkte beziehungsweise des generierten Wissens. Im Bezug auf diese Arbeit insbesondere die Erstellung eines verwendbaren Produktes beziehungsweise Teilproduktes.	Austausch von Informationen zur Nutzung der im Rahmen dieser Arbeit erstellten Produkte im Bezug auf mögliche Geschäftsmodelle.

Tabelle 3.3: Interessen der Gruppen

Den Einfluss, das Konfliktpotenzial und die Einstellung zum Projekt der Stakeholder werden im Folgenden nach Rollen gruppiert, da die Auswertung der Fragebögen ergab, dass diese Faktoren innerhalb der verschiedene Rollen weitgehend deckungsgleich sind. In Tabelle 3.4 werden die drei genannten Faktoren für die jeweiligen Rollen jeweils eingeordnet. Dabei können den Faktoren die folgenden Werte zugeordnet werden:

Einfluss Keiner - Gering - Mittel - Hoch - Sehr hoch

Konfliktpotential Keines - Gering - Mittel - Hoch - Sehr hoch

Einstellung Stark ablehnend - Ablehnend - Neutral - Befürwortend

Rolle	Einfluss	Konfliktpotential	Einstellung
Projektleitung	Sehr hoch	Gering	Befürwortend
Projektmanagement	Hoch	Gering	Befürwortend
Softwareentwicklung	Mittel	Hoch	Befürwortend
Geschäftsmodellentwicklung	Mittel	Mittel	Neutral

Tabelle 3.4: Einfluss, Konfliktpotential und Einstellung der Gruppen.

Die Einschätzung des Einflusses der Projektleitung als „Sehr hoch“ kommt dadurch zustande, dass diese sowohl die Budgetverantwortung des Projektes als auch die Weisungsbefugnis gegenüber den restlichen Projektmitgliedern innehat. Der Einfluss des Projektmanagements wird als „Hoch“ eingeschätzt, da dieses eng mit der Projektleitung zusammenarbeitet und außerdem ein Mitspracherecht bei der Verteilung der Aufgaben innerhalb des Projektes hat. Der Einfluss der Mitglieder der Softwareentwicklung wird hingegen als „Mittel„ eingestuft, da diese keine direkte Weisungsbefugnis innehaben und nur indirekt Einfluss nehmen können.

Im Gegensatz zum Einfluss der Projektleitung wird deren Konfliktpotenzial nur als „Gering“ eingeschätzt, da sie das Speichern von Messdaten im DICONDE-Format als eines der Projektziele des DiNA4.0-Projektes festgelegt hat und diese Arbeit zu eben diesem Ziel beiträgt. Für das Projektmanagement gilt dieselbe Einschätzung. Das Konfliktpotenzial der Mitglieder der Softwareentwicklung wird hingegen als „Hoch“ eingeschätzt, da diese teilweise selbst mit der Erstellung von Softwarelösungen zur Verarbeitung und Speicherung von Messdaten beauftragt sind. Das Konfliktpotenzial wird dabei insbesondere deshalb als hoch eingeschätzt, da es möglicherweise zu Inkompatibilitäten zwischen den verschiedenen Lösungen kommen könnte, woraus Konflikte resultieren könnten. Das Konfliktpotenzial der Geschäftsmodellentwicklung wird als „Mittel“ eingeschätzt, da diese mit der Monetarisierung der im Rahmen dieses Projektes entwickelten Produkte und Erkenntnisse betraut sind. Dies kann unter Umständen die Verteilung der Projektressourcen beeinflussen.

Bezüglich der Einstellung der verschiedenen Gruppen ergibt die Auswertung der Fragebögen eine befürwortende Einstellung bei der Projektleitung, dem Projektmanagement sowie den Mitgliedern der Softwareentwicklung. Die Mitglieder der Geschäftsmodellentwicklung stehen der Arbeit wohlwollend gegenüber.

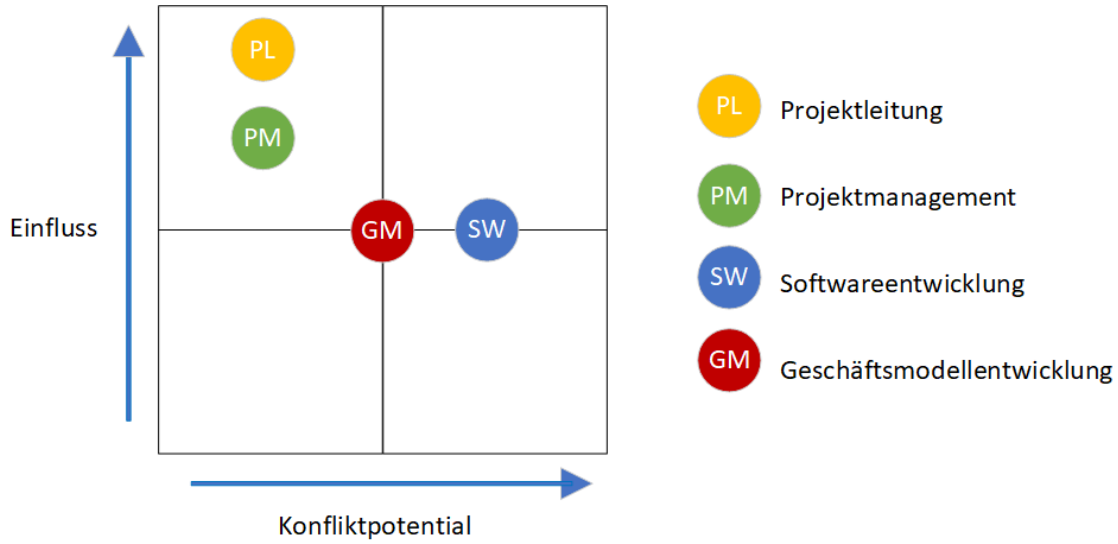

Abbildung 3.1: Verteilung von Einfluss und Konfliktpotential

In der Abbildung 3.1 ist Einfluss und Konfliktpotenzial der verschiedenen Gruppen zur besseren Übersicht noch einmal grafisch abgebildet.

Insgesamt geht aus der Auswertung der Fragebögen hervor, dass bei der Projektkommunikation beziehungsweise beim Treffen der Kommunikationsmaßnahmen ein besonderen Fokus auf die Mitglieder der Projektleitung sowie der Softwareentwicklung gelegt werden sollte. Welche Maßnahmen genau daraus geschlussfolgert werden, wird im nächsten Kapitel 3.2.3 beschrieben.

3.2.3 Maßnahmen ableiten

Nachdem alle notwendigen Informationen über die Stakeholder gesammelt und ausgewertet sind, werden daraus Maßnahmen zur Kommunikation beziehungsweise zur Einbindung der Stakeholder abgeleitet. Dazu müssen zum einen die konkreten Ziele gegenüber den Stakeholdern innerhalb des Projektes definiert werden und zum anderen muss definiert werden, welche Maßnahmen zum Erreichen dieser Ziele geeignet und erforderlich sind. Eine Übersicht dieser beiden Punkte ist in Tabelle 3.5 zu sehen.

Rolle	Ziele im Projekt	Maßnahmen
Projektleitung	Unterstützung bei der Durchführung des Projektes.	Vorstellung des Projektfortschrittes in den 14-tägigen Projektmeetings
Projektmanagement	Unterstützung bei der Durchführung des Projektes.	Vorstellung des Projektfortschrittes in den 14-tägigen Projektmeetings
Softwareentwicklung	Abstimmung bezüglich der technischen Fragestellung im Zusammenhang mit verschiedenen Softwaresystemen.	Vorstellung des Projektfortschrittes in den 14 Tägigen Projektmeetings sowie den ebenfalls 14-tägigen Meetings des Digitalisierung-Teilprojektes. Außerdem regelmäßiges Bekanntmachen von erarbeiten Dokumenten
Geschäftsmodellentwicklung	Abstimmung bezüglich der Monetarisierung der Arbeit im Rahmen des Projektes.	Vorstellung des Projektfortschrittes in den 14-tägigen Projektmeetings

Tabelle 3.5: Abgeleitete Maßnahmen.

3.3 Anforderungsanalyse

In diesem Abschnitt werden die Anforderungen, an die im Rahmen dieser Arbeit erstellten Softwarebibliothek, ermittelt und aufgelistet.
Die Liste der Anforderungen wurde in Abstimmung mit den in Kapitel 3.2 ermittelten Stakeholdern erstellt.

3.3.1 Anwendungsbereich

Der Anwendungsbereich der Softwarebibliothek ist das Erstellen, Einlesen und Übertragen von DICONDE-Dateien.

3.3.2 Zielgruppe

Die direkte Zielgruppe der Softwarebibliothek sind Softwareentwickler innerhalb der Fraunhofer-Gesellschaft, die DICONDE-Funktionalitäten in ihre Software integrieren möchten. Indirekt ist die Zielgruppe jedoch jeder innerhalb der Fraunhofer-Gesellschaft, der eine Zerstörungsfreie Prüfung durchführt und diese im DICONDE-Format speichern will beziehungsweise im DICONDE-Format gespeicherte Messungen aufrufen will.

3.3.3 Technische Produktumgebung

Die Softwarebibliothek soll unter den im Folgenden aufgelisteten Bedingungen voll funktionsfähig sein. Dies bedeutet, dass die konsumierende Softwarelösung bei Verwendung dieser Bibliothek weiterhin funktionsfähig ist, falls sie es zuvor war.

3.3.3.1 Software

Betriebssystem Die Softwarebibliothek soll unter Verwendung des Betriebssystems Windows 10 in der Version 1909 und neuer funktionsfähig sein.

Konsumierende Software Die Softwarebibliothek soll in Software verwendbar sein, die auf LabView in der Version 2018 oder neuer basiert. Außerdem soll die Verwendung der Bibliothek in Software verwendbar sein, die auf .NET Framework in der Version 4.5 basiert.

3.3.3.2 Hardware

Bei der Entwicklung der Softwarebibliothek wird von keiner bestimmten Hardwarekonfiguration ausgegangen, abgesehen davon, dass es sich um ein mit .NET Framework in Version 4.5 oder höher kompatibles System handeln muss. Allerdings wird die Überprüfung der Performance Anforderungen auf der folgenden Hardwarekonfiguration durchgeführt:

Prozessor AMD Ryzen 5 3600

Primärspeicher 16GB DDR4

Sekundärspeicher Kingston A2000 SSD

3.3.3.3 Schnittstellen

Die Softwarebibliothek soll eine Schnittstelle zu einem PACS-Server bereitstellen, die zum Übertragen von DICONDE-Dateien zu diesem PACS-Server, beziehungsweise zum Herunterladen von DICONDE-Dateien von diesem dient.

3.3.4 Funktionale Anforderungen

Folgende funktionale Anforderungen werden festgelegt.

F1 Konvertierung von Bilddaten in das DICONDE-Format Es soll möglich sein, Bilddaten und die dazugehörigen Metainformationen in einer DICONDE-Datei zu speichern.

F2 Konvertierung von Ultraschall-Rohdaten in das DICONDE-Format Es soll möglich sein, Ultraschall-Rohdaten und die dazugehörigen Metainformationen in einer DICONDE-Datei zu speichern.

F3 Laden von Bilddaten aus dem DICONDE-Format Es soll möglich sein, Bilddaten und die dazugehörigen Metainformationen aus einer DICONDE-Datei zu laden.

F4 Laden von Ultraschall-Rohdaten aus dem DICONDE-Format Es soll möglich sein, Ultraschall-Rohdaten und die dazugehörigen Metainformationen aus einer DICONDE-Datei zu laden.

F5 Hochladen von Daten auf einen PACS-Server Es soll möglich sein, die erstellten DICONDE-Dateien auf einen PACS-Server hochzuladen.

F6 Herunterladen von Daten von einem PACS-Server Es soll möglich sein, DICONDE-Daten von einem PACS-Server herunterzuladen.

3.3.5 Nichtfunktionale Anforderungen

Im Folgenden werden die nichtfunktionale Anforderungen festgelegt und erläutert.

3.3.5.1 Performance

Auf der im Kapitel 3.3.3.2 beschriebenen Hardware sollen die folgenden Werte erreicht werden. Die MB Angaben beziehen sich dabei auf die reinen Rohdaten, die üblicherweise als Array von 16-Bit Werten vorliegen.

NF1 Performance der Konvertierung von Ultraschall-Rohdaten in das DICONDE-Format Die Konvertierung der Daten in das DICONDE-Format soll nicht langsamer sein als die Konvertierung in das bisherige Format. Als Richtwert wird die Konvertierung von 500 MB an Ultraschall-Rohdaten, die innerhalb von 3 Minuten konvertiert werden sollen, verwendet.

NF2 Performance des Ladens von Ultraschall-Rohdaten aus dem DICONDE-Format Das Laden der Daten aus dem DICONDE-Format soll nicht langsamer sein als das Laden aus dem bisherigen Format. Als Richtwert wird das Laden von 500 MB an Ultraschall-Rohdaten innerhalb von 3 Minuten verwendet. Außerdem soll es möglich sein, die Positionsdaten der Rohdaten innerhalb von 1,5 Minuten oder weniger zu laden.

3.3.5.2 Wartbarkeit

Um die Wartbarkeit zu gewährleisten sollen die folgenden Dokumente erstellt werden.

NF3 Dokumentation-Quellcode Ein aus den im Quellcode vorhandenen Klassen- und Methodenkommentaren bestehendes Dokument. Zweck dieses Dokumentes ist die Nachvollziehbarkeit des Programmcodes, was dazu dient, nach Abschluss dieses Projektes eventuell nötige Änderungen einfacher durchführen zu können.

NF4 Schnittstellen-Dokumentation Ein Dokument, das alle erstellten Schnittstellen beschreibt. Zweck dieses Dokumentes ist zum einen das einfachere Verwenden der Softwarebibliothek. Außerdem soll es aber auch der besseren Anpassbarkeit der Schnittstellen dienen.

3.3.5.3 Technische Anforderungen

Folgende technische Anforderungen sind vorgesehen.

NF5 Betreibssystem Wie bereits aus Abschnitt 3.3.3 hervorgeht, ist das Zielsystem der Software Windows 10 in der Version 1909, weitere System sind optional.

NF6 Kompatibilität Wie ebenfalls bereits aus Abschnitt 3.3.3 hervorgeht, sollen die innerhalb dieser Software implementierten Funktionalitäten in Anwendungen, die auf LabView in der Version 2018 oder .Net Framework in der Version 4.5 basieren, integrierbar sein.

3.3.6 Überprüfung der Anforderungen

Die Anforderungen werden nach Abschluss der Implementierungsphase überprüft. Die Überprüfung der funktionalen Anforderungen wird dabei durch die Integration der Funktionalitäten in bereits existierende Softwarelösungen realisiert. Die nichtfunktionalen Anforderungen werden, wo möglich, durch Messung überprüft. Die Anforderungen bezüglich Dokumentation werden überprüft, indem das Vorhandensein der entsprechenden Dokumente geprüft wird.

3.4 Use-Cases

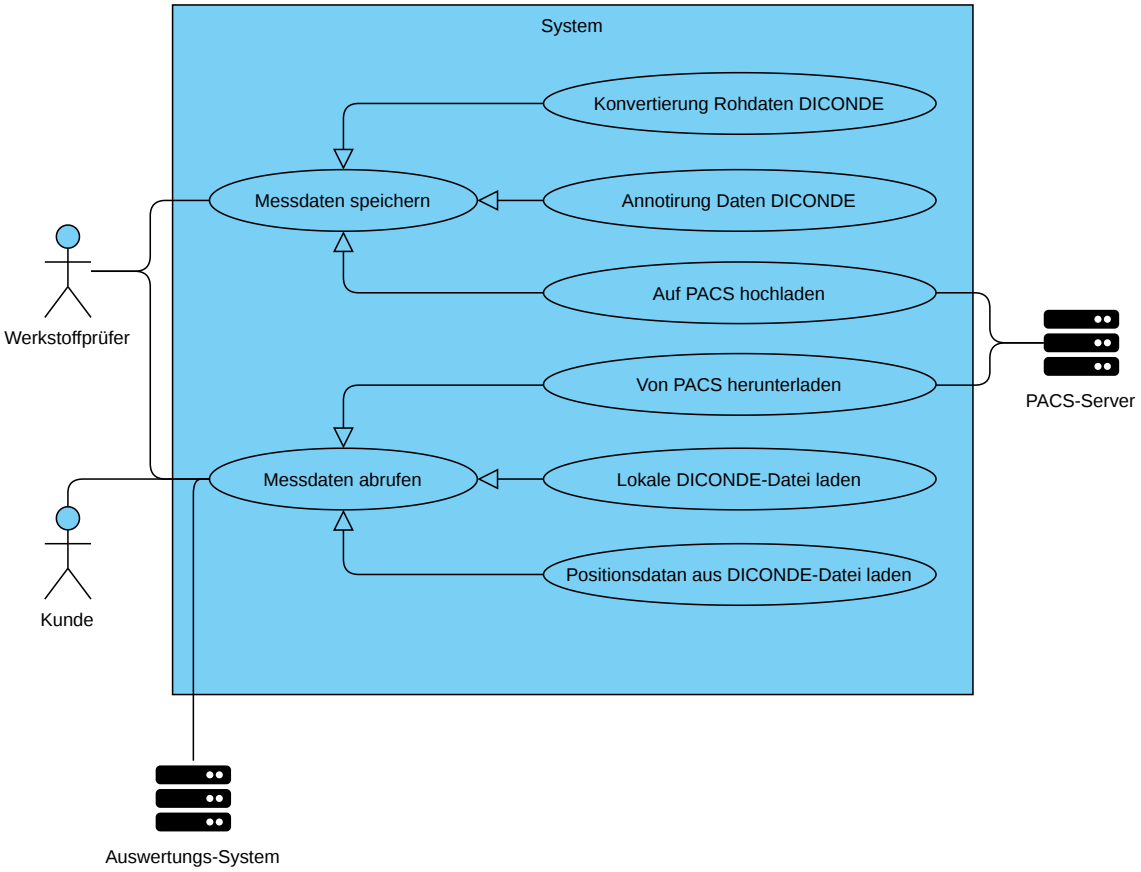

Abbildung 3.2: Übersicht der Use-Cases.

Wie in Abbildung 3.2 zu sehen, gibt es mehrere Akteure, die mit dem System interagieren. Die entsprechenden Akteure und eine kurze Erklärung dieser ist in der nachfolgenden Aufzählung dargestellt.

Werkstoffprüfer führen Messungen mittels zerstörungsfreier Prüfverfahren durch. Dabei ist im Kontext dieser Arbeit wichtig, dass diese Daten anschließend im DICONDE-Format abgelegt werden sollen.

Kunde Die Kunden rufen im Kontext dieser Arbeit von Werkstoffprüfern erstellte, im DICONDE-Format vorliegende, Messdaten ab.

Auswertungs-System ruft auf dem PACS-Server liegende Messdaten ab, um Auswertungen darauf auszuführen.

PACS-Server speichert und archiviert die im DICONDE-Format vorliegenden Daten. Darüber hinaus werden die abgelegten Daten über verschiedene Schnittstellen zum Abruf bereitgestellt.

Außerdem werden die in der Abbildung 3.2 dargestellten Use-Cases im Folgenden genauer aufgeschlüsselt.

Der folgende Use-Case 3.6 beschreibt den Fall, dass eine Messung abgespeichert werden soll. Dabei wird sowohl das Speichern auf einem PACS-Server, als auch das lokale Abspeichern der Messung abgebildet.

Name	Messdaten Speichern
Nummer	1
Ziel im Kontext	Erzeugte Messdaten werden gespeichert
Akteure	Werkstoffprüfer, PACS-Server
Trigger	Werkstoffprüfer speichert fertige Messung
Essenzielle Schritte	**1.** Werkstoffprüfer übergibt fertige Messung **2.** Werkstoffprüfer wählt aus, dass die Daten im DICONDE-Format abgelegt werden sollen **3.** Werkstoffprüfer wählt PACS-Server aus **4.** Werkstoffprüfer startet das Hochladen der Messung
Erweiterungen	**1a.** Werkstoffprüfer gibt zusätzliche Informationen zur Messung an
	3a. Werkstoffprüfer stellt fest, dass im Netzwerk kein PACS-Server verfügbar ist **3a1.** Werkstoffprüfer wählt aus dass die DICONDE-Datei lokal gespeichert werden soll

Tabelle 3.6: Use-Case Messdaten Speichern.

Der Use-Case 3.7 stellt die Konvertierung von Ultraschall-Rohdaten in das DICONDE-Format da. Der Use-Case hängt dabei mit dem Use-Case 3.6 zusammen, da die Daten nach der Konvertierung gespeichert werden können.

Name	Konvertierung Rohdaten DICONDE
Nummer	2
Ziel im Kontext	Übergebene Rohdaten werden in das DICONDE-Format konvertiert
Akteure	Werkstoffprüfer
Trigger	Werkstoffprüfer speichert fertige Messung
Essenzielle Schritte	**1.** Werkstoffprüfer übergibt fertige Messung **2.** Werkstoffprüfer wählt aus, dass die Daten im DICONDE-Format abgelegt werden sollen

Tabelle 3.7: Use-Case Konvertierung Rohdaten.

Der Use-Case 3.8 behandelt das Hinzufügen von zusätzlichen Informationen zu einer DICONDE-Datei. Dabei kann es sich zum Beispiel um die Position des Messkopfes für jeden durchgeführten A-Scan handeln. Der Use-Case hängt dabei mit dem Use-Case 3.6 zusammen, da die zusätzlichen Informationen ebenfalls in der DICONDE-Datei gespeichert werden müssen.

Name	Annotierung Daten DICONDE
Nummer	3
Ziel im Kontext	Daten die im Kontext zu der durchgeführten Messung beziehungsweise zu den Rohdaten stehen werden der DICONDE-Datei hinzugefügt
Akteure	Werkstoffprüfer
Trigger	Werkstoffprüfer speichert fertige Messung
Essenzielle Schritte	**1.** Werkstoffprüfer übergibt fertige Messung **2.** Werkstoffprüfer wählt aus, dass die Daten im DICONDE-Format abgelegt werden sollen

Tabelle 3.8: Use-Case Annotierung Daten.

Der Use-Case 3.9 beschreibt das Hochladen einer DICONDE-Datei auf einen PACS-Server. Der Use-Case hängt dabei mit dem Use-Case 3.6 zusammen, da eine Option beim Speichern der Messdaten darin besteht, die DICONDE-Datei auf einen PACS-Server hochzuladen.

Name	Auf PACS hochladen
Nummer	4
Ziel im Kontext	Eine neu erstellte DICONDE-Datei wird auf den PACS-Server hochgeladen
Akteure	Werkstoffprüfer, PACS-Server
Trigger	Werkstoffprüfer speichert fertige Messung und wählt PACS-Server aus
Essenzielle Schritte	**1.** Werkstoffprüfer übergibt fertige Messung **2.** Werkstoffprüfer wählt aus, dass die Daten im DICONDE-Format abgelegt werden sollen **3.** Werkstoffprüfer wählt PACS-Server aus **4.** Werkstoffprüfer startet das Hochladen der Messung
Erweiterungen	**1a.** Werkstoffprüfer gibt zusätzliche Informationen zur Messung an

Tabelle 3.9: Use-Case Auf PACS hochladen.

Dieser Use-Case 3.10 beschreibt das Abrufen von Messdaten. Die Messdaten werden dabei aus einer DICONDE-Datei gelesen. Die DICONDE-Datei kann dabei entweder von einem PACS-Server heruntergeladen werden oder aus einem lokalen Verzeichnis geladen werden.

Name	Messdaten abrufen
Nummer	5
Ziel im Kontext	Messdaten werden zur Sichtung, Auswertung oder Veränderung geladen
Akteure	Werkstoffprüfer, Kunde, Auswertungs-System, PACS-Server
Trigger	Werkstoffprüfer, Kunde oder ein Auswertungs-System will Messung laden
Essenzielle Schritte	**1.** Werkstoffprüfer, Kunde oder ein Auswertungs-System wählt zu ladende Messung aus
Erweiterungen	**1a.** Werkstoffprüfer, Kunde oder ein Auswertungs-System wählt einen PACS-Server aus **1a1.** Werkstoffprüfer, Kunde oder ein Auswertungs-System wählt welche Messung vom PACS-Server geladen werden soll
	1b. Werkstoffprüfer, Kunde oder ein Auswertungs-System wählt welche Messung vom lokalen System geladen werden soll

Tabelle 3.10: Use-Case Messdaten abrufen.

Der Use-Case 3.11 beschreibt das Herunterladen einer DICONDE-Datei von einem PACS-Server. Der Use-Case hängt dabei mit dem Use-Case 3.6 zusammen, da die aufzurufenden Daten eventuell von einem PACS-Server heruntergeladen werden müssen.

Name	Von PACS herunterladen
Nummer	6
Ziel im Kontext	Messdaten werden vom PACS-Server heruntergeladen
Akteure	Werkstoffprüfer, Kunde, Auswertungs-System, PACS-Server
Trigger	Werkstoffprüfer, Kunde oder ein Auswertungs-System will Messung laden
Essenzielle Schritte	**1.** Werkstoffprüfer, Kunde oder ein Auswertungs-System wählt einen PACS-Server aus **2.** Werkstoffprüfer, Kunde oder ein Auswertungs-System wählt welche Messung vom PACS-Server geladen werden soll

Tabelle 3.11: Use-Case Von PACS herunterladen.

Der Use-Case 3.12 beschreibt das Herunterladen einer DICONDE-Dateien aus einem lokalen Verzeichnis. Der Use-Case hängt dabei mit dem Use-Case 3.6 zusammen, da die aufzurufenden Daten eventuell von einem lokalen Verzeichnis geladen werden müssen.

Name	Lokale DICONDE-Datei laden
Nummer	7
Ziel im Kontext	Lokale Messdaten werden geladen
Akteure	Werkstoffprüfer, Kunde, Auswertungs-System
Trigger	Werkstoffprüfer, Kunde oder ein Auswertungs-System will Messung laden
Essenzielle Schritte	**1.** Werkstoffprüfer, Kunde oder ein Auswertungs-System wählt welche Messung vom lokalen System geladen werden soll

Tabelle 3.12: Use-Case Lokale DICONDE-Datei laden.

Der Use-Case 3.12 beschreibt das Laden von Positionsdaten aus einer DICONDE-Datei. Der Use-Case hängt dabei mit dem Use-Case 3.13 zusammen, da eventuell vorhandene Positionsdaten beim Abrufen der Daten mitgeladen werden können.

Name	Positionsdaten aus DICONDE-Datei laden
Nummer	8
Ziel im Kontext	Nur Positionsdaten der einzelnen Messungen werden geladen
Akteure	Werkstoffprüfer, Kunde, Auswertungs-System
Trigger	Werkstoffprüfer, Kunde oder ein Auswertungs-System will Positionsdaten einer Messung laden
Essenzielle Schritte	**1.** Werkstoffprüfer, Kunde oder ein Auswertungs-System wählt, für welche Messung Positionsdaten vom lokalen System geladen werden soll

Tabelle 3.13: Use-Case Positionsdaten aus DICONDE-Datei laden.

Der Use-Case 3.13 das Laden von Positionsdaten aus einer DICONDE-Datei.

3.5 Marktanalyse und Vergleich

Für die infrage kommenden DICOM Lösungen wird jeweils eine exemplarische Implementierung durchgeführt, um die Eignung der Lösung für diese Arbeit fundiert bestimmen zu können. Mehr dazu in den jeweiligen Abschnitten zu den DICOM Lösungen.

3.5.1 Auswahlkriterien

Die Entscheidung, welches Framework verwendet werden soll, wird anhand der folgenden Kriterien getroffen.

Codequalität Die Softwarequalität wird in der Form beurteilt, dass überprüft wird, ob die Struktur der Software leicht ersichtlich ist, ob Code-Kommentare vorhanden sind und ob eine Test-Suite vorhanden ist.

Dokumentation Die Qualität der vorhandenen Dokumentation spielt insofern eine Rolle, dass sie die Einarbeitung in das entsprechende Framework deutlich erleichtern kann. Somit kann das Vorhandensein einer hochwertigen Dokumentation dazu beitragen, die Implementierungsphase zu verkürzen und die Qualität des Endproduktes zu steigern. Außerdem wird dadurch die Wartbarkeit verbessert.

Abdeckung der benötigen DICOM-Funktionen Zum einen muss die Verarbeitung von DICOM Dateien möglich sein (Siehe Abschnitt 2.1.8), zum anderen müssen die in Abschnitt 2.1.5 beschriebenen DICOM Netzwerk Services beschrieben werden.

Programmiersprache Da im Fraunhofer IZFP hauptsächlich C++, C#, Python und Labview genutzt werden, wird eine dieser Sprachen präferiert. Eine auf einer anderen Sprache basierende Lösung wird nur bei fehlender Verfügbarkeit an Werkzeugen in den genannten Sprachen in Betracht gezogen. Des Weiteren ist zu erwähnen, dass Labview auf in C++ und C# geschriebene Programmbibliotheken zugreifen kann und somit bei der Wahl einer dieser beiden Sprachen ebenfalls abgedeckt ist.

Plattformunabhängigkeit Die Plattformunabhängigkeit des Frameworks ist deshalb wünschenswert, da die Software unter Umständen auf unterschiedlichen Betriebssystemen ausgeführt werden soll. Da die Zielplattform für diese Software allerdings Windows ist, ist dieses Kriterium optional.

Erweiterbarkeit Da die meisten verfügbaren Frameworks nur DICOM und nicht DICONDE unterstützen, ist es von Vorteil, wenn sich das Framework erweitern lässt, um die fehlende Funktionalität hinzuzufügen.

Wartbarkeit Da davon auszugehen ist, dass die Software in der Zukunft angepasst werden muss, zum Beispiel wenn sich der DICONDE-Standard ändert, ist es wichtig, dass das Framework die Entwicklung einer einfach wartbaren Software unterstützt.

Sofwarelizenz Auch die Lizenz des Frameworks spielt eine Rolle, da es eventuell wichtig ist, das Framework anzupassen. Außerdem ist es von Vorteil, wenn das Framework kostenfrei genutzt werden kann.

3.5.2 fo-DICOM

fo-DICOM ist das aktuell für die Implementierung von DICONDE genutzte Framework. Die aktuelle Implementierungen bringen allerdings einen kleinen Nachteil mit sich.

So ist das Framework eigentlich zur Implementierung von DICOM gedacht. Es wird aber firmenintern, mit Hilfe von Private-Tags, für die Implementierung von DICONDE erweitert. Auch ist die Implementierung mittels dieses Frameworks etwas unübersichtlicher, da die benötigten Tags einzeln, wie im Listing 3.1 beispielhaft zu sehen, zu einem Dataset (2.1.9) hinzugefügt werden müssen.

Listing 3.1: fo-DICOM Beispiel

```
// Module - Patient
// Module - Patient - Type 1 attributes
// Module - Patient - Type 2 attributes
dataset.Add(DicomTag.PatientID, "A patient id");
dataset.Add(DicomTag.PatientName, "Example file^Raw Data Storage with
    private tags");
dataset.Add(DicomTag.PatientBirthDate, string.Empty);
dataset.Add(DicomTag.PatientSex, string.Empty);

// Module - General Study
// Module - General Study - Type 1 attributes
dataset.Add(DicomTag.StudyInstanceUID, DicomUID.Generate());
// Module - General Study - Type 2 attributes
dataset.Add(DicomTag.StudyDate, DateTime.Now.ToString("yyyyMMdd"));
dataset.Add(DicomTag.StudyTime, DateTime.Now.ToString("HHmmss"));
dataset.Add(DicomTag.ReferringPhysicianName, string.Empty);
dataset.Add(DicomTag.StudyID, "0");
dataset.Add(DicomTag.AccessionNumber, string.Empty);
```

Exemplarische Implementierung Da die eventuelle Nutzung von fo-DICOM in diesem Projekt so aussehen würde, dass mit Hilfe von fo-DICOM die benötigten DICONDE Funktionalitäten implementiert werden soll, wird eine exemplarische Implementierung des in Abbildung 3.4 dargestellten US-Image durchgeführt. Da fo-DICOM keine Schnittstelle zum Implementieren einer IOD anbietet, müssen die benötigten Data Elements (Siehe Abschnitt 2.1.10) im Code aufgelistet und separat gesetzt werden.

Einschätzung

Codequalität Da Code-Kommentare vorhanden sind, die Klassenstruktur leicht zu verstehen ist und Unit-Tests vorhanden sind, wird die Codequalität als gut bewertet.

Dokumentation Da eine umfangreiche Dokumentation [42] vorhanden ist und außerdem Codebeispiele für die wichtigsten Funktionalitäten vorhanden sind [41], wird die Dokumentation als sehr gut bewertet.

Abdeckung der benötigen DICOM-Funktionen Alle für diese Arbeit notwendigen DICOM Funktionalitäten sind von fo-DICOM abgedeckt.

Programmiersprache Im Rahmen dieser Arbeit wird als sehr positiv beurteilt, dass fo-DICOM mit der Programmiersprache C# entwickelt wurde. Dadurch lässt es sich voraussichtlich einfach in die bereits vorhandenen Softwarelösungen integrieren.

Plattformunabhängigkeit Fo-Dicom kann unter andrem mit .NET Core übersetzt werden. Somit werden die wichtigsten Windows und Linux Betriebssystem unterstützt. [42]

Erweiterbarkeit Da es sich um ein Open-Source-Projekt handelt, ist die Erweiterbarkeit grundsätzlich gegeben. Eine Schwierigkeit stellt allerdings dar, dass fo-DICOM keine Schnittstelle anbietet um eine IOD zu implementieren.

Wartbarkeit Die Wartbarkeit wird insbesondere wegen der hohen Codequalität als gut bewertet.

Sofwarelizenz Fo-Dicom steht unter der Microsoft Public License (MS-PL) [41] und darf somit kostenfrei genutzt werden.

Abschließend lässt sich daher zu fo-DICOM sagen, dass sowohl die allgemeine Qualität als gut zu beurteilen ist, als auch die Kompatibilität mit dieser Arbeit, da alle benötigten DICOM Funktionalitäten abgedeckt werden und die Programmiersprache C# verwendet wird. Das einzige Problem im Zusammenhang mit dieser Arbeit stellt allerdings dar, dass fo-DICOM keine Schnittstelle zum Implementieren neuer IODs anbietet. Dies kann den Arbeitsaufwand zum Implementieren der benötigten Funktionalitäten eventuell erhöhen.

3.5.3 DCMTK

Bei der Software DICOM-Toolkit (DCMTK) handelt es sich um eine Open-Source Implementierung großer Teile des DICOM-Standards in C++. Da es sich jedoch um eine reine DICOM-Implementierung handelt, muss das Framework erst so erweitert werden, dass es auch den DICONDE-Standard unterstützt. [14]

Vorteilhaft ist dabei, dass DCMTK modular aufgebaut ist, sprich, dass es etwa zu jedem DICOM-Modul eine Klasse gibt, die eben diese Modul abbildet.

Wenn man also beispielsweise eine Instanz einer IOD vom Typ CT-Image [5, 620ff] erstellen will, würde das dazugehörige Klassendiagramm wie in Abbildung 3.3 aussehen. Dabei ist zu beachten, dass es sich hier um eine vereinfachte Darstellung handelt, die einen Überblick geben soll. Deshalb wurde darauf verzichtet, die Methoden und Attribute darzustellen.

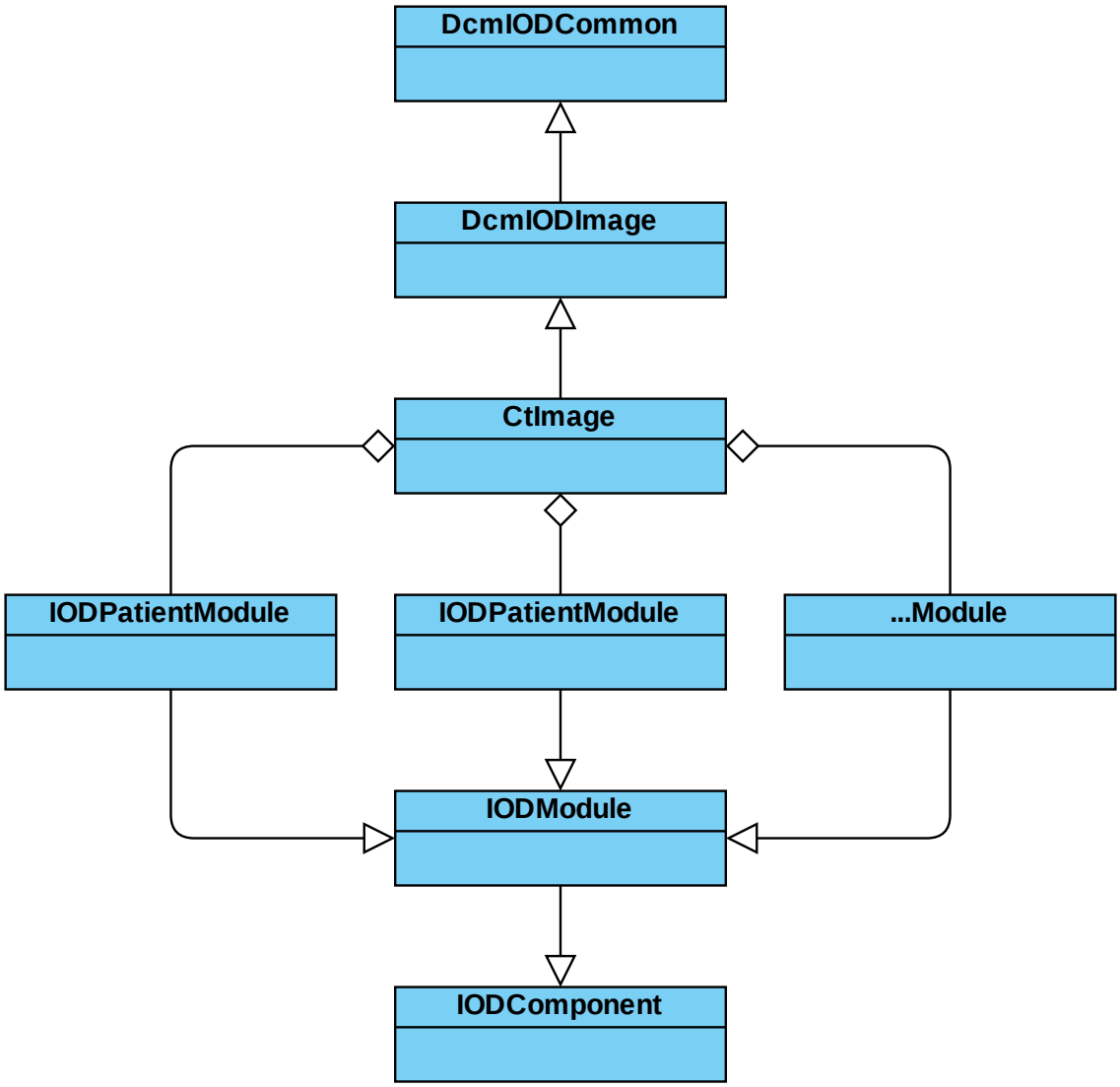

Abbildung 3.3: Klassendiagramm zur Abbildung der DICOM-IOD CT-Image in DCMTK.

Wichtig ist dabei, dass die Interfaceklassen „DcmIODCommon“ und „DcmIODImage“ sowie die Klassen „IODModule“ „IODCOmponent“ und die Klassen, die die einzelnen Module repräsentieren, wie etwa „IODPatientModule“ bereits in DCMTK enthalten sind. Es müssen also nur die Klasse „CTImage“ sowie eventuell nicht vorhandene Module implementiert werden.

Für die Erweiterung auf DICONDE bedeutet dies, dass große Teile der bestehenden Implementierung wiederverwendet werden könne und lediglich die Klassen zur Abbildung der IODs und der benötigten Module implementiert werden müssen. Dies ist beispielhaft für die DICONDE IOD Us-Image in der Abbildung 3.4 zu sehen. Dabei sind dieselben Hinweise wie bereits bei der Abbildung 3.3 zu beachten.

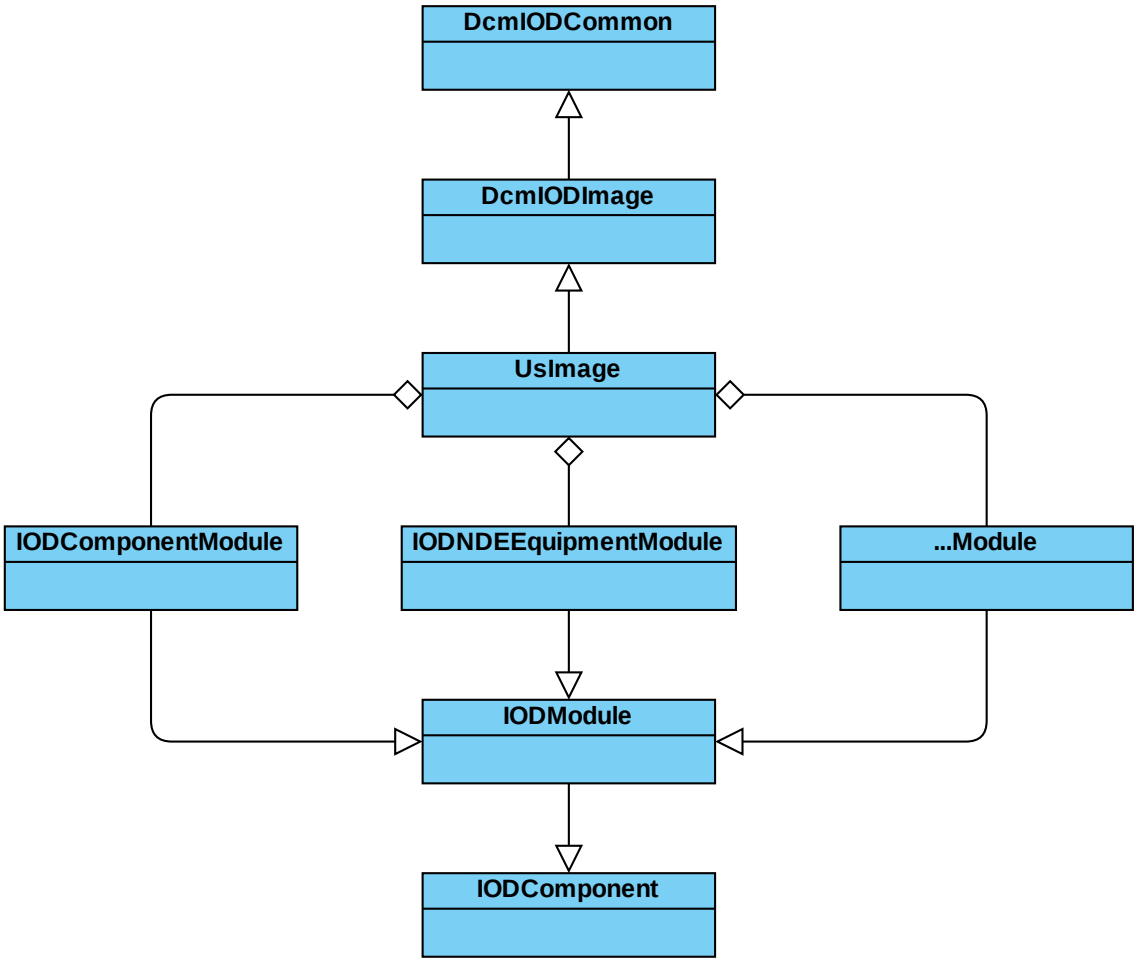

Abbildung 3.4: Klassendiagramm zur Abbildung der DICONDE-IOD US-Image in DCMTK.

Außerdem bietet DCMTK noch eine weitere Besonderheit auf die in den Folgenden eingegangen wird.

Mit dem Konzept der Regeln DCMTK bietet die Möglichkeit, für jedes DICOM Attribut Regeln zu definieren, um bereits zur Laufzeit zu erkennen, wenn ungültige Werte gesetzt werden und dieses zu verhindern. Die Regeln beziehen sich dabei auf die VR und VM des jeweiligen Attributes. Die Implementierung der Regeln erfolgt durch die in der Abbildung 3.5 dargestellte Klasse „IODRule“.

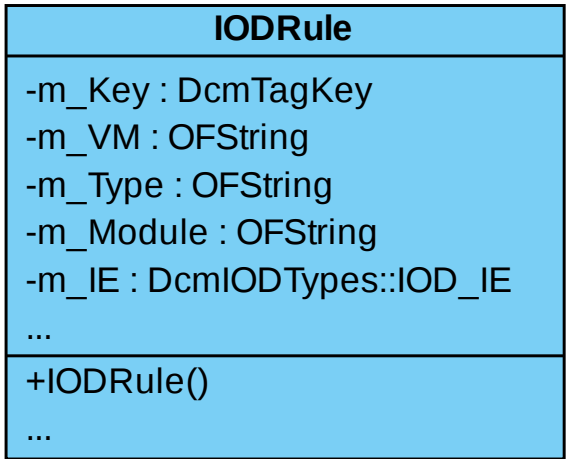

Abbildung 3.5: Klassendiagramm der DCMTK Klasse IODRule

Exemplarische Implementierung Da die eventuelle Nutzung von DCMTK in diesem Projekt so aussehen würde, dass DCMTK um die DICONDE Funktionalitäten erweitert werden soll, wird eine exemplarische Implementierung des in Abbildung 3.4 dargestellten US-Image durchgeführt.

Dazu musste zum einen für jedes in der IOD (Siehe Abschnitt 2.1.4) US-Image [31] vorhandenen Modul eine Klasse implementiert werden, die diese abbildet. Dabei muss man von der DCMTK Klasse „IODModule“ ableiten, um die in DCMTK vorhandene Funktionalitäten bezüglich der Module nutzen zu können.

In der jeweiligen Klasse müssen zum anderen die Regeln implementiert werden, die abbilden, welches Tag, welche VR, welche VM und welchen Type die jeweiligen Attribute des Moduls besitzen. (Siehe Abschnitt 2.1.10) Dies wird umgesetzt indem, für jedes Attribut ein Objekt vom Typ „IODRule“ erzeugt wird. Im Listing 3.2 ist dies beispielhaft für das „NDE Equipment Module“ [33, S. 7] zu sehen.

Listing 3.2: Rules für NDE Equipment Module

```
void NdeEquipmentModule::resetRules()
{
        // parameters are tag, VM, type.
        m_Rules->addRule(new IODRule(DCM_SoftwareVersion, "
            1-n", "1", getName(), DcmIODTypes::IE_COMPONENT)
            , OFTrue);
        m_Rules->addRule(new IODRule(DCM_Manufacturer, "1",
             "3", getName(), DcmIODTypes::IE_COMPONENT),
            OFTrue);
        m_Rules->addRule(new IODRule(DCM_CompanyName, "1",
            "3", getName(), DcmIODTypes::IE_COMPONENT),
            OFTrue);
        m_Rules->addRule(new IODRule(DCM_CompanyAddress, "1
            ", "3", getName(), DcmIODTypes::IE_COMPONENT),
            OFTrue);
        m_Rules->addRule(new IODRule(DCM_StationName, "1",
            "3", getName(), DcmIODTypes::IE_COMPONENT),
            OFTrue);
        m_Rules->addRule(new IODRule(DCM_DepartmentName, "1
            ", "3", getName(), DcmIODTypes::IE_COMPONENT),
            OFTrue);
        m_Rules->addRule(new IODRule(
```

```
                DCM_ManufacturersModelName, "1", "3", getName(),
                 DcmIODTypes::IE_COMPONENT), OFTrue);
            m_Rules->addRule(new IODRule(DCM_DeviceSerialNumber
                , "1", "3", getName(), DcmIODTypes::IE_COMPONENT
                ), OFTrue);
            m_Rules->addRule(new IODRule(DCM_ScannerId, "1", "3
                ", getName(), DcmIODTypes::IE_COMPONENT), OFTrue
                );
            m_Rules->addRule(new IODRule(DCM_SpatialResolution,
                 "1", "3", getName(), DcmIODTypes::IE_COMPONENT)
                , OFTrue);
            m_Rules->addRule(new IODRule(
                DCM_DateOfLastCalibration, "1-n", "3", getName()
                , DcmIODTypes::IE_COMPONENT), OFTrue);
            m_Rules->addRule(new IODRule(
                DCM_TimeOfLastCalibration, "1-n", "3", getName()
                , DcmIODTypes::IE_COMPONENT), OFTrue);
            m_Rules->addRule(new IODRule(DCM_PixelPaddingValue,
                 "1", "3", getName(), DcmIODTypes::IE_COMPONENT)
                , OFTrue);
    }
```

Die Rules sind dabei wie folgt aufgebaut.

Der erste Übergabeparameter bestimmt den Tag des Attributes, dieses ist im Listing nicht eindeutig zu erkennen, da Makros verwendet werden. „DCM_Softwareversion“ etwa stellt den Tag (0x0018, 0x1020) da. Der zweite Parameter bestimmt die VM des Attributes. Der nächste Parameter bestimmt den Type des Attributes. Die folgenden Parameter sind an dieser Stelle weniger relevant und bestimmen die interne Zuordnung des Attributes.

Außerdem müssen innerhalb des jeweiligen Modul Getter und Setter Methoden für jedes Attribut implementiert werden. In Listing 3.3 beispielhaft für das Tag „CompanyName“ zu sehen.

Listing 3.3: Getter und Setter für NDE Equipment Module

```
    OFCondition NdeEquipmentModule::getCompanyName(OFString &
        value, const signed long pos) const
    {
            return DcmIODUtil::getStringValueFromItem(
                DCM_CompanyName, *m_Item, value, pos);
    }

    OFCondition NdeEquipmentModule::setCompanyName(const
        OFString & value, const OFBool checkValue)
    {
            OFCondition result = (checkValue) ? DcmLongString::
                checkStringValue(value, "1") : EC_Normal;
            if (result.good())
            result = m_Item->putAndInsertOFStringArray(
                DCM_CompanyName, value);
            return result;
    }
```

Zum anderen muss für jede IOD eine Klasse implementiert werden, die die zur IOD gehörenden Module kapselt.

Einschätzung Im Bezug auf die im Kapitel 3.5.1 erläuterten Kriterien wird die Eignung von DCMTK für das aktuelle Projekt wie folgt eingeschätzt.

Codequalität Zum eine sind generell sprechende Klassen und Methodennamen sowie verständliche Kommentare vorhanden, was die Einarbeitung deutlich erleichtert. Zum anderen sind Unit-Tests vorhanden, die Integration von neuem Code einfacher machen, da so eventuelle Fehler einfacher auffallen.

Dokumentation Ein umfängliche Dokumentation mit Programmbeispielen ist vorhanden. [13]

Abdeckung der benötigen DICOM-Funktionen Alle für diese Arbeit relevanten Funktionalitäten sind abgedeckt, insbesondere die Netzwerkfunktionalitäten. [12]

Programmiersprache Die Programmiersprache, in der DCMTK entwickelt ist, stellt ein Problem dar, da es sich um C++ handelt. Da aber die bisher vorhandene Software zum größten Teil in C# einwickelt ist, müsste zusätzlich zu der Erweiterung von DCMTK noch ein Wrapper implementiert werden, der die Kompatibilität mit den bereits existierenden Softwarelösungen sicherstellt.

Plattformunabhängigkeit DCMTK kann für die wichtigsten Betriebssysteme übersetzt werden. Darunter Windows und Linux. [14]

Erweiterbarkeit Die Erweiterbarkeit ist grundsätzlich gegeben, da es sich um ein OpenSource-Projekt handelt. Außerdem wurde die Architektur von DCMTK bereits so entworfen, dass es ohne Probleme möglich ist, neue IODs zu implementieren.

Wartbarkeit Der Punkt Wartbarkeit könnte bei DCMTK eventuell ein Problem darstellen. Um die DICONDE-Erweiterung in die offizielle Version von DCMTK einzubringen, müsste dies von den Maintainern von DCMTK akzeptiert werden. Sollte dies nicht gelingen, würde das Problem entstehen, dass die DICONDE-Erweiterung für jede neue Version von DCMTK angepasst werden muss.

Sofwarelizenz Die Software darf frei verwendet werden. [11]

Abschließend lässt sich daher feststellen, dass DCMTK für den Anwendungszweck generell geeignet wäre und die Möglichkeit der Erweiterung auf DICONDE gegeben ist, allerdings ist bei der Verwendung von DCMTK davon auszugehen, dass ein erhöhter Aufwand betrieben werden muss, um die Software mit den bereits vorhandenen Softwaresystemen kompatibel zu machen.

3.5.4 Dcm4che

Bei Dcm4che handelt es sich um ein in Java geschriebenes DICOM-Framework. Da es sich auch hier um ein reines DICOM-Framework handelt, dass keine DICONDE Funktionalitäten unterstützt, müsste auch Dcm4che erst entsprechend erweitert werden. [17] Wie eine solche Erweiterung durchgeführt von Dcm4che durchgeführt wird, wird im Folgenden beschrieben.

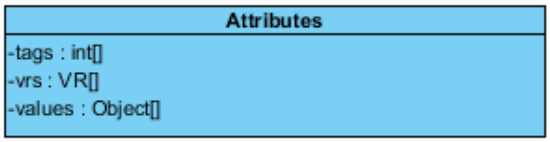

Abbildung 3.6: Vereinfachtes Klassendiagramm der Klasse Attributes

In der Abbildung 3.6 ist eine vereinfachte Darstellung der Klasse „Attributes“ zu sehen. Dabei wurden sämtliche Methoden und für diese Erklärung nicht relevanten Attribute weggelassen.

Die relevanten Attribute werden im Folgenden aufgezählt und erläutert.

tags Array das eine Liste von Tags beinhaltet.

vrs Array das die VRs zu den Tags beinhaltet.

values Array das die Werte zu den Tags beinhaltet.

Exemplarische Implementierung Da die eventuelle Nutzung von Dcm4Che in diesem Projekt so aussehen würde, dass Dcm4Che um die DICONDE Funktionalitäten erweitert werden soll, wird eine exemplarische Implementierung des in Abbildung 3.4 dargestellten US-Image durchgeführt.

Die exemplarische Implementierung besteht bedingt durch den Aufbau von Dcm4Che allerdings nur aus einer Klasse, die die benötigten Tags gekapselt in einem Attribut vom Typ „Attributes “ beinhaltet. Außerdem sind noch einige Methoden zum Abfragen und setzen der Werte vorhanden.

Einschätzung

Codequalität Dcm4Che weist eine schwer zu durchblickende Klassenstruktur auf und es sind außerdem kaum Kommentare vorhanden. Außerdem gestaltet sich die Implementierung einer neuen IOD, wie an der exemplarische Implementierung zu sehen, unübersichtlich. Daher wird die Codequalität als unterdurchschnittlich bewertet.

Dokumentation Es existiert zwar eine Dokumentation zu Dcm4Che, diese ist aber sehr lückenhaft und deckt etwa die DICOM Netzwerkdienste nur teilweise ab. [15]

Abdeckung der benötigen DICOM-Funktionen Es sind Funktionalitäten zum Verwenden der benötigten DICOM-Funktionen, wie etwa der DICOM Netzwerkdienste vorhanden, allerdings konnte die Eignung wegen lückenhafter Dokumentation nicht abschließend festgestellt werden.

Programmiersprache Dcm4Che ist in der Programmiersprache Java entwickelt. Generell ist dies als positiv zu bewerten, da Java als eine der weit verbreitetsten Programmiersprachen gilt [10], allerdings würde sich eine Integration in die größtenteils in C# einwickelten, breites vorhandenen Softwarelösungen sehr aufwendig bis unmöglich gestalten.

Plattformunabhängigkeit Da Dcm4Che in Java entwickelt ist, ist es auf den meisten Windows und Linux Betriebssystemen lauffähig. [37]

Erweiterbarkeit Da es sich um ein OpenSource-Projekt handelt, ist die Erweiterbarkeit generell gegeben.

Wartbarkeit Da der Quellcode von Dcm4Che kaum kommentiert ist und die Klassenstruktur nur schwer nachzuvollziehen ist, wird die Wartbarkeit als schwierig eingeschätzt. Als positiver Punkt ist allerdings das Vorhandensein von Unit-Tests zu nennen.

Sofwarelizenz Die Software darf frei verwendet werden. [16]

Abschließend lässt sich zu Dcm4Che sagen, dass es für diese Arbeit nicht geeignet ist. Insbesondere wegen des Aufwands, der betrieben werden müsste, um die Java Software in die bereits vorhandenen C# Softwarelösungen zu integrieren. Aber auch wegen der geringen Codequalität und lückenhaften Dokumentation.

3.5.5 Entscheidung Framework

Nach Betrachtung der verschiedenen zur Auswahl stehenden Frameworks zur Implementierung der geplanten Funktionalitäten wird fo-DICOM gewählt. Zum einen erfüllt es alle erforderlichen Kriterien, zum anderen ist fo-DICOM in der Programmiersprache C# implementiert. Da auch die meisten der bereits bestehenden Softwarelösungen in C# implementiert sind, ist bei der Verwendung von fo-DICOM ein geringerer Aufwand zu erwarten, was die Kompatibelmachung mit den vorhandenen Lösungen angeht. Dies ist auch der Hauptgrund, weshalb DCMTK nicht zur Umsetzung dieses Projektes verwendet wird, obwohl es einige Vorteile bieten würde.

4 Architektur

Ín diesem Kapitel wird die Ausarbeitung der Architektur dargestellt. Dies wird auf Basis der im vorherigen Kapitel festgelegten Anforderungen erstellt.

Um die im Kapitel 3.3 festgelegten Anforderungen erfüllen zu können, muss die Architektur der Software die folgenden Bedingungen erfüllen.

Flexible Integration Es soll möglich sein, die Software in verschiedene, bereits existierende Softwarelösungen zu integrieren. Mögliche Schwierigkeiten können dabei dadurch entstehen, dass die bereits vorhandenen Softwarelösungen auf unterschiedlichen Technologien aufbauen. So baut etwa die in 2.4.3 beschriebene Softwarelösung auf der in 2.5.1 beschriebene Systementwicklungssoftware LabView auf. Die in 2.4.2 beschriebenen Softwarelösung zur Durchführung und Auswertung von Ultraschallmessungen hingegen basiert auf dem in 2.5.2 beschriebenen .NET Framework und der Programmiersprache C#.

Spezialisierung auf DICONDE Die Architektur soll auf die Verwendung von DICONDE spezialisiert sein. Insbesondere bedeutet das, dass es nicht notwendig ist DICONDE-Tags durch DICOM-Tags zu ersetzen.

Wartbarkeit Ein besonderer Fokus soll auf die Wartbarkeit der Software gelegt werden, um diese z. B. bei Änderungen im DICONDE-Standard einfach anpassen zu können.

4.0.1 Ursprüngliche Softwarearchitektur

Da zu Beginn dieser Arbeit DICONDE bereits institutsintern genutzt wird, existiert bereits eine Softwarearchitektur. Der Aufbau dieser Softwarearchitektur ist in der Abbildung 4.1 zu sehen.

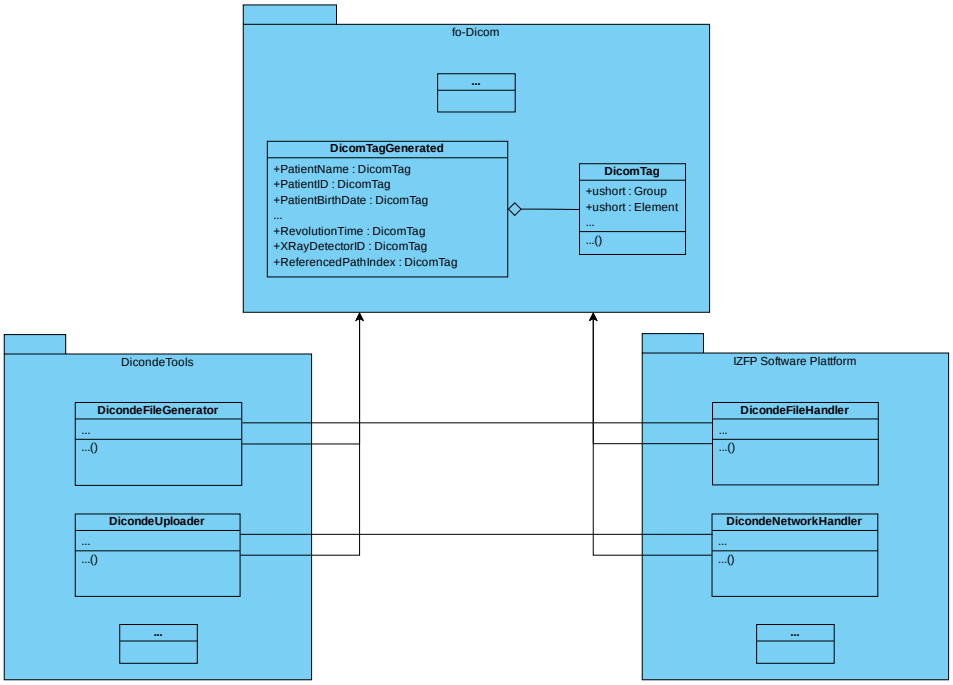

Abbildung 4.1: Softwarearchitektur zu Beginn der Arbeit

Dabei ist zu beachten, dass es sich hierbei um eine skizzenhafte Darstellung der Softwarearchitektur handelt. Es werden nur die Komponenten dargestellt, die zur Veranschaulichung der Softwarearchitektur notwendig sind. Außerdem existieren neben der Ultraschall-Software-Plattform (USP) noch weitere Programme, die dieselbe oder ähnliche Abhängigkeiten aufweisen.

Die in der Abbildung 4.1 skizzierte Softwarearchitektur bringt allerdings folgende Problematiken mit sich:

Doppelte Abhängigkeit von fo-DICOM Das jeweilige konsumierende Programm, in 4.1 beispielsweise USP, ist einmal von fo-DICOM direkt abhängig und einmal transitiv über 'DicondeTools'.

Überschneidungen der Tags Da fo-DICOM nur für die Verwendung von DICOM und nicht für die Verwendung mit DICONDE vorgesehen ist und es zwischen DICOM und DICONDE Überschneidungen gibt, was die Verwendung der Tag-Nummer angeht, kommt es zu folgendem Problem.

Um ein Tag zu verwenden, das in DICOM und DICONDE verwendet wird, muss zunächst im DICONDE-Standard nachgeschaut werden, welche Tag-Nummer das entsprechende Feld verwendet. Daraufhin muss im DICOM-Standard das Feld mit der entsprechenden Tag-Nummer herausgesucht und verwendet werden.

Nicht zentrale Funktionalitäten Einige der verwendeten Funktionalitäten werden von der eingesetzten Software selbst implementiert, was zu Inkonsistenzen führen könnte. Außerdem erhöht es den Wartungsaufwand der Softwarelandschaft.

4.0.2 Neue Softwarearchitektur

Ziel der neuen Architektur soll es sein, so viele Funktionalitäten wie möglich in einer zentralen Bibliothek zu implementieren. So soll vermieden werden, dass die gleiche Funktionalität doppelt implementiert werden muss. Außerdem soll so die Wartbarkeit und Konsistenz der Softwarelandschaft erhöht werden.

Die Lösung wird dabei als Softwarebibliothek umgesetzt, die die folgenden Funktionalitäten bereitstellt. Diese Funktionalitäten beruhen dabei auf den im Kapitel 3.3.4 festgelegten Anforderungen und sind im Detail wie folgt umgesetzt.

Erzeugen von DICONDE-Datein aus gerenderten Bildern Es gibt die Möglichkeit, fertig gerenderte Bilder, wie etwa von einer visualisierten Ultraschallmessung zusammen mit den dazugehörigen Informationen in eine DICONDE-Datei zu schreiben.

Erzeugen von DICONDE-Datein aus Ultraschall Rohdaten Außerdem besteht die Möglichkeit, Ultraschall-Rohdaten zusammen mit den entsprechenden Positionsdaten der Messungen in das DICONDE Format zu bringen.

Laden von Daten aus DICONDE-Dateien Des Weiteren besteht die Möglichkeit, die erzeugten Daten wieder einzulesen. Beim Laden der Rohdaten ist es außerdem möglich, nur die gespeicherten Positionsdaten zu laden, um lange Ladezeiten zu umgehen, falls die eigentlichen Messdaten nicht benötigt werden.

Upload von DICONDE-Dateien auf einen PACS-Server Die erzeugten DICONDE-Daten können auch auf einen PACS-Server hochgeladen werden.

Dowload von DICONDE-Dateien von einen PACS-Server Außerdem können auch DICONDE-Daten vom Server heruntergeladen werden.

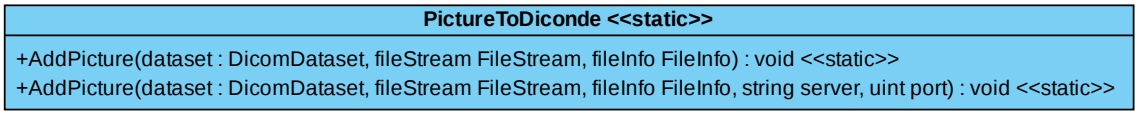

Abbildung 4.2: Schnittstelle zum Erstellen einer DICONDE-Datei aus einem Bild

Konvertieren von Bildern In Abbildung 4.2 ist die Schnittstelle zum Erstellen einer DICONDE-Datei aus einem Bild dargestellt. Dabei ist zu beachten, dass die Methode 'AddPicture' überladen ist.

Der ersten Version der Methode muss zum einen ein 'DicomDataset' übergeben werden, das zusätzliche Informationen enthält. Zum anderen muss ein 'FileStream' auf das zu verwendende Bild und ein Objekt vom Typ 'FileInfo', das den Pfad zum lokalen Speichern des Bildes enthält, übergeben werden. Der zweiten Version der Methode müssen die selben Parameter und zusätzlich noch der String 'server' und der uint 'port' übergeben werden. Das hat folgenden Hintergrund.

Innerhalb einer DICOM\DICONDE-Serie sollten die enthaltenen Bilder durchnummeriert sein. [5, S. 513] Für den Fall, dass eine bereits bestehende Serie, die schon auf einen PACS-Server hochgeladen wurde, aktualisiert werden soll, muss die Nummerierung der Bilder mit dem PACS-Server abglichen werden, um Inkonsistenzen zu vermeiden.

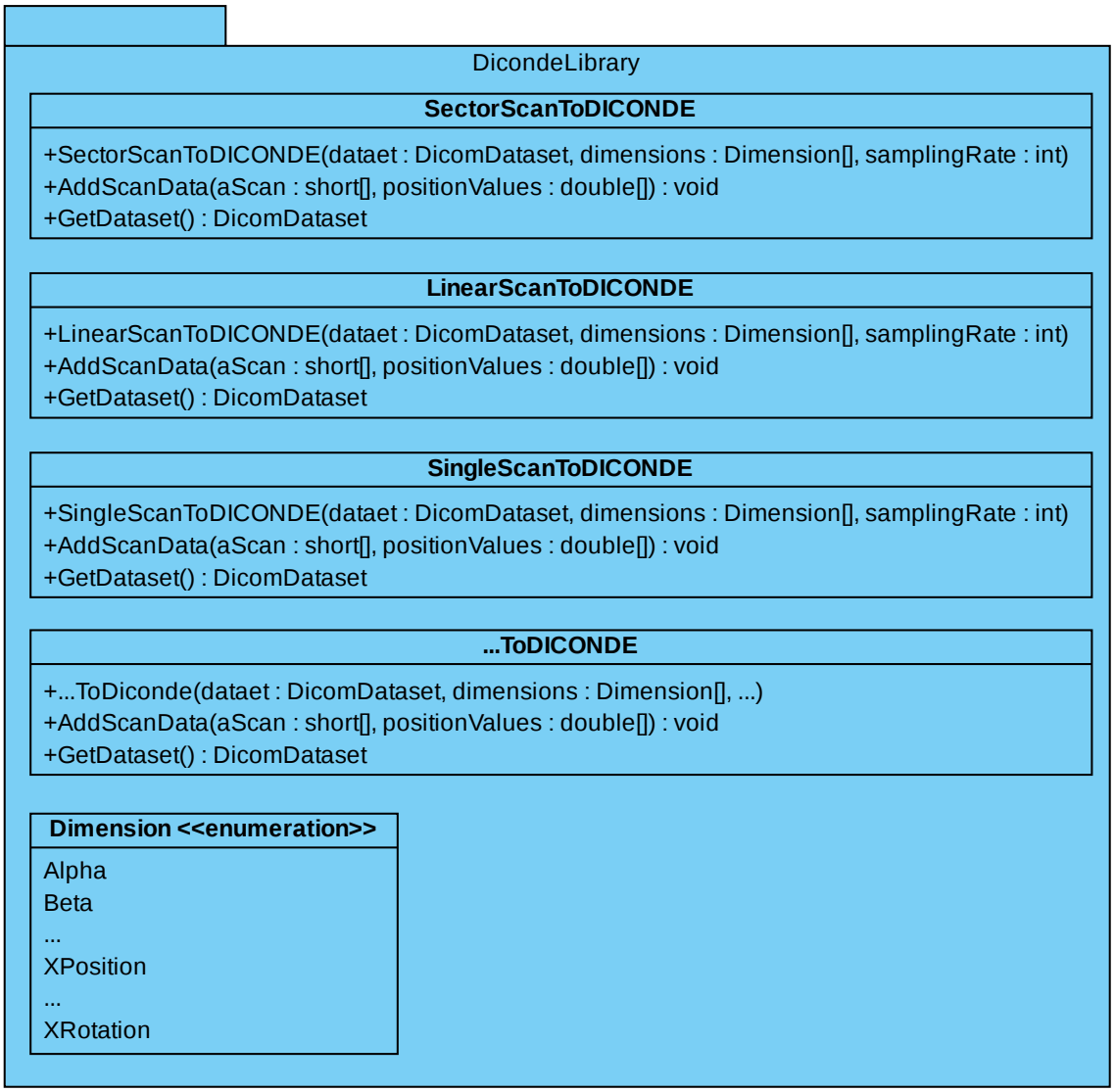

Abbildung 4.3: Schnittstelle zum Erstellen einer DICONDE-Datei Ultraschall-Rohdaten

Konvertieren von Rohdaten In Abbildung 4.3 ist die Schnittstelle zum Erstellen von DICONDE-Dateien aus Ultraschall-Rohdaten zu sehen. Diese Schnittstelle ist wie folgt aufgebaut.

Für jeden Scan-Typ existiert eine Klasse. Um eine DICONDE-Datei aus Ultraschall-Rohdaten zu erstellen, muss ein Objekt der passenden Klasse erstellt werden. Dem Konstruktor muss zum einen ein 'DicomDataset' übergeben werden, dem die Daten hinzugefügt werden sollen. Außerdem muss ein Array von Enumerationen des ebenfalls in Abbildung 4.3 zu sehenden Typs 'Dimension' übergeben werden. Anhand dieses Arrays wird festgelegt, welche Raumdimensionen zu den A-Scans abgelegt werden sollen. Dabei ist zu beachten, dass für einige Scan-Typen bestimmte Dimensionen unbedingt erforderlich sind. Sollte einer dieser erforderlichen Scan-Typen nicht mit übergeben werden, wird eine Ausnahme ausgelöst. Diese Ausnahme ist jedoch in Abbildung 4.3 nicht dargestellt, um die Abbildung übersichtlicher zu halten. Letztlich müssen noch vom Scan-Typ abhängige Parameter, wie zum Beispiel die 'SamplingRate', übergeben werden.

Nachdem das Objekt erstellt ist, können mithilfe der Methode 'AddScanData' Rohdaten hinzugefügt werden. Die eigentlichen Rohdaten sind dabei im Short-Array 'aScan' enthalten. Im Double-Array 'positionValues' müssen die Werte zu den im Konstruktor

übergebenen Raumdimensionen enthalten sein. Sollte die Anzahl der Werte nicht mit denen der Dimensionen übereinstimmen, wird ebenfalls eine in Abbildung 4.3 nicht dargestellte Ausnahme ausgelöst.

Laden von Daten aus DICONDE Bezüglich der Architektur des Ladens von Rohdaten aus dem DICONDE-Format, können Metainformationen zu der jeweiligen Messung separat von den Rohdaten geladen werden.

Up- und Download Die Architektur des Up- und Downloads kapselt die entsprechenden Funktionalitäten in unterschiedlichen Klassen, da für den jeweiligen Fall unterschiedliche Technologien zum Einsatz kommen. Der Upload wird über die im Abschnitt 2.1.5 beschriebenen DICOM-Services realisiert und der Download über die in 2.1.6 beschriebene WADO-RS-Schnittstelle.

5 Implementierung

In diesem Kapitel wird die eigentliche Implementierung der Bibliothek sowie die Integration in bereits vorhandene Software beschrieben. Außerdem wird die Software zum Generieren der DICONDE-Tags beschrieben.

5.1 DICONDE-Bibliothek

Bei der DICONDE-Bibliothek handelt es sich um die zentrale Implementierung dieser Arbeit. In ihr werden die zentralen Funktionalitäten implementiert, die dann in andere Softwareprodukte integriert werden.

5.1.1 Ablegen von Rohdaten

In diesem Kapitel wird beschrieben, wie das Ablegen von Rohdaten in DICOM/DICONDE realisiert wird.

Da, wie im Kapitel 2.3 beschrieben, verschiedene Scan-Verfahren existieren, muss zunächst eine Entscheidung getroffen werden, mithilfe welchen Verfahren die Daten erzeugt wurden. Diese Entscheidung ist wichtig, da je nach Scan-Verfahren andere Metadaten zu den Rohdaten abgelegt werden müssen und/oder die Rohdaten anders zugeordnet beziehungsweise gruppiert werden müssen.

Die Daten zu den verschiedenen Scan-Verfahren sind dabei wie folgt zu strukturieren:

Linear Scan Bei einem Linear Scan muss sowohl die Position des Prüfkopfes für jeden Schuss, als auch mit welchen Elementen des Prüfkopfes geschossen wird, gespeichert werden. Die Position des Prüfkopfes wird dabei durch die X-, Y- und Z-Koordinate und einen Winkel definiert. Um abzubilden, mit welchen Elementen geschossen wird, wird gespeichert, wie viele Elemente pro Schuss verwendet werden und welches Element das erste des Schusses ist. Außerdem muss die Schrittweite gespeichert werden.

Sector Scan Bei einem Sector Scan muss die Position des Prüfkopfes, in Form von X-, Y- und Z-Koordinate und einem Winkel gespeichert werden. Die Position muss dabei für jeden Schuss gespeichert werden. Außerdem muss zu jedem A-Scan gespeichert werden, zu welchem Schuss er gehört.

Single Scan Bei einen Single-Scan muss lediglich einmalig die Position des Prüfkopfs gespeichert werden. Diese ist durch X-, Y- und Z-Koordinate sowie einem Winkel definiert. Außerdem muss ein einzelner A-Scan gespeichert werden.

5.1.1.1 Abbildung auf DICOM/DICONDE-Tags

In diesem Kapitel wird die Zuordnung der im Kapitel 2.3 beschriebenen Eigenschaften der verschiedenen Scan-Typen in das DICOM/DICONDE-Format beschrieben. Dabei wird zuerst ein erster Ansatz beschrieben, der aber zugunsten eines besseren Ansatzes verworfen wird.

Zunächst einmal muss unabhängig vom Scan-Typ eine Folge von A-Scans abgelegt werden. Jeder A-Scan wird dabei als Array von Short-Werten repräsentiert. Da der DICONDE-Standard zum Zeitpunkt dieser Arbeit kein Ablegen solcher Daten vorsieht, muss dafür eine spezialisierte Lösung entwickelt werden.

Im ersten Ansatz wurde dazu auf den DICOM-Standard zurückgegriffen. Im DICOM-Standard existiert ein Modul „Waveform“ [5, S. 1307–1311], das zum Ablegen von Zeit basierten Wellenformen vorgesehen und daher für den vorgesehenen Zweck geeignet ist.

Außerdem müssen zu jedem A-Scan, abhängig von dem verwendeten Scan-Verfahren, verschiedene Positionsdaten gespeichert werden. Diese Positionsdaten geben die Position des Prüfkopfes zum Zeitpunkt der Erstellung des A-Scans an und bestehen aus bis zu 3 Raumdimensionen und bis zu 2 Winkeln. Da das „Waveform“ Modul aber nicht speziell für diesen Anwendungsfall vorgesehen ist, existieren darin keine Elemente, in denen diese Daten abgelegt werden können. Allerdings wird in der Definition des „Waveform“ Modul innerhalb des Elementes „Channel Definition Sequence“ [5, S. 1312] auf das „Acquisition Context Module“ [5, S. 570–572] verwiesen. Dieses wiederum enthält das Sequenz-Element „Acquisition Context Sequence“ das im DICOM-Standard wie folgt beschrieben wird:

> A Sequence of Items that describes the conditions present during the acquisition of the data of the SOP Instance.

Daher ist „Acquisition Context Sequence“ laut Definition passend für den angeführten Verwendungszweck. Die genaue Zuordnung stellt sich dabei wie folgt dar.

Für jeden A-Scan wird eine weitere Sequenz in der „Acquisition Context Sequence“ angelegt. Diese Sequenz beinhaltet für jede zu speichernde Positionsangabe ein Item. Innerhalb des Items wird das Element „Referenced Frame Number“ auf die Nummer des A-Scans gesetzt, auf den sich das Item bezieht. Außerdem wird das Sequenz-Element „Measurement Units Code Sequence“ hinzugefügt. Dieses enthält ein einzelnes, wie in der Tabelle 5.1 dargestellt, strukturiertes Item. Die „Measurement Units Code Sequence“ hat den Zweck zu definieren, welche Einheiten die hinterlegten Werte besitzen.

Element-Name	Zuordnung
Code Value	Die Abkürzung der Einheit des zu speichernden Wertes nach dem Unified Code for Units of Measure [36].
Coding Scheme Designator	Der feste Eintrag UCUM, um zu definieren, dass alle angegebenen Werte nach [36] zu interpretieren sind.
Coding Scheme Name	Der Name des in „Coding Scheme Designator“ definierten Kodierungsschemas. In diesem Fall entspricht das dem Wert „Unified Code for Units of Measure“.
Code Meaning	Eine menschenlesbare Erklärung des in „Code Value“ angegebenen Codes.

Tabelle 5.1: Aufbau der MeasurementUnitsCodeSequence.

Des Weiteren in das Element „Value Type“ der feste Wert „NUMERIC“ eingetragen, um zu definieren, dass es sich bei dem Wert um eine Zahl handelt.

Außerdem wird eine „Concept Code Sequence“ mit einem Item hinterlegt. Diese definiert, welche Raumrichtung oder welchen Winkel der hinterlegte Wert darstellt.

Da es sich bei den zu speichernden Werten ausschließlich um Gleitkommazahlen handelt, werden die eigentlichen Werte unter dem Element „Floating Point Value“ abgelegt.

Da dieser Ansatz allerdings zum größten Teil auf DICOM statt DICONDE basiert, wird im Rahmen dieser Arbeit die Entscheidung getroffen, eine neue IOD für das Ablegen von Ultraschall-Daten zu definieren. Diese IOD soll dann zunächst als institutsinterner Standard verwendet werden und nach dem dieser sich als praxistauglich erwiesen hat, bei der ASTM zu Aufnahme in den DICONDE-Standard eingereicht werden.

Im Folgenden wird dieser neue Ansatz beschrieben.

Der neue Ansatz gliedert sich dabei in vier wesentliche Bestandteile auf. Zum einen ein Feld „Scan Type“ das festlegt, mit welchem Scan-Typ die Daten produziert wurden.

Scan-Type Feld Das Scan-Type Feld legt fest, mit welchem Scan-Type die abgelegten Daten produziert wurden.

Wave Source Dimensions Sequence Der Zweck der „Wave Source Dimensions Sequence“ besteht dabei darin zu definieren, welche Raumdimensionen für jeden A-Scan abgespeichert werden müssen. Dies wird erreicht, indem der Sequenz für jede benötigte Raumdimension ein Item hinzugefügt wird. In diesem Item wird dann zum einen eine eindeutige Nummer im Feld „Dimension Number“ hinterlegt, um den Eintrag referenzieren zu können. Zum anderen wird mit Hilfe der Felder „Dimension Name“ „Dimension Code Value“ „Dimension Coding Scheme Designator“ „Dimension Coding Scheme Version“ „Dimension Code Meaning“ „Dimension Coding Scheme Name“ und „Dimension Coding Scheme Responible Organisation“ definiert, in welcher Einheit die Werte zu dieser Dimension abgelegt werden müssen. Eine detaillierte Beschreibung der genannten Felder ist in der, an diese Arbeit angehängten, Erweiterung der US-Image IOD [31] zu finden. Außerdem beinhaltet jedes Item noch ein Feld „Value Type“ das den Datentyp der abgelegten Werte definiert.

Waveform Sequence Die „Waveform Sequence“ ist größtenteils aus [5, S. 1307–1311] übernommen. Allerdings wurde die im nächsten Eintrag beschriebene „Wave Source Values Sequence“ hinzugefügt, um Raumdimensionen zu den einzelnen A-Scans speichern zu können.

Wave Source Values Sequence Die „Wave Source Values Sequence“ beinhaltet die Werte für die in der „Wave Source Dimensions Sequence“ definierten Dimensionen jeweils für einen A-Scan. Die Sequenz beinhaltet dabei für jede Dimension ein Item. In jedem dieser Items wird dann mithilfe des Feldes „Referenced Dimension“ auf das Feld „Dimension Number“ der „Wave Source Dimensions Sequence“ referenziert, um zu definieren, auf welche Dimension sich der Wert bezieht. Der eigentliche Wert wird ‚abhängig vom „Value Type“ Feld der „Wave Source Dimensions Sequence“ entweder in „Numeric Value“ „Rational Number Value“ oder „Floating Point Value“ abgelegt.

Die größte Vorteile des neuen Ansatzes liegt dabei darin, dass die Beschreibung der Dimensionen nur noch einmal und nicht mehr für jeden A-Scan separat abgespeichert werden muss. Dies hat eine deutliche Reduzierung der Größe der erzeugten DICOM-Dateien zur Folge. Des Weiteren können die DICOM-Dateien so schneller geschrieben und gelesen werden. Außerdem können mit diesem Ansatz auch weitere Scan-Typen verarbeitet werden, da nur entsprechend andere Dimensionen hinterlegt werden müssen.

Auch können Variationen der bereits implementierten Scan-Typen verwendet werden. So wird bei bestimmten Sektor-Scan-Verfahren ein weiterer Winkel benötigt, der mithilfe dieses Ansatzes einfach hinzugefügt werden kann.

Eine tabellarische Aufführung des Ansatzes bzw. die Erweiterung der besehenden US-Image IOD [31] ist im Anhang A dieser Arbeit zu finden. Die dort zu findende Definition wurde mit dem Zweck der Einbringung in den DICONDE Standard im Verlauf dieser Arbeit an das entsprechende Gremium der Deutsche Gesellschaft für Zerstörungsfreie Prüfung (DGZfP) eingereicht.

5.1.2 Rohdaten aus DICONDE laden

Das Laden der Daten wurde so implementiert, dass es möglich ist, die Positionsdaten und die eigentlichen Messdaten separat zu laden. Das Ziel dieser Implementierung ist, dass schnellere Auslesen, falls nur die Positionsdaten benötigt werden. Nachdem die Positionsdaten ausgelesen wurden, können die Rohdaten für jede Position einzeln oder für mehrere Positionen auf einmal nachgeladen werden.

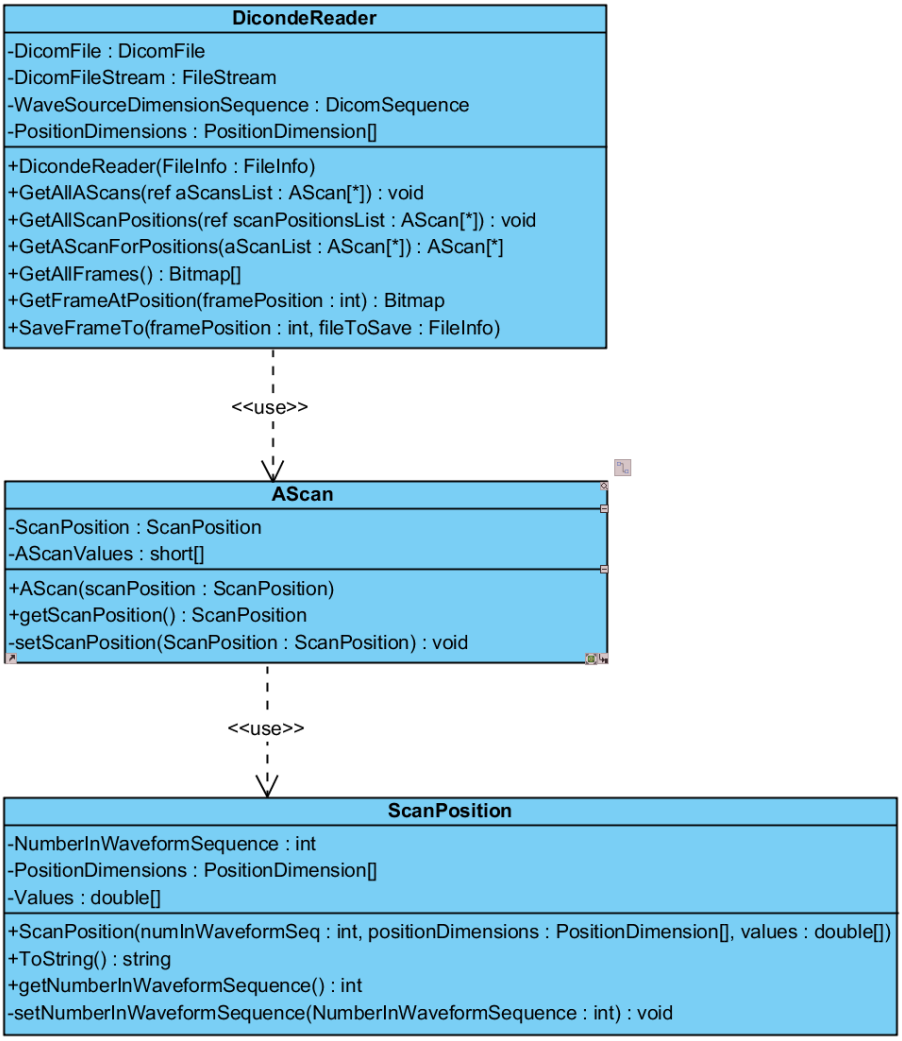

Abbildung 5.1: Klassendiagramm zum Auslesen einer DICONDE-Datei

In der Abbildung 5.1 ist die Funktionalität zum Laden einer DICONDE-Datei dargestellt. Die für das Laden der Rohdaten relevanten Funktionalitäten werden im Folgenden

beschrieben. Außerdem wird das Laden von Bildern aus DICONDE-Dateien im Kapitel 5.1.3 beschrieben.

Die abgebildete Klasse „ScanPosition“ beschreibt die Position eines einzelnen A-Scans. Das Attribut „NumberInWaveformSequence“ dient dabei der Zuordnung der Scan-Position zu dem entsprechenden Item innerhalb der „WaveformSequence“. Da diese Position je nach Scan-Verfahren durch verschieden viele Dimensionen definiert ist, besitzt die Klasse das Attribut „PositionDimensions“Ḋabei handelt es sich um ein Array von Objekten der Klasse „PositionDimension“. Diese Klasse, zu sehen in Abbildung 5.2, wiederum beinhaltet die Beschreibung der jeweiligen Dimension. Dabei wird jede Position durch ihre Einheit, ihren Typ und ihren Namen definiert. Außerdem hat die Klasse das Attribut „Values“ vom Typ Double-Array, dass die Werte der entsprechenden Dimensionen hält.

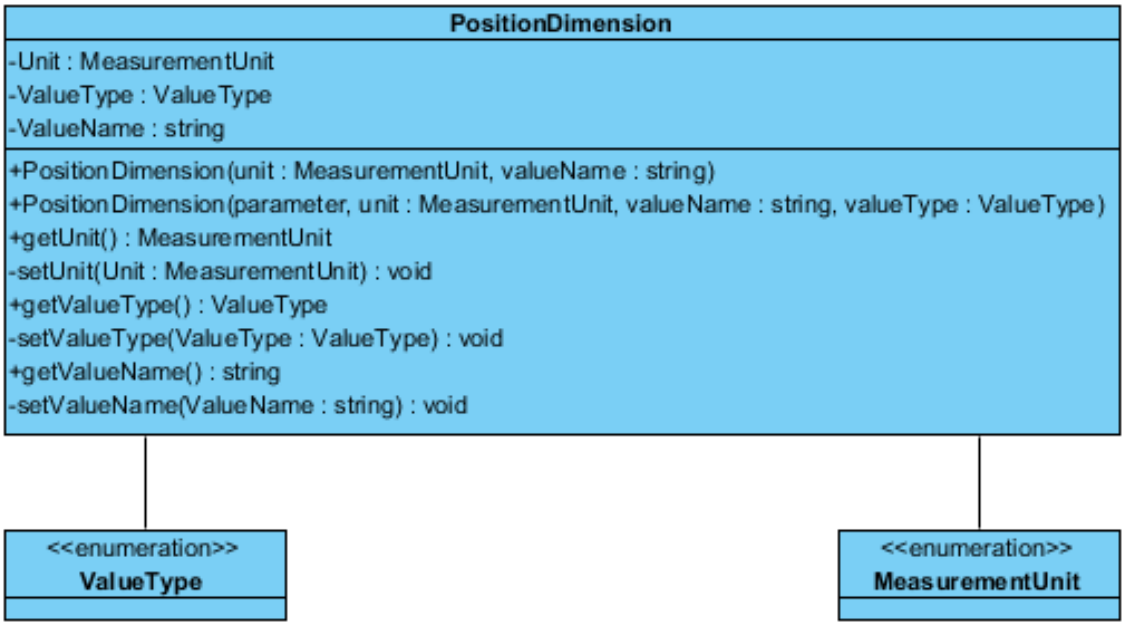

Abbildung 5.2: Klassendiagramm zur Klasse „PositionDimension“.

Die Klasse „A-Scan“ beinhaltet sowohl die Werte eines A-Scan, dargestellt als Array von Short-Werten als auch ein Objekt der Klasse „ScanPosition“ das die zum A-Scan gehörenden Positionsdaten enthält.

Die Klasse „DicondeReader“ stellt die öffentlichen Methoden zum Auslesen der DICONDE-Dateien bereit. Dabei ist zu beachten, dass die privaten Methoden im Klassendiagramm nicht dargestellt sind, um die Übersichtlichkeit zu wahren.

Im Folgenden werden die für das Laden der A-Scans relevanten öffentlichen Methoden beschrieben.

Konstruktor Dem Konstruktor muss ein Parameter vom Typ „FileInfo“ übergeben werden, der festlegt, welche DICONDE-Datei ausgelesen werden soll.

GetAllAScans Dieser Methode kann eine Liste von A-Scans als Referenz übergeben werden. Die Liste wird dabei mit den in der DICONDE Datei vorhandenen Werten gefüllt. Dabei werden sowohl die Rohdaten als auch die Positionsdaten ausgelesen.

GetAllAscanPositions Diese Methode funktioniert wie „GetAllAScans“ mit dem Unterschied, dass nur Positionsdaten ausgelesen werden.

GetAScanForPositions Dieser Methode wird eine Liste von A-Scans übergeben, die bereits Positionsdaten enthält. Als Rückgabewert der Methode wird eine Liste von A-Scans zurückgegeben, die sowohl die übergebenen Positionsdaten als auch die zu den Positionen passenden Rohdaten enthält.

Außerdem sind an dieser Stelle noch folgende relevanten privaten Methoden der Klasse zu nennen.

DetermineUnit Ermittelt für jede Dimension die entsprechende Einheit.

DetermineValueName Ermittelt für jede Dimension den entsprechenden Namen.

DetermineValueType Ermittelt, von welchen Typ die Werte der auszulesenden Dimension sind.

ReadDataFromFile Ermittelt die eigentlichen Rohdaten.

5.1.2.1 Performance-Probleme

Bei der Implementierung der beschriebenen Funktionalität wird folgendes Problem deutlich. Durch das Auslesen der Positionsdaten gegenüber dem Auslesen der kompletten Daten kann nur ein geringer Performance-Vorteil erzielt werden.

Nach eingehender Beschäftigung mit möglichen Ursachen wird folgende Ursache als wahrscheinlich angenommen. Fo-Dicom verwendet zum Auslesen von DICOM-Dateien die Klasse „DicomFile“. [40] Der Methode „Open“ kann zwar ein Enum-Parameter übergeben werden, der laut Dokumentation dazu führen soll, dass große Tags nicht eingelesen werden. Nach Begutachtung des Fo-Dicom Codes stellt sich jedoch heraus, dass dabei nicht die tatsächliche Größe der Tags berücksichtigt wird, sondern dass festgelegte Tags übersprungen werden. Dies führt wiederum dazu, dass die Dicom-Datei vollständig in den Arbeitsspeicher geladen werden muss, bevor die einzelnen Tags ausgelesen werden können.

5.1.3 Bilder aus DICONDE laden

Wie in Abbildung 5.1 zu sehen ist, sind dort neben den Methoden zum Laden der Rohdaten auch solche zum Laden von Bildern vorhanden. Diese werden im Folgenden kurz aufgezählt und erläutert.
Die Methoden haben dabei alle gemeinsam, das sie die Bilder aus der im Konstruktor der Klasse übergebenen DICOM bzw. DICONDE-Datei auslesen.

GetAllFrames Diese Methode liefert alle in der Datei vorhandenen Bilder aus und liefert sie als Array von Bitmaps zurück.

GetFrameAtPosition Diese Methode liefert das Bild mit der übergebenen Position als Bitmap zurück.

SaveFrameTo Diese Methode speicher das Bild mit der übergebenen Position unter dem übergebenen Dateinamen.

5.1.4 Bilder konvertieren

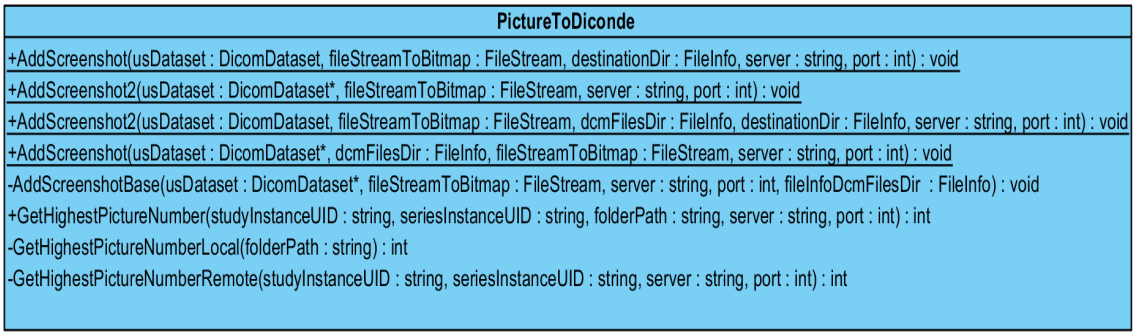

Abbildung 5.3: Klassendiagramm DICONDE-Libary Picture To DICONDE

In der Abbildung 5.12 ist ein Klassendiagramm der Klasse zum Erstellen von DICONDE-Dateien aus Bild-Dateien zu sehen. Es handelt sich dabei um eine Umsetzung der in Kapitel 4.0.2 geplanten Funktionalität.
Wie bereits in Kapitel 4.0.2 geplant gibt es die öffentliche überladene Methode „AddPicture“ in vier Varianten. Dabei werden die folgenden Parameter folgenden Parameter verwendet.

usDataset Bei Parameter vom Typ „DicomDataset“ handelt es sich um das Dataset, dem das Bild hinzugefügt werden soll. Je nachdem welche Version der Methode aufgerufen wird, wird es als Referenz übergeben.

fileStreamToBitmap Dieser Parameter vom Typ „FileStream“ hält einen Stream auf das Bild, das gespeichert werden soll.

dcmFilesDir Dieser Parameter vom Typ „FileInfo“ beinhaltet die Information, wo bereits erstellte Dateien abgelegt wurden.

destinationDir Dieser Parameter vom Typ „FileInfo“ beinhaltet die Information, in welchem Verzeichnis die resultierende Datei gespeichert werden soll.

server Dieser Parameter vom Typ „string“ beinhaltet die Netzwerkadresse des PACS-Servers, mit dem die Nummerierung der Bilder abgeglichen werden soll.

port Dieser Parameter vom Typ „int“ beinhaltet den Port des PACS-Servers, mit dem die Nummerierung der Bilder abgeglichen werden soll.

Die vier Varianten der Methode unterscheiden sich dabei wie folgt.

In der ersten Version wird der Parameter „destinationDir“ übergeben, der die Information enthält, in welchem Verzeichnis die resultierende Datei gespeichert werden soll. Der zweiten Variante hingegen wird dieser Parameter nicht übergeben, sondern der Parameter „usDataset“ wird als Referenz übergeben sodass keine Datei erzeugt wird, sondern das Dataset im aufrufenden Code weiter verwendet werden kann. Bei diesen beiden Varianten wird der im weiteren Verlauf dieses Kapitels erklärte ein Abgleich der Bildnummern in dem Verzeichnis vorgenommen, in dem das ursprüngliche Bild gespeichert ist.

Bei der dritten und vierten Variante dieser Methode handelt es sich jeweils um eine Variation der ersten bzw. der zweiten Variante der Methode, bei der noch der Parameter „dcmFilesDir“ übergeben wird. Bei diesen Varianten wird der Abgleich der Bildnummern in dem mittels „dcmFilesDir“ übergebenen Verzeichnis vorgenommen.

Bei der privaten Methode „AddScreenshotBase“ handelt es sich um eine Hilfsmethode, in der die eigentliche Konvertierung der Bilddaten implementiert ist und die innerhalb der vier zuvor beschriebenen Methoden aufgerufen wird. Außerdem gibt es die öffentliche Methode „GetHighestPictureNumber“ die auch innerhalb der Version der „AddPicture“ Methode mit Abfrage des PACS-Servers verwendet wird, um die Bildnummer zu synchronisieren.

Die Methode ermittelt die höchste vorhandene Bildnummer zum einen in dem in „folderPath“ übergebenen Ordner, zum andern auf dem mittels „server“ und „port“ definierten PACS-Server und gibt die insgesamt höchste Nummer zurück. Dabei werden nur DICONDE-Dateien berücksichtigt, die sowohl die übergebene „studyInstanceUID“ als auch die übergebene „seriesInstanceUID“ beinhalten. Die Abfrage wird dabei mittels des in Kapitel 2.1.5 beschriebenen CFind-Services realisiert. Um die höchste lokale Nummer zu ermitteln, greift die Methode dabei auf die ebenfalls im Klassendiagramm dargestellte private Methode „GetHighestPictureNumberLocal“ zurück. Gleiches gilt für die höchste Bildnummer auf dem PACS-Server und die Methode „GetHighestPictureNumberRemote“.

5.1.5 Upload

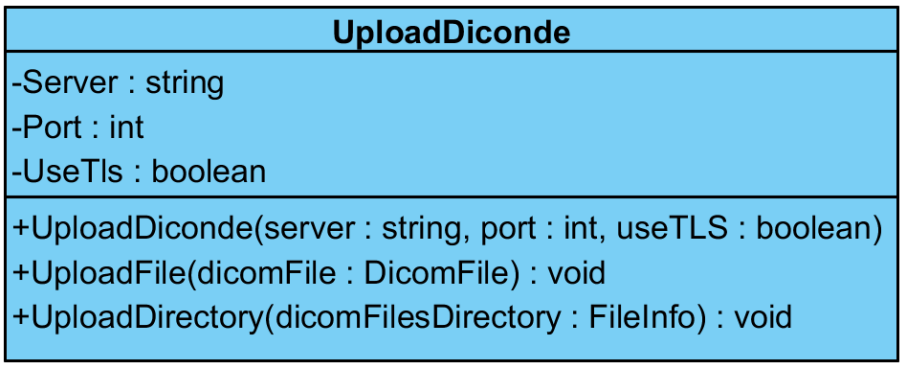

Abbildung 5.4: Klassendiagramm der Klasse zum Hochladen von Daten von einem PACS-Server.

Das Hochladen von Daten wird über die in Abbildung 5.4 dargestellte Klasse realisiert.

Beim Aufruf des Konstruktor müssen die folgenden Parameter mitgegeben werden.

server Dieser Parameter beinhaltet die Netzwerkadresse des PACS-Servers.

port Der Parameter Port beinhaltet den entsprechenden Port der des PACS-Servers.

useTLS Dieser Parameter gibt an, ob bei der Kommunikation mit dem PACS-Server Transport Layer Security (TLS) genutzt werden soll.

Die beiden Methoden „UploadFile“ und „UploadDirectory“ dienen dem Hochladen von DICOM/DICONDE Dateien. Der Unterschied besteht darin, dass die Methode „UploadFile“ nur die eine übergebene Datei hochlädt und die Methode „UploadDirectory“ alle passenden Dateien im übergebenen Verzeichnis. Um den weiteren Programmablauf beim Aufrufen der Methoden nicht zu blockieren, handelt es sich dabei um asynchrone Methoden.

5.1.6 Download

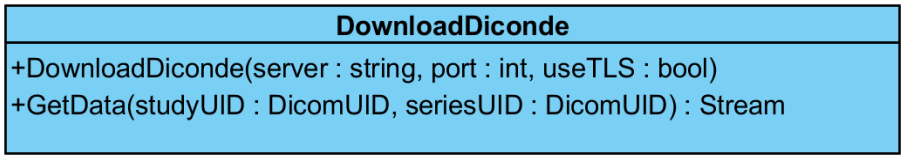

Abbildung 5.5: Klassendiagramm der Klasse zum Herunterladen von Daten von einem PACS-Server.

Bei der in Abbildung 5.5 dargestellten Klasse handelt es sich um eine Klasse zum Laden von Daten von einem PACS-Server. Dem Konstruktor der Klasse müssen die im Folgenden aufgezählten und erläuterten Parameter übergeben werden.

server Dieser Parameter beinhaltet die Netzwerkadresse des PACS-Servers.

port Der Parameter WADO-Port beinhaltet den entsprechenden Port der WADO-Schnittstelle des PACS-Servers.

useTLS Dieser Parameter gibt an, ob bei der Kommunikation mit dem PACS-Server TLS genutzt werden soll.

Die Methode zum eigentlichen Abrufen der Daten „GetData“ vom PACS-Server besitzt die beiden folgenden Parameter.

studyUID Dieser Parameter beinhaltet eine ID, die die Studie, aus der die Daten abgerufen werden sollen, eindeutig identifiziert.

seriesUID Dieser Parameter beinhaltet eine ID, die die Serie, aus der die Daten abgerufen werden sollen, eindeutig identifiziert.

Das Abrufen der Daten wird dabei mit im Abschnitt 2.1.6 erwähnten WADO-RS Schnittstelle umgesetzt.

5.2 Integration in existierende Softwarelösungen

In diesem Kapitel wird die Integration der in der DICONDE-Bibliothek implementierten Funktionalitäten in andere Softwarelösungen dargestellt.

Implementierung DICONDE Bilder In der USP wird im Rahmen dieser Arbeit eine Funktion implementiert, die dazu dient, Bilder von verschiedenen Ultraschallauswertungen anzufertigen. Dabei sollen diese nach [31] und [33] in einer DICONDE-Datei gespeichert werden, welche dann auf einen PACS Server hochgeladen wird.

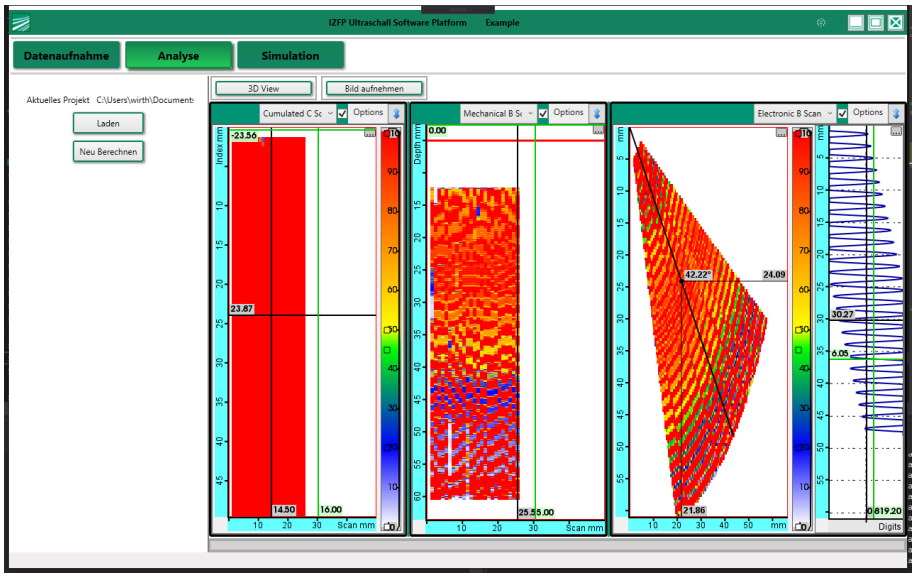

Abbildung 5.6: In dieser Ansicht kann der Werkstoffprüfer bereits angefertigte Messungen analysieren. Es handelt sich dabei um eine Simulation einer Messfahrt eines Sektor-Scans.

In der in Abbildung 5.6 dargestellten Ansicht kann der Werkstoffprüfer bereits angefertigte Messungen analysieren. Dabei hat er die Wahl zwischen verschiedenen Darstellungen der Messergebnisse. Die im Rahmen dieser Arbeit hinzugefügte Funktion ermöglicht es nur, Bilder von diesen Ansichten anzufertigen und diese im DICONDE-Format abzulegen. Neben den Bildern der jeweiligen Ansichten werden dabei noch zusätzliche, in Verbindung mit der Messung stehende Daten in der DICONDE-Datei abgelegt. Dazu zählen unter anderem Informationen zu den verwendeten Ultraschallgeräten, dem Werkstoffprüfer und dem geprüften Objekt. Außerdem werden Informationen hinterlegt, aus denen hervorgeht, zu welcher Untersuchung und zu welcher Messung die Daten zugeordnet sind.

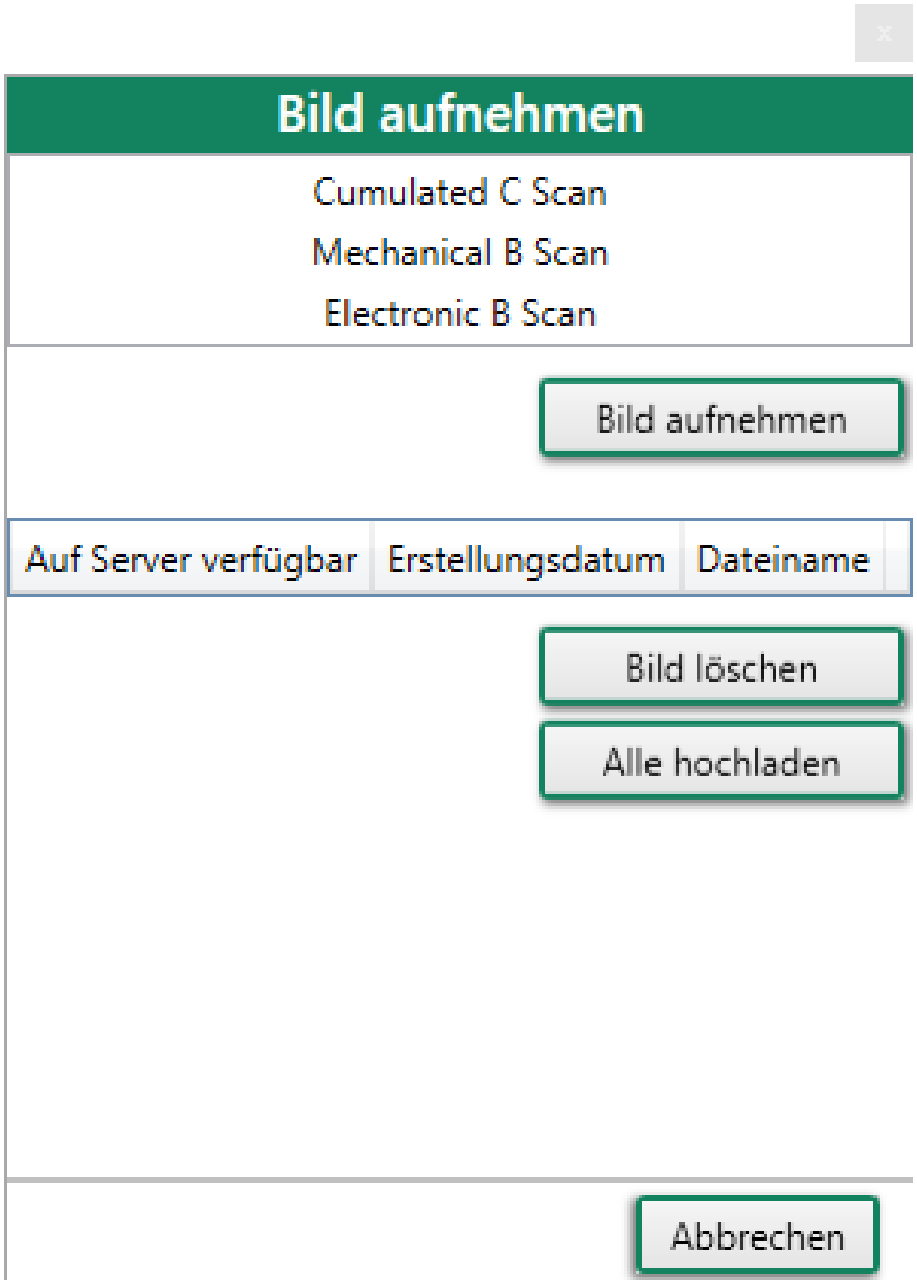

Abbildung 5.7: In diesem Dialog können Bilder, aller zum aktuellen Zeitpunkt dargestellten Ansichten, erstellt werden.

In Abbildung 5.7 ist der Dialog zum Erstellen der Bilder zu sehen. Dabei wird dem Benutzer die Möglichkeit geboten, Bilder von allen aktuell dargestellten Ansichten zu erstellen.

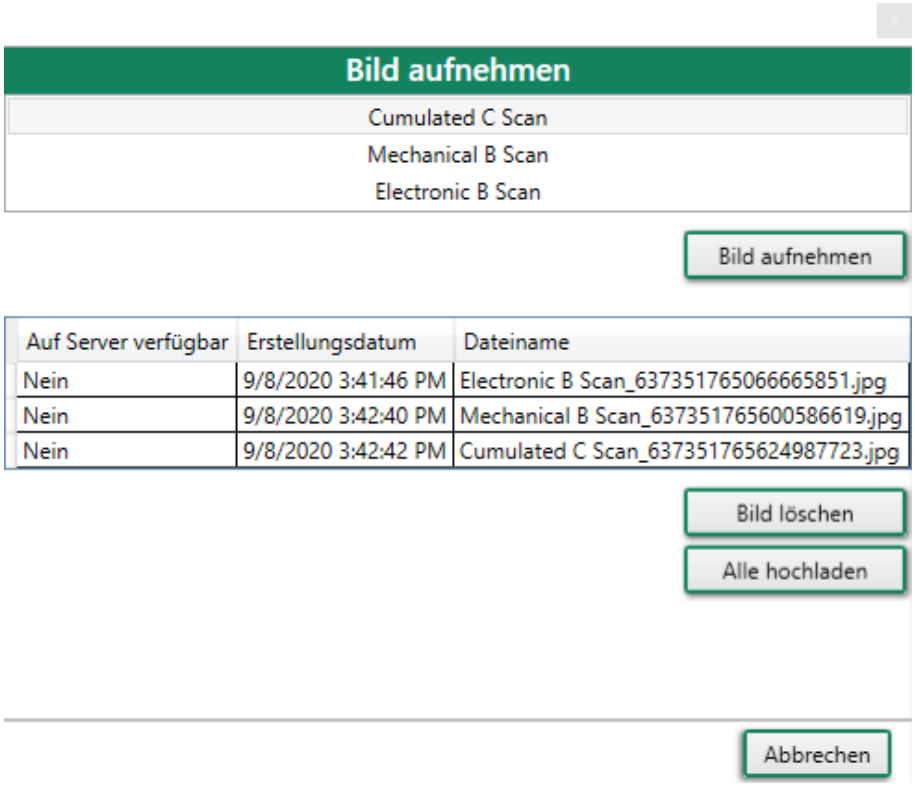

Abbildung 5.8: Bereits angefertigte Bilder werden aufgelistet und es wird angezeigt, ob diese bereits hochgeladen wurden.

In Abbildung 5.8 ist derselbe Dialog zu sehen, nachdem mehrere Bilder angefertigt wurden. Die bereits angefertigten Bilder werden dabei tabellarisch aufgelistet. Zu beachten ist, dass die Auflistung der Bilder mithilfe der „FileSystemWatcher“ Klasse [20] aktualisiert wird. So ist sichergestellt, dass nur Dateien, die wirklich im Dateisystem vorhanden sind, angezeigt werden. Außerdem wird in der Spalte „Auf Server verfügbar“ angezeigt, ob die entsprechende Datei bereits auf den PACS-Server hochgeladen wurden, was in der Abbildung 5.8 noch nicht der Fall ist.

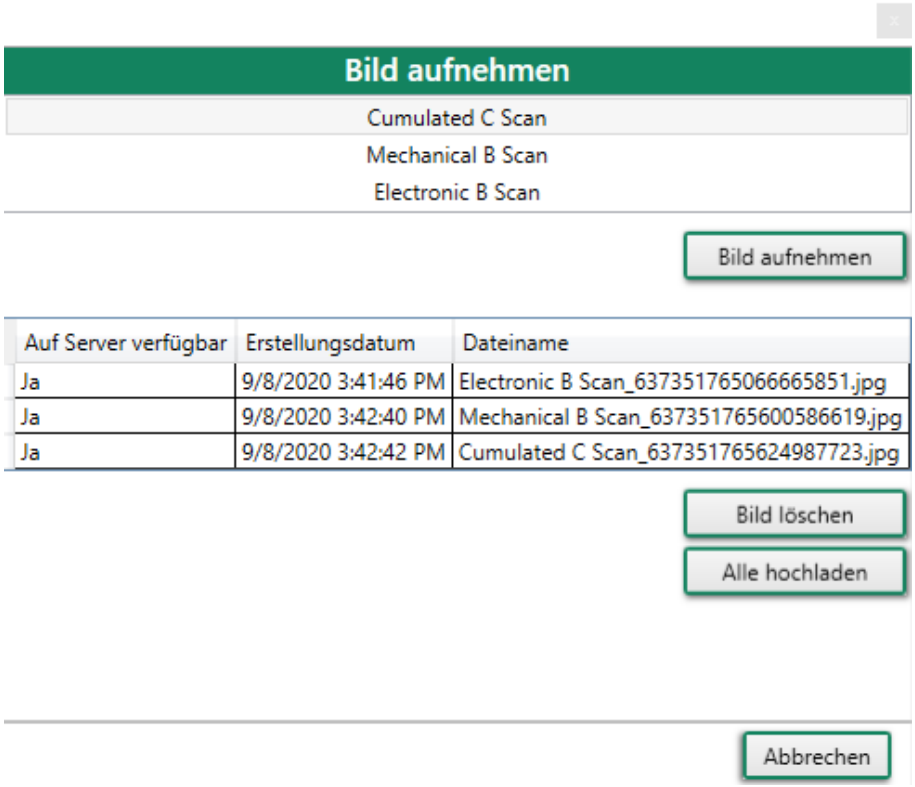

Abbildung 5.9: Nachdem die Dateien hochgeladen wurden, wird dies in der Spalte „Auf Server verfügbar“ angezeigt.

Nachdem die Daten nach einem Klick auf „Alle hochladen“ auf den PACS-Server hochgeladen wurden, wird dies wie in Abbildung 5.9 dargestellt angezeigt.

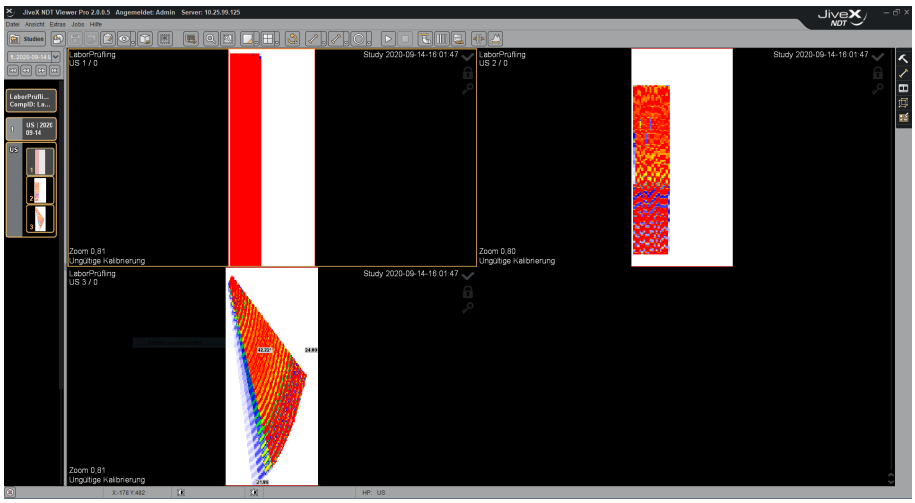

Abbildung 5.10: Darstellung der aus der USP hochgeladenen Studie.

Die entsprechende Darstellung der hochgeladenen Studie ist in Abbildung 5.10 zu sehen. Dabei kann die genaue Darstellung je nach verwendetem Anzeigeprogramm leicht abweichen. Bei der in 5.10 verwendeten Software handelt es sich um den „JiveX NDT viewer Pro“ der Firma „DIMATE“.

DICONDE Dump

Suche:

Schlüssel	Name	Länge	Typ	Anzahl	Wert
DICONDE Object					
(0002,0000)	File Meta Information Gr...	4	UL	1	[188]
(0002,0001)	File Meta Information Ve...	2	OB	1	[{byte array #=2}]
(0002,0002)	Media Storage SOP Cla...	28	UI	1	[1.2.840.10008.5.1.4.1.1.6.1]
(0002,0003)	Media Storage SOP Inst...	44	UI	1	[2.25.325412521221025120755931432875610454 87]
(0002,0010)	Transfer Syntax UID	20	UI	1	[1.2.840.10008.1.2.1]
(0002,0012)	Implementation Class ...	30	UI	1	[1.3.6.1.4.1.52349.20100301.5.0]
(0002,0013)	Implementation Version...	12	SH	1	[JIVEX_TK_5.0]
(0008,0005)	Specific Character Set	10	CS	1	[ISO_IR 192]
(0008,0008)	Image Type	38	CS	4	[DERIVED, SECONDARY, B_SCAN, LONGITUDINAL]
(0008,0016)	SOP Class UID	28	UI	1	[1.2.840.10008.5.1.4.1.1.6.1]
(0008,0018)	SOP Instance UID	44	UI	1	[2.25.325412521221025120755931432875610454 87]
(0008,0020)	Study Date	8	DA	1	[20200914]
(0008,0030)	Study Time	6	TM	1	[160147]
(0008,0050)	Accession Number	0	SH	0	[]
(0008,0060)	Modality	2	CS	1	[US]
(0008,0070)	Manufacturer	16	LO	1	[Fraunhofer IZFP]
(0008,0081)	Institution Address	24	ST	1	[info@izfp.fraunhofer.de]
(0008,0090)	Referring Physician's N...	0	PN	0	[]
(0008,1090)	Manufacturer's Model N...	4	LO	1	[File]
(0008,9123)	Creator-Version UID	18	UI	1	[49.66123.0.0.1.100]
(0010,0010)	Component Name	14	PN	1	[LaborPrüfling]
(0010,0020)	Component ID	14	LO	1	[LaborPrüfling]
(0010,0030)	Component Manufacturi...	0	DA	0	[]
(0010,0040)	Patient's Sex	2	CS	1	[O]
(0014,4010)	Transmit Transducer S...	-1	SQ	1	[[sequence: 1 items]]
Item 0					
(0008,0005)	Specific Character Set	10	CS	1	[ISO_IR 192]
(0008,0070)	Manufacturer	8	LO	1	[OLYMPUS]
(0008,1090)	Manufacturer's Model N...	8	LO	1	[5L16-A1]
(0014,4012)	Number of Elements	2	US	1	[16]
(0014,4013)	Element Shape	6	CS	1	[ROUND]
(0014,401A)	Nominal Frequency	8	DS	1	[5000000]
(0014,5100)	Active Aperture	2	US	1	[16]
(0018,1000)	Device Serial Number	8	LO	1	[unknown]
(0018,6031)	Transducer Type	12	CS	1	[LINEAR ARRAY]
(0014,4011)	Receive Transducer Se...	-1	SQ	1	[[sequence: 1 items]]
Item 0					
(0008,0005)	Specific Character Set	10	CS	1	[ISO_IR 192]
(0008,0070)	Manufacturer	8	LO	1	[OLYMPUS]
(0008,1090)	Manufacturer's Model N...	8	LO	1	[5L16-A1]
(0014,4012)	Number of Elements	2	US	1	[16]
(0014,4013)	Element Shape	6	CS	1	[ROUND]
(0014,401A)	Nominal Frequency	8	DS	1	[5000000]
(0014,5100)	Active Aperture	2	US	1	[16]
(0018,1000)	Device Serial Number	8	LO	1	[unknown]
(0018,6031)	Transducer Type	12	CS	1	[LINEAR ARRAY]
(0018,1020)	Software Version(s)	22	LO	2	[DICONDE15, 1.0.0.3d7e3c]

Kopieren Schließen

Abbildung 5.11: Ausschnittsweise Darstellung der zur Messung gehörigen Daten.

Außerdem können die bereits erwähnten, zu der Messung gehörige Daten, als Auflistung dargestellt werden. Dies ist ausschnittsweise in Abbildung 5.10 zu sehen.

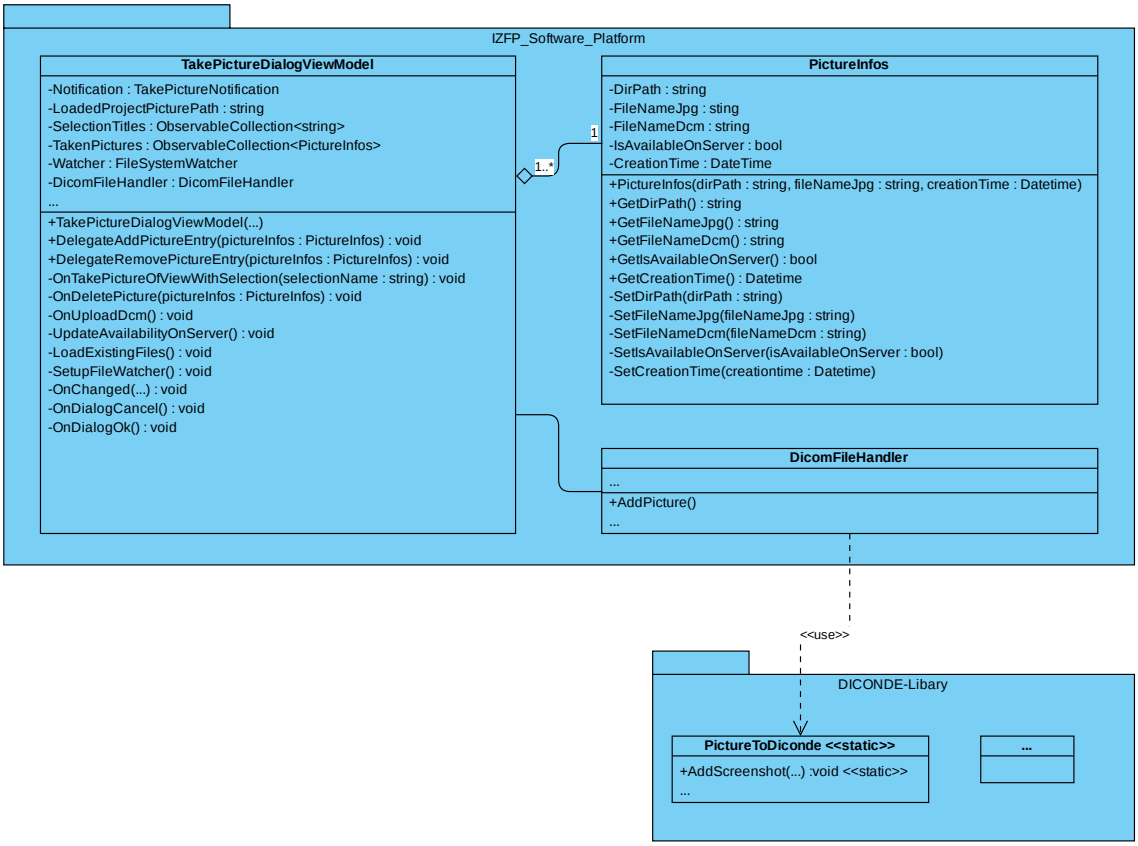

Abbildung 5.12: Klassendiagramm der Bild-Aufnahmefunktion

In der Abbildung 5.12 ist das passende Klassendiagramm zu der in diesem Kapitel beschriebenen Funktion abgebildet. Die Software ist dabei nach dem im Kapitel 2.6 beschrieben MVVM Entwurfsmuster aufgebaut.

Die Implementierung der USP regelt dabei das Ablegen und Verwalten der Daten auf dem lokalen System sowie das Anfertigen der Bilder. Um die Bilder in das DICONDE-Format zu bringen, wird jedoch auf die in Kapitel 5.1.4 beschriebene Funktionalität der DICONDE-Bibliothek zurückgegriffen.

5.3 DICONDE-Tag Generierung

Da die in [33], [31], [29], [32], [30], und [34] definierten DICONDE-Tags nicht in maschinenlesbarer Form vorliegen, sondern nur in Tabellenform in den jeweiligen Dokumenten vorhanden sind, können sie nicht ohne Weiteres in Quellcode überführt werden. Aus diesem Grund wurde mithilfe der Programmiersprache Python und des Frameworks Tabula-Py 2.5.7 eine Software erstellt, die dazu dient, die DICONDE-Tags zu extrahieren und daraus den entsprechenden C# Quellcode zu generieren.

Dazu extrahiert die Python-Software zuerst mithilfe von Tabula-Py die in den PDF-Dokumenten gefundenen Tabellen. Im ersten Ansatz wird anschließend aus den Daten mithilfe von zusammengesetzten Strings und regulären Ausdrücken C# Quellcode generiert.

TABLE 2 Component Module

Attribute Name	Tag	VR	VM	Type	Description
Component					
Component Name	(0010,0010)	PN	1	2	Component Name or Part name
Component ID Number	(0010,0020)	LO	1-N	2	Component ID or Part ID
Other Component IDs	(0010,1000)	LO	1-N	3	Additional Component IDs when multiple parts in one image
Other Component Names	(0010,1001)	PN	1	3	Additional Component IDs when multiple parts in one image
Component Manufacturing Date	(0010,0030)	DA	1	2	
Patient Sex	(0010,0040)	CS	1	2	Required for DICOM compliance. Should either contain zero value or the enumerated value of "O" for OTHER.
Component Notes	(0010,4000)	LT	1	3	
Component Manufacturing Procedure	(0014,0025)	ST	1	3	
Component Manufacturer	(0014,0028)	ST	1	3	
Component Welder IDs	(0014,0100)	LO	1-N	3	A text string identifying the individual or machine performing welding operations on the component.
Material					
Material Name	(0010,2160)	SH	1	2	Steel, copper, etc. (16 char. max.)
Material Grade	(0014,0042)	ST	1	3	
Material Properties Description	(0014,0044)	ST	1	3	
Material Notes	(0014,0046)	LT	1	3	
Geometry					
Material Thickness	(0014,0030)	DS	1-N	3	Wall/material thickness in mm
Material Pipe Diameter	(0014,0032)	DS	1-N	3	Retired
Material Isolation Diameter	(0014,0034)	DS	1-N	3	Retired
Component Shape	(0014,0050)	CS	1	3	General description of the shape of the test piece. See 7.4.1.1for additional information.
Curvature Type	(0014,0052)	CS	1	3	Type of curvature present in the test piece. See 7.4.1.2for additional information.
Outer Diameter	(0014,0054)	DS	1	3	Outer diameter of curved test specimen in mm
Inner Diameter	(0014,0056)	DS	1	3	Inner diameter of curved test specimen in mm

Abbildung 5.13: ASTM E2339_15 Tabelle 2 Component Module [33, S. 5]

So ergibt sich beispielsweise aus der in 5.13 in zu sehenden Tabelle, die 5.1 zu sehende Ausgabe. Diese kann dann in einem entsprechenden C#-Programm verwendet werden.

Listing 5.1: Ausgabe Tag-Generierung für Component Module

```
//(0x0010,0x0010) VR=PN VM=1
var ComponentName = new DicomTag(0x0010,0x0010);
//(0x0010,0x0020) VR=LO VM=1
var ComponentIDNumber = new DicomTag(0x0010,0x0020);
//(0x0010,0x1000) VR=LO VM=1
var OtherComponentIDs = new DicomTag(0x0010,0x1000);
//(0x0010,0x1001) VR=PN VM=1
var OtherComponentNames = new DicomTag(0x0010,0x1001);
//(0x0010,0x0030) VR=DA VM=1
var ComponentManufacturingDate = new DicomTag(0x0010,0x0030);
//(0x0010,0x0040) VR=CS VM=1
var PatientSex = new DicomTag(0x0010,0x0040);
//(0x0010,0x4000) VR=LT VM=1
var ComponentNotes = new DicomTag(0x0010,0x4000);
//(0x0014,0x0025) VR=ST VM=1
var ComponentManufacturingProcedure = new DicomTag(0x0014,0x0025);
//(0x0014,0x0028) VR=ST VM=1
var ComponentManufacturer = new DicomTag(0x0014,0x0028);
//(0x0014,0x0100) VR=LO VM=1
var ComponentWelderIDs = new DicomTag(0x0014,0x0100);
```

Nach firmeninterner Kommunikation stellt sich jedoch heraus, dass auch in anderen Projekten Interesse an den maschinenlesbaren DICONDE-Tags besteht. Daher wurde die Python-Software so erweitert, dass sie die Daten auch im JSON-Format ausgibt, um sie universell verwendbar zu machen. Eine beispielhafte Ausgabe ist in 5.2 zu sehen. In den folgenden Kapiteln wird außerdem beschrieben, wie verschiedene Codefragmente erzeugt werden.

Listing 5.2: Ausgabe Tag-Generierung JSON

```
        ...
        "ASTM E2339_15.pdf": {
                "ComponentName": {
                        "tagname": "ComponentName",
                        "group": "0010",
                        "element": "0010",
                        "vr": "PN",
                        "vm": "1"
                },
                "ComponentIDNumber": {
                        "tagname": "ComponentIDNumber",
                        "group": "0010",
                        "element": "0020",
                        "vr": "LO",
                        "vm": "1"
                },
                "OtherComponentIDs": {
                        "tagname": "OtherComponentIDs",
                        "group": "0010",
                        "element": "1000",
                        "vr": "LO",
                        "vm": "1-n"
```

```
        },
        "OtherComponentNames": {
                "tagname": "OtherComponentNames",
                "group": "0010",
                "element": "1001",
                "vr": "PN",
                "vm": "1"
        },
        ...
        "ASTM E2663_14-US.pdf": {
                "PulserEquipmentSequence": {
                        "tagname":
                           ↪ "PulserEquipmentSequence",
                        "group": "0014",
                        "element": "4002",
                        "vr": "SQ",
                        "vm": "1"
                },
                "GateName": {
                        "tagname": "GateName",
                        "group": "0008",
                        "element": "2127",
                        "vr": "SH",
                        "vm": "1"
                },
                "GateNumber": {
                        "tagname": "GateNumber",
                        "group": "0008",
                        "element": "2128",
                        "vr": "IS",
                        "vm": "1"
                },
                "Manufacturer": {
                        "tagname": "Manufacturer",
                        "group": "0008",
                        "element": "0070",
                        "vr": "LO",
                        "vm": "1"
                },
                ...
```

5.3.1 Generierung fo-Dicom Code

Der generierte Code für fo-DICOM, zur Darstellung der Tags, bleibt weiterhin wie bereits dargestellt, mit dem Unterschied, dass nach der Umstellung auf den JSON-Ansatz ein zusätzliches Script erstellt wurde, das diesen Code aus der zuvor beschriebenen JSON-Datei generiert. Außerdem wurde das Script so erweitert, dass Getter und Setter Methoden für die jeweiligen Tags generiert werden können. Eine beispielhafte Ausgabe ist im Listing 5.3 zu sehen.

Listing 5.3: Ausgabe Getter und Setter fo-DICOM

```
public String getComponentName(){
        return dataset.Get<String>(DICONDETags.
            ComponentName);
}
public void setComponentName(String componentName){
        dataset.AddOrUpdate<String>(DicomVR.PN, DICONDETags
            .ComponentName, componentName);
}
public String getComponentIDNumber(){
        return dataset.Get<String>(DICONDETags.
            ComponentIDNumber);
}
public void setComponentIDNumber(String componentIDNumber){
        dataset.AddOrUpdate<String>(DicomVR.LO, DICONDETags
            .ComponentIDNumber, componentIDNumber);
}
public String getOtherComponentIDs(){
        return dataset.Get<String>(DICONDETags.
            OtherComponentIDs);
}
```

5.3.2 Generierung Dcm4Che Code

Bei der im Kapitel 3.5.4 beschriebenen Erweiterung von Dcm4che, bei der die DICONDE IOD „US-Image“ [31] implementiert wird um die Verwendbarkeit von Dcm4che für dieses Projekt zu evaluieren, kommen die hier aus dem DICONDE-Standard extrahierten Daten ebenfalls zum Einsatz. Dazu wird die im Kapitel 5.3 beschriebene Json-Datei mithilfe einer in Python entwickelten Software eingelesen. Anschließend werden daraus zum einen Klassen generiert, die für jedes DICONDE-Tag ein Integer-Attribut beinhaltet, das nach dem Tag-Namen benannt ist und dem als Wert die Tag-Nummer des entsprechenden DICONDE-Attributes zugewiesen wird. Außerdem wird zu jedem DICONDE-Tag ein Kommentar generiert, der die VR, VM, den Typ und die Beschreibung aus dem DICONDE-Standard zu dem jeweiligen DICONDE-Attribut beinhaltet. Diese Klasse ist ausschnittweise im Listing 5.4 zu sehen.

Listing 5.4: Ausgabe Tag-Generierung Dcm4Che

```
public class DicondeTag
{
        /**
        * (0010,0010) VR=PN VM=1 Type=2
        * Component Name or Part name
        */
        public static final int ComponentName =  0x00100010
            ;

        /**
        * (0010,0020) VR=LO VM=1-n Type=2
        * Component ID or Part ID
        */
        public static final int ComponentIDNumber =  0
            x00100020;

        ...

        /**
        * (0014,409D) VR=DS VM=1 Type=3
        * Scanner speed for motion in this direction
        */
        public static final int TranslationRateYDirection =
             0x0014409D;

        /**
        * (0014,409F) VR=DS VM=1 Type=3
        * Length of the common response between channels
        */
        public static final int ChannelOverlap =  0
            x0014409F;
}
```

Des Weiteren werden daraus Getter- und Setter-Methoden für die jeweiligen Klassen generiert, die die DICONDE-Module abbilden. Ein ausschnittweises Beispiel für das „NDE US Image Module“ ist im Listing 5.5 zu sehen.

Listing 5.5: Ausgabe Getter/Setter-Generierung Dcm4Che

```
...
public short getSamplesPerPixel(){
        return dataset.GetValue<short>(DICONDETags.
            SamplesPerPixel, 0);
}
public void setSamplesPerPixel(short samplesPerPixel){
        dataset.AddOrUpdate<short>(DicomVR.US, DICONDETags.
            SamplesPerPixel, samplesPerPixel);
}

...

public String getPhotometricInterpretation(){
        return dataset.GetValue<String>(DICONDETags.
            PhotometricInterpretation, 0);
}
public void setPhotometricInterpretation(String
    photometricInterpretation){
        dataset.AddOrUpdate<String>(DicomVR.CS, DICONDETags
            .PhotometricInterpretation,
            photometricInterpretation);
}

...

public double getPhysicalDeltaX(){
        return dataset.GetValue<double>(DICONDETags.
            PhysicalDeltaX, 0);
}
public void setPhysicalDeltaX(double physicalDeltaX){
        dataset.AddOrUpdate<double>(DicomVR.FD, DICONDETags
            .PhysicalDeltaX, physicalDeltaX);
}

...
```

5.3.3 Probleme im DICONDE-Standard

Aus der Ausgabe des Scripts lassen sich außerdem einige Probleme im DICONDE-Standard entnehmen. So ließen sich zum Zeitpunk der Erstellung dieser Arbeit folgende logische Konflikte aufdecken.

Component ID Number Dieses Tag ist im Standard mehrfach mit einer abweichenden VM definiert. Einmal in [33, S. 5] mit der VM „1-n“ und in [33, S. 6] und [33, S. 10] mit der VM „1“.

Transmit Transducer Sequence Dieses Tag ist im Standard mehrfach mit einer abweichenden Nummer definiert. Einmal in [31, S. 6] mit der Nummer „0014,4010“ und einmal in [31, S. 9] mit der Nummer „0014,4050“.

Receive Transducer Sequence Dieses Tag ist im Standard mehrfach mit einer abweichenden Nummer definiert. Einmal in [31, S. 7] mit der Nummer „0014,4011“ und einmal in [31, S. 9] mit der Nummer „0014,4051“.

Photometric Interpretation Dieses Tag ist im Standard mehrfach mit einer abweichenden VR und unterschiedlichen Nummern definiert. Zum einen ist es in [31, S. 4], [32, S. 4], [32, S. 7] und [30, S. 4] mit der VR „Code String (CS)“ und der Nummer „0028,0004“ definiert. Zum anderen ist das Tag in [29, S. 5] und [32, S. 6] der VR „Unsigned Short (US)“ und der Nummer „0028,0103“ definiert.

Date of Gain Calibration Dieses Tag ist im Standard mehrfach mit einer abweichenden VM definiert. Einmal in [29, S. 5] mit der VM „1-n“ und einmal in [32, S. 6] mit der VM „1“.

Time of Gain Calibration Dieses Tag ist im Standard mehrfach mit einer abweichenden VM definiert. Einmal in [29, S. 5] mit der VM „1-n“ und einmal in [32, S. 6] mit der VM „1“.

Drive Probe Sequence Dieses Tag ist im Standard mehrfach mit einer abweichenden VR definiert. Einmal in [30, S. 6] mit der VR „Long Text (LT)“ und einmal in [30, S. 9] mit der VR „SQ“.

Receive Probe Sequence Dieses Tag ist im Standard mehrfach mit einer abweichenden VR definiert. Einmal in [30, S. 7] mit der VR „LT“ und einmal in [30, S. 10] mit der VR „SQ“.

Die hier aufgedeckten Konflikte wurden zur Verbesserung des Standards an die DGZfP gemeldet, die diese an das entsprechende Komitee der Standardisierungsorganisation „ASTM International“ weitermeldet.

6 Evaluierung

In diesem Kapitel wird der Erfolg sowie die Auswirkungen und Akzeptanz der Arbeit untersucht.

6.1 Überprüfung der Anforderungen

Abgleichen der umgesetzten Anforderungen mit den geplanten Anforderungen.

6.2 Gegenwärtige Verwendung der DICONDE-Bibliothek

Bei der erste Anwendung der DICONDE-Bibliothek außerhalb dieser Arbeit handelt es sich um die Integration in die LabVIEW-Software (Siehe Abschnitt 2.5.1) US-Lab (Siehe Abschnitt 2.4.3).

Dabei wurden die folgenden Funktionen integriert:

Konvertieren von Bildern Diese Funktion dient dazu, Bilder der verschiedenen Darstellungen einer Messung anzufertigen und diese in das DICONDE-Format zu konvertieren.

Konvertieren von Messungen in das DICONDE-Format Diese Funktion dient dazu, mit der Software US-Lab aufgezeichnete Messungen in das DICONDE-Format zu konvertieren.

Laden von Messungen aus dem DICONDE-Format Diese Funktion dient dazu, Messungen aus dem DICONDE-Format in US-Lab zu laden.

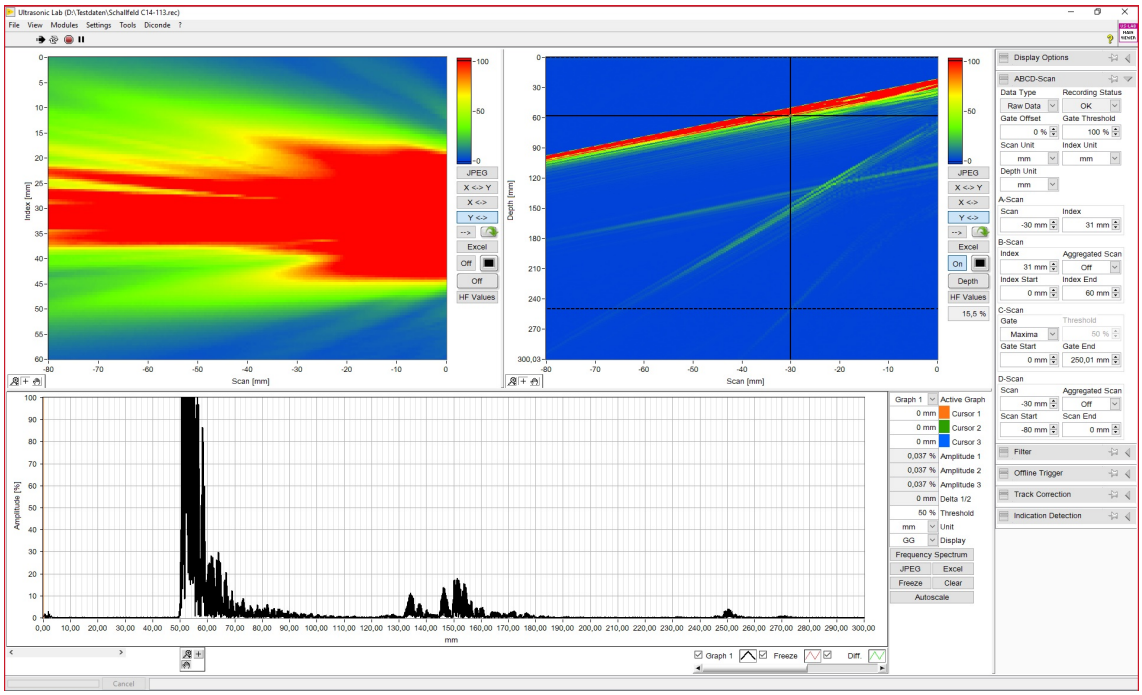

Abbildung 6.1: Darstellung einer Messung in US-Lab.

In der Abbildung 6.1 ist die Hauptansicht der Software US-Lab zu sehen. In dieser Ansicht können Messungen durchgeführt und dargestellt werden. Auf Basis der DICONDE-Bibliothek wurde nun auch die Funktion hinzugefügt, die verschiedenen Darstellungsformen der Messung als Bild im DICONDE-Format zu speichern.

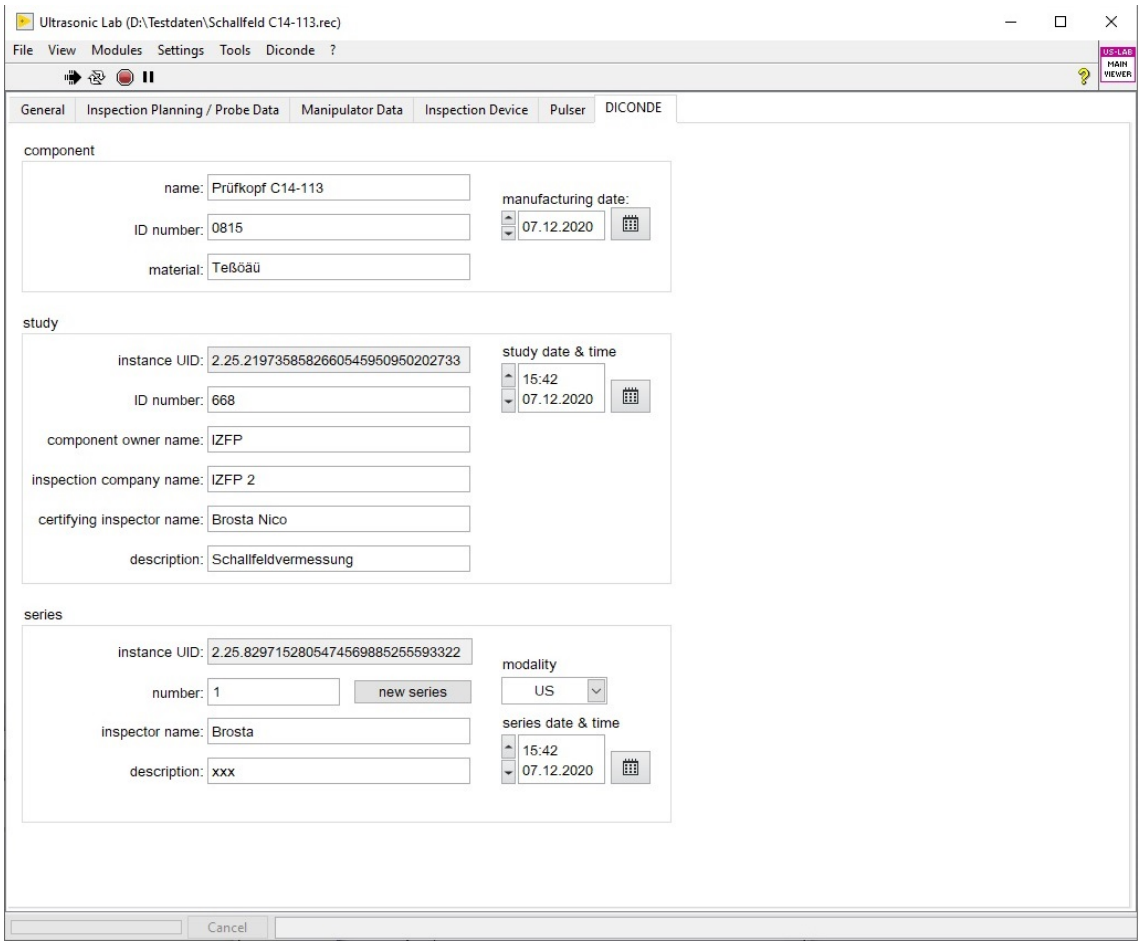

Abbildung 6.2: Bearbeiten von DICONDE-Detals in US-Lab

6.3 Überprüfung der Anforderungen

In diesem Kapitel werden die im Kapitel 3.3 gestellten Anforderungen überprüft. Wie im Folgenden zu sehen, konnte der Großteil der Anforderungen umgesetzt werden. Allerdings konnten die Anforderungen an die Performance des Ladens der Rohdaten nicht ganz erreicht werden. Außerdem wurde die geplante Schnittstellendokumentation nicht fertiggestellt.

6.3.1 Funktionale Anforderungen

Die funktionalen Anforderungen wurden dabei wie folgt evaluiert.

F1 Konvertierung von Bilddaten in das DICONDE-Format Die Konvertierung der Bilddaten wurde zum einen durch eine Integration der Funktionalität im Rahmen dieser Arbeit in die Softwarelösung Ultraschall-Software-Plattform evaluiert. Zum anderen wurde die Funktionalität von einem anderen Entwickler in die Software US-Lab integriert.

F2 Konvertierung von Ultraschall-Rohdaten in das DICONDE-Format Die Konvertierung der Ultraschall-Rohdaten wurde zum einen durch eine Integration der Funktionalität im Rahmen dieser Arbeit in die Softwarelösung Ultraschall-Software-Plattform

evaluiert. Zum anderen wurde die Funktionalität von einem anderen Entwickler in die Software US-Lab integriert.

F3 Laden von Bilddaten aus dem DICONDE-Format Das Laden der Bilddaten wurde noch nicht in eine bestehende Anwendung integriert. Zur Überprüfung der Anforderungen wurde die Funktionalität aber mit einem für diesen Zweck erstellten Konsolenanwendung getestet.

F4 Laden von Ultraschall-Rohdaten aus dem DICONDE-Format Das Laden der Rohdaten wurde von einem anderen Entwickler in die Softwarelösung US-Lab integriert und konnte somit überprüft werden.

F5 Hochladen von Daten auf einen PACS-Server Das Hochladen der Daten wurde zum einen durch eine Integration der Funktionalität im Rahmen dieser Arbeit in die Softwarelösung Ultraschall-Software-Plattform evaluiert. Zum anderen wurde die Funktionalität von einem anderen Entwickler in die Software US-Lab integriert.

F6 Herunterladen von Daten von einem PACS-Server Herunterladen von Daten von einem PACS-Server wurde noch nicht in eine bestehende Anwendung integriert. Zur Überprüfung der Anforderungen wurde die Funktionalität aber mit einem für diesen Zweck erstellten Konsolenanwendung getestet.

6.3.2 Nichtfunktionale Anforderungen

Die nichtfunktionalen Anforderungen wurden dabei wie folgt evaluiert.

NF1 Performance der Konvertierung von Ultraschall-Rohdaten in das DICONDE-Format Beim Konvertieren der Rohdaten können die folgenden Zeiten festgestellt werden.

Anzahl A-Scans	Rohdaten in MB	Zeit (mm:ss)
25000	84	00:09
50000	167	00:18
100000	334	00:35
200000	669	01:23

Tabelle 6.1: Zeiten beim Konvertieren der Rohdaten..

Die festgelegten Anforderungen an die Performance beim Konvertieren der Rohdaten sind somit erreicht.

NF2 Performance des Ladens von Ultraschall-Rohdaten aus dem DICONDE-Format Beim Laden der Rohdaten können die folgenden Zeiten festgestellt werden.

Anzahl A-Scans	Rohdaten in MB	Zeit (mm:ss) Alle Daten	Zeit (mm:ss) Positionsdaten
25000	84	00:29	00:27
50000	167	00:56	00:54
100000	334	01:52	01:48
200000	669	03:46	03:36

Tabelle 6.2: Zeiten beim Lesen der Daten.

Somit konnten die Anforderungen an die Performance beim Laden der Rohdaten nicht vollständig umgesetzt werden.

NF3 Dokumentation-Quellcode Die Dokumentation ist wie festgelegt fertiggestellt.

NF4 Schnittstellen-Dokumentation Eine Schnittstellendokumentation wurde bisher noch nicht umgesetzt.

NF5 Betreibssystem Die Anforderungen bezüglich des Betriebssystems wurden umgesetzt. Dies wird evaluiert, indem die Softwarelösungen die Funktionalitäten der Bibliothek verwenden auf dem Betriebssystem „Windows 10 in der Version 1909“ getestet werden.

NF7 Kompatibilität Die Kompatibilität mit den genanten Plattformen konnte hergestellt werden. Dies wurde für einzelne Funktionalitäten durch Integration in Softwarelösungen evaluiert, die auf der entsprechenden Plattform aufbauen.

6.4 Evaluierung Stakeholdermanagement

Im Folgenden wird der Erfolg der aus dem Stakeholdermanagement abgeleiteten Maßnahmen evaluiert. Die Evaluierung wird dabei nach den im Verlauf der Stakeholderanalyse ermittelten Gruppen aufgeschlüsselt.

Projektleitung Durch die alle 14 Tage stattfindenden Projektmeetings konnte Fortschritte klar kommuniziert werden und auf Wünsche bzw. Vorschläge der Projektleitung eingegangen werden. So konnte sicher gestellt werden, dass die im Verlauf dieser Arbeit erstellten Produkte sinnvoll in das Projektkonzept eingegliedert werden konnten.

Projektmanagement Für die Kommunikationsmaßnahmen gegenüber dem Projektmanagement gilt im Wesentlichen das Gleiche wie gegenüber der Projektleitung.

Softwareentwicklung Der alle 14 Tage stattfindende Austausch mit dem technischen Teil des Projektes kann ebenfalls als erfolgreich gewertet werden. Zum einen konnten so Anpassungen vorgenommen werden, um die bestmögliche Integration in die entsprechenden Softwarelösungen zu gewährleisten. Zum anderen führte dies zu einem besseren Überblick über das Gesamtprojekt.

Geschäftsmodellentwicklung Die Kommunikation Geschäftsmodellentwicklung trug ebenfalls zur Akzeptanz der aus dieser Arbeit hervorgehenden Produkte bei.

Insgesamt kann somit festgehalten werden, dass das Stakeholdermanagement im Verlauf dieser Arbeit erfolgreich verlaufen ist.

6.5 Fo-DICOM

Wie in der Arbeit beschrieben wurde zur Umsetzung des Projektes fo-DICOM verwendet. Mit diesem Framework konnte prinzipiell alles, wie geplant, umgesetzt werden. Im Folgenden wird aufgearbeitet, welche Vor- bzw. Nachteile diese Entscheidung mit sich bringt.

Folgende Vorteile können festgehalten werden.

Kompatibilität Positiv hervorzuheben ist, dass die Kompatibilität mit den bereits vorhandenen Softwarelösungen leicht herzustellen war, da das Framework in C# entwickelt ist.

Plattformunabhängig Fo-DICOM ist auf den meisten Windows und Linux Betriebssystemen lauffähig. Daher können die damit implementierten Systeme breitflächig eingesetzt werden.

Außerdem können folgende Nachteile festgehalten werden.

Keine DICONDE Spezialisierung Hinderlich für die Umsetzung dieses DICONDE Projektes ist, dass es sich eigentlich um ein DICOM-Framework handelt und die DICONDE-Funktionalitäten zusätzlich implementiert werden mussten.

7 Fazit und Ausblick

7.1 Fazit

Im Rahmen dieser Arbeit konnte ein Softwarebibliothek implementiert werden, die die grundlegenden Funktionalitäten zum Konvertieren von Ultraschall Rohdaten in das DICONDE-Format bereitstellt. Außerdem ist es möglich, fertig gerenderte Bilder von Ultraschallprüfungen im DICONDE-Format abzulegen. Die Funktionalitäten der Softwarebibliothek konnten außerdem zu großen Teilen erfolgreich evaluiert werden, indem sie in bereits bestehende Softwarelösungen integriert wurden. Dies geschah teilweise durch diese Arbeit und teilweise durch die Integration der Funktionalitäten innerhalb anderer Projekte.

Außerdem wurde im Verlauf der Arbeit eine Erweiterung des DICONDE-Standards verfasst, die zum Speichern von Ultraschall Rohdaten dient und bereits institutsintern genutzt wird. Des Weiteren wurden die im DICONDE-Standard definierten Tags in maschinenlesbare Form gebracht, wobei einige Fehler im DICONDE-Standard entdeckt wurden. Damit wurde mit dieser Arbeit sowohl ein Beitrag zur Verwendung des DICONDE-Standards innerhalb der Fraunhofer-Gesellschaft geleistet, als auch der Standard als solches weiterentwickelt.

7.2 Weiterentwicklung

Eine mögliche Weiterentwicklung besteht im Erweitern der Softwarebibliothek auf andere zerstörungsfreie Prüfverfahren wie etwa Thermografie oder Wirbelstromprüfung. Insbesondere Verfahren, deren Daten hauptsächlich als Bilder vorliegen, wie etwa Thermografie, können auf Basis der bestehenden Implementierung leicht in das DICONDE-Format konvertiert werden.

Außerdem steht die eventuelle Übernahme der im Rahmen dieser Arbeit erstellen Erweiterung des DICONDE-Standards, in den offiziellen Standard noch aus.

Des Weiteren kann die Softwarebibliothek in der Form weiterentwickelt werden, dass eine Möglichkeit gefunden wird, das Laden der Rohdaten performanter zu gestalten.

Auch kann die bisher noch nicht fertiggestellte Schnittstellendokumentation noch angefertigt werden.

7.3 Weiternutzung

Im Rahmen dieser Arbeit wurden die Funktionalitäten der DICONDE-Bibliothek bereits erfolgreich in bestehende Softwarelösungen integriert. Auch für die Zukunft ist abzusehen, dass die implementierten Funktionalitäten, in weitere bestehende als auch neue Softwarelösungen integriert werden. Insbesondere gilt dies mit der fortschreitenden Verbreitung und Umsetzung des DICONDE-Standards.

Literatur

[1] *A Brief History of Phased Array Testing*. URL: https://www.olympus-ims.com/en/ndt-tutorials/intro/breif-history/. (besucht am: 11.02.2021).

[2] Medical Imaging & Technology Alliance. *DICOM Standard Part 1*. URL: http://dicom.nema.org/medical/dicom/current/output/pdf/part01.pdf (besucht am 12. 06. 2020).

[3] Medical Imaging & Technology Alliance. *DICOM Standard Part 10*. URL: http://dicom.nema.org/medical/dicom/current/output/pdf/part10.pdf (besucht am 12. 06. 2020).

[4] Medical Imaging & Technology Alliance. *DICOM Standard Part 18*. URL: http://dicom.nema.org/medical/dicom/current/output/pdf/part18.pdf (besucht am 12. 06. 2020).

[5] Medical Imaging & Technology Alliance. *DICOM Standard Part 3*. URL: http://dicom.nema.org/medical/dicom/current/output/pdf/part03.pdf (besucht am 12. 06. 2020).

[6] Medical Imaging & Technology Alliance. *DICOM Standard Part 4*. URL: http://dicom.nema.org/medical/dicom/current/output/pdf/part04.pdf (besucht am 12. 06. 2020).

[7] Medical Imaging & Technology Alliance. *DICOM Standard Part 5*. URL: http://dicom.nema.org/medical/dicom/current/output/pdf/part05.pdf (besucht am 12. 06. 2020).

[8] Medical Imaging & Technology Alliance. *DICOM Standard Part 6*. URL: http://dicom.nema.org/medical/dicom/current/output/pdf/part06.pdf (besucht am 12. 06. 2020).

[9] Medical Imaging & Technology Alliance. *DICOM Standard Part 7*. URL: http://dicom.nema.org/medical/dicom/current/output/pdf/part07.pdf (besucht am 12. 06. 2020).

[10] *Belibteste Programmiersprachen*. URL: https://de.statista.com/statistik/daten/studie/678732/umfrage/beliebteste-programmiersprachen-weltweit-laut-pypl-index/. (besucht am: 02.02.2021).

[11] *DCMTK License*. URL: https://support.dcmtk.org/docs/file_copyright.html. (besucht am: 01.02.2021).

[12] *DCMTK dcmnet*. URL: https://support.dcmtk.org/docs/mod_dcmnet.html. (besucht am: 01.02.2021).

[13] *DCMTK docs*. URL: https://support.dcmtk.org/docs/. (besucht am: 01.02.2021).

[14] *DCMTK*. URL: https://dicom.offis.de/dcmtk.php.de. (besucht am: 31.01.2021).

[15] *Dcm4che Documentation*. URL: https://dcm4che.atlassian.net/wiki/spaces/proj/overview?mode=global. (besucht am: 02.02.2021).

[16] *Dcm4che License*. URL: https://dcm4che.atlassian.net/wiki/spaces/proj/pages/950309/License. (besucht am: 02.02.2021).

[17] *Dcm4che*. URL: https://www.dcm4che.org/. (besucht am: 02.02.2021).

[18] *Dicom Real World Model*. URL: https://de.wikipedia.org/wiki/Digital_Imaging_and_Communications_in_Medicine#/media/Datei:Real_World_Model.svg. (besucht am: 28.01.2021).

[19] Noel Dube. *Advances in Phased Array Ultrasonic Technilogy Applications*. Olympus NDT, 2007. ISBN: 0-9735933-4-2.

[20] *FileSystemWatcher Klasse*. URL: https://docs.microsoft.com/de-de/dotnet/api/system.io.filesystemwatcher?view=netcore-3.1. (besucht am: 05.10.2020).

[21] *Fraunhofer IZFP Institutsprofil*. URL: https://www.izfp.fraunhofer.de/de/institutsprofil.html. (besucht am: 30.01.2021).

[22] *Fraunhofer Intern*. (besucht am: 26.09.2020).

[23] Herbert Krautkrämer Josef Krautkrämer. *Werkstoff-Prüfung mit Ultraschall*. Bd. 5. Berlin, DE: Springer-Verlag, 2013. ISBN: 9783662109106.

[24] Donald E. Knuth. „Computer Programming as an Art“. In: *Communications of the ACM* 17.12 (1974), S. 667–673.

[25] *MVVM*. URL: https://docs.microsoft.com/en-us/previous-versions/msp-n-p/hh848246(v=pandp.10)?redirectedfrom=MSDN. (besucht am: 27.01.2021).

[26] *.NET Standard*. URL: https://docs.microsoft.com/de-de/dotnet/standard/net-standard. (besucht am: 31.01.2021).

[27] *.NET*. URL: https://dotnet.microsoft.com/learn/dotnet/what-is-dotnet. (besucht am: 31.01.2021).

[28] Sarath Kodagoda Nalika Ulapane Lasitha Piyathilaka. „Some Convolution and Scale Transformation Techniques to Enhance GPR Images“. In: *Kein Jornal / Preprint* (2019).

[29] *Standard Practice for Digital Imaging and Communication in Nondestructive Evaluation (DICONDE) for Digital Radiographic (DR) Test Methods*. Standard. West Conshohocken, PA: ASTM International, 2013.

[30] *Standard Practice for Digital Imaging and Communication in Nondestructive Evaluation (DICONDE) for Eddy Current (EC) Test Methods*. Standard. West Conshohocken, PA: ASTM International, 2014.

[31] *Standard Practice for Digital Imaging and Communication in Nondestructive Evaluation (DICONDE) for Ultrasonic Test Methods*. Standard. West Conshohocken, PA: ASTM International, 2018.

[32] *Standard Practice for Digital Imaging and Communication in Nondestructive Evaluation (DICONDE) for X-ray Computed Tomography (CT) Test Methods*. Standard. West Conshohocken, PA: ASTM International, 2013.

[33] *Standard Practice for Digital Imaging and Communication in Nondestructive Evaluation (DICONDE)*. Standard. West Conshohocken, PA: ASTM International, 2015.

[34] *Standard Practice for Flux Leakage Examination of Ferromagnetic Steel Tubular Products*. Standard. West Conshohocken, PA: ASTM International, 2015.

[35] *Tabula-Py*. URL: https://pypi.org/project/tabula-py/. (besucht am: 31.01.2021).

[36] *The Unified Code for Units of Measure*. Standard. Indianapolis, US: Regenstrief Institute, Inc. und The UCUM Organization, Mai 2017.

[37] *Von Java unterstütze Betriebssysteme*. URL: https://www.oracle.com/java/technologies/javase/products-doc-jdk11certconfig.html. (besucht am: 02.02.2021).

[38] *Was ist LabView*. URL: https://www.ni.com/de-de/shop/labview.html. (besucht am: 31.01.2021).

[39] *dicom*. URL: https://de.wikipedia.org/wiki/Digital_Imaging_and_Communications_in_Medicine#/media/Datei:ImageIOD.svg. (besucht am: 31.01.2021).

[40] *fo-dicom DicomFile*. URL: https://github.com/fo-dicom/fo-dicom/blob/development/DICOM/DicomFile.cs. (besucht am: 10.02.2021).

[41] *fo-dicom Github*. URL: https://github.com/fo-dicom/fo-dicom. (besucht am: 03.02.2021).

[42] *fo-dicom*. URL: https://fo-dicom.github.io/html/632e5303-a1e0-492f-8f6a-8b78e9037c40.htm. (besucht am: 27.01.2021).

Abbildungsverzeichnis

Tabellenverzeichnis

Listings

Abkürzungsverzeichnis

DICOM Digital Imaging and Communications in Medicine

CS Code String

LT Long Text

US Unsigned Short

PACS Picture Archiving and Communication System

DGZfP Deutsche Gesellschaft für Zerstörungsfreie Prüfung

MS-PL Microsoft Public License

DIMSE DICOM message service element

SOP Service-Object Pair

DICONDE Digital Imaging and Communication in Non Destructive Evaluation

ACR American College of Radiology

NEMA National Electrical Manufacturers Association

DCMTK DICOM-Toolkit

IOD Information Object Definition

IODs Information Object Definitions

ZfP Zerstörungsfreien Prüfung

USP Ultraschall-Software-Plattform

Fraunhofer IZFP Fraunhofer-Institut für Zerstörungsfreie Prüfverfahren

fo-DICOM Fellow Oak DICOM

VM Value Multiplicity

VR Value Representation

UCUM Unified Code for Units of Measure

WADO-RS Web Access to DICOM Objects by RESTful Services

WPF Windows Presentation Foundation

MVVM Model View ViewModel

OPC-UA Open Platform Communications Unified Architecture

TLS Transport Layer Security

MB Megabyte

Anhang

A Ultrasonic Waveform IOD

In diesem Abschnitt wird die in der Arbeit thematisierte Änderung der „US Image IOD “ [31] dargestellt. Zu beachten ist hierbei, dass es sich bei den grün dargestellten Feldern um die in diesem Vorschlag neu hinzugekommenen Felder handelt.

Ultrasonic Waveform IOD

February 25, 2021

1 Ultrasonic Waveform IOD

1.1 Ultrasonic Waveform IOD Description

Ultrasonic Waveform IOD is the specification of digitized Waveforms (A-Scans) taken from the surface of a specimen, which has been acquired by an US modality.

1.2 Ultrasonic Waveform IOD Definition

IE	Module	Reference	Usage
Component	Component	E2339, Section 7	M
	NDE Indication	E2339, Section 7	U
	NDE Geometry	E2339, Section 7	U
Study	General Study	E2339, Section 7	M
Series	General Series	E2339, Section 7	M
Equipment	NDE Equipment	E2339, Section 7	M
	NDE US Equipment	E2339, Section 7.2	U
	NDE Equipment Settings	E2339, Section 7.3	U
Image	NDE US Image	E2663, Section 7.1	M
Ultrasonic Waveform	Ultrasonic Waveform	Section 2	M

1.3 Ultrasonic Waveform IOD Content Constraints

1.3.1 Modality

The value of Modality (0008,0060) shall be US.

2 Ultrasonic Waveform Module

2.1 Ultrasonic Waveform Module Defintion

Attribute Name	Tag	Type	VR	Attribute Description
Scan Type	(0019,0010)	1	SH	Defines which scan type was used to create the stored AScans. Possible values are SECTORSCAN, LINEARSCAN, SINGLESCAN, MULTISCAN, COMPOUND_BSCAN, PWI
Wave Source Dimensions Sequence	(0019,0011)	1	SQ	Sequence of elements, each representing a physical dimension describing the generation point of the referenced wave.
>Dimension Number	(0019,0012)	1	UL	Number to reference this dimension.
>Dimension Name	(0019,0013)	1	ST	Human readable description of the spatial direction or angle.
>Dimension Code Value	(0019,0014)	1	ST	Dimension Code Value (0019,0013) is an identifier that is unambiguous within the Coding Scheme denoted by Dimesnsion Coding Scheme Designator (0019,0014) and Dimesnsion Coding Scheme Version (0019,0015).
>Dimesnsion Coding Scheme Designator	(0019,0015)	1	ST	An identifier of the version of the coding scheme if necessary to resolve ambiguity.
>Dimesnsion Coding Scheme Version	(0019,0016)	1	ST	An identifier of the version of the coding scheme if necessary to resolve ambiguity.
>Dimension Code Meaning	(0019,0017)	1	ST	Dimension Code Meaning (0019,0016) is text that has meaning to a human and conveys the meaning of the term defined by the combination of Dimension Code Value (0019,0013) , and Destination Coding Scheme Designator (0019,0014). Though such a meaning can be "looked up" in the dictionary for the coding scheme, it is encoded for the convenience of applications that do not have access to such a dictionary.
>Dimension Coding Scheme Name	(0019,0018)	1	ST	The coding scheme full common name
>Dimension Coding Scheme Responsible Organisation	(0019,0019)	1	ST	Name of the organization responsible for the Coding Scheme. May include organizational contact information.
>Value Type	(0019,0020)	1	ST	Defines whether the value is saved as text or numerically. Defined Terms: FLOATINGPOINT, RATIONALNUMERATOR
Waveform Sequence	(5400,0100)	1	SQ	Sequence of Items, each representing one waveform multiplex group. One or more Items shall be included in this Sequence. Ordering of Items in this Sequence is significant for external reference to specific multiplex groups.

>Wave Source Values Sequence	(0019,0021)	1	SQ	Each item contains a value for the dimension defined by an item of the Wave Source Position Sequence (0019,0012).
>>Referenced Dimension	(0019,0022)	1	UL	Indicates to which item in Wave Source Dimension Sequence (0019,0012) the current item refers.
>>Numeric Value	(0019,0023)	1C	ST	This is the Value component of a Name/Value pair when the Value Type (0019,0019) implied a set of one or more numeric values. Required if (0019,0019) is set to Numeric.
>>Rational Numerator Value	(0019,0024)	1C	SL	The integer numerator of a rational representation of Numeric Value (0019,0015). Encoded as a signed integer value. The same number of values as Numeric Value (0019,0015) shall be present. Required if (0019,0013) is set to RATIONALNUMERATOR.
>>Floating Point Value	(0019,0025)	1C	FD	The floating point representation of Numeric Value (0019,0015). The same number of values as Numeric Value (0040,A30A) shall be present. Required if (0019,0013) is set to FLOATINGPOINT.
>Multiplex Group Time Offset	(0018,1068)	1C	DS	Offset time in milliseconds from a reference time (see Section C.10.9.1.1). Required if Acquisition Time Synchronized (0018,1800) value is Y; may be present otherwise.
>Trigger Time Offset	(0018,1069)	1C	DS	Offset time in milliseconds from a synchronization trigger to the first sample of a waveform multiplex group. May be positive or negative. Required if waveform acquisition is synchronized to a trigger.
>Trigger Sample Position	(0018,106E)	3	UL	Sample number whose time corresponds to a synchronization trigger (see Section C.10.9.1.2).
>Waveform Originality	(003A,0004)	1	CS	See Section C.10.9.1.3. Enumerated Values: ORIGINAL, DERIVED
>Number of Waveform Channels	(003A,0005)	1	US	Number of channels for this multiplex group.
>Number of Waveform Samples	(003A,0010)	1	UL	Number of samples per channel in this multiplex group.
>Sampling Frequency	(003A,001A)	1	DS	Frequency in Hz
>Multiplex Group Label	(003A,0020)	3	SH	Label for multiplex group
>Channel Definition Sequence	(003A,0200)	1	SQ	Sequence of Items, with one Item per channel (see Section C.10.9.1.4). One or more Items shall be included in this Sequence. Ordering of Items in this Sequence is significant for reference to specific channels.

>>Waveform Channel Number	(003A,0202)	3	IS	Equipment physical channel number used for acquisition.
>>Channel Label	(003A,0203)	3	SH	Text label for channel, which may be used for display purposes
>>Channel Status	(003A,0205)	3	CS	One or more values for the status of this channel within this SOP Instance. Defined Terms: OK TEST DATA DISCONNECTED QUESTIONABLE INVALID UNCALIBRATED UNZEROED Precise location of a change in status may be noted in an Annotation.
>> Channel Source Sequence	(003A,0208)	1	SQ	A coded descriptor of the waveform channel source (metric, anatomical position, function, and technique). Only a single Item shall be included in this Sequence. (see Section C.10.9.1.4.1)
>>>Include Table 8.8-1 "Code Sequence Macro Attributes".				Baseline CID determined by IOD specialization
>> Channel Source Modifiers Sequence	(003A,0209)	1C	SQ	Sequence of Items that further qualify the Waveform Source. One or more Items shall be included in this Sequence. Ordering of Items in this Sequence may be semantically significant. Required if Channel Source Sequence (003A,0208) does not fully specify the semantics of the source.
>>>Include Table 8.8-1 "Code Sequence Macro Attributes".				
>> Source Waveform Sequence	(003A,020A)	3	SQ	A Sequence that provides reference to a DICOM waveform from which this channel was derived. One or more Items are permitted in this Sequence.
>>>Include Table 10-11 "SOP Instance Reference Macro Attributes"				
>>> Referenced Waveform Channels	(0040,A0B0)	1		Identifies the waveform multiplex group and channel within the referenced SOP Instance. Pair of values (M,C).
>>Channel Derivation Description	(003A,020C)	3		Additional description of waveform channel derivation
>>Channel Sensitivity	(003A,0210)	1C		Nominal numeric value of unit quantity of sample. Required if samples represent defined (not arbitrary) units.

>>Channel Sensitivity Units Sequence	(003A,0211)	1C	SQ	A coded descriptor of the Units of measure for the Channel Sensitivity. Only a single Item shall be included in this Sequence. (see Section C.10.9.1.4.2). Required if Channel Sensitivity (003A,0210) is present.
>>>Include Table 8.8-1 "Code Sequence Macro Attributes"				DCID 82 "Units of Measurement".
>>Channel Sensitivity Correction Factor	(003A,0212)	1C	DS	Multiplier to be applied to encoded sample values to match units specified in Channel Sensitivity (003A,0210) (e.g., based on calibration data) (see Section C.10.9.1.4.2). Required if Channel Sensitivity (003A,0210) is present.
>>Channel Baseline	(003A,0213)	1C	DS	Offset of encoded sample value 0 from actual 0 using the units defined in the Channel Sensitivity Units Sequence (003A,0211). Required if Channel Sensitivity (003A,0210) is present.
>>Channel Time Skew	(003A,0214)	1C	DS	Offset of first sample of channel from waveform multiplex group start time, in seconds (see Section C.10.9.1.4.3) Required if Channel Sample Skew is not present.
>>Channel Sample Skew	(003A,0215)	1C	DS	Offset of first sample of channel from waveform multiplex group start time, in samples (see Section C.10.9.1.4.3) Required if Channel Time Skew is not present.
>>Channel Offset	(003A,0218)	3	DS	Additional offset of first sample of channel to be used in aligning multiple channels for presentation or analysis, in seconds (see Section C.10.9.1.4.3)
>>Waveform Bits Stored	(003A,021A)	1	US	Number of significant bits within the waveform samples (see Section C.10.9.1.4.4)
>>Filter Low Frequency	(003A,0220)	3	DS	Nominal 3dB point of lower frequency of pass band; in Hz
>>Filter High Frequency	(003A,0221)	3	DS	Nominal 3dB point of upper frequency of pass band; in Hz
>>Notch Filter Frequency	(003A,0222)	3	DS	Center frequency of notch filter(s); in Hz
>>Notch Filter Bandwidth	(003A,0223)	3	DS	Nominal 3dB bandwidth of notch filter(s); in Hz
>>Channel Minimum Value	(5400,0110)	3	OB	Minimum valid sample value as limited by the acquisition equipment (see Section C.10.9.1.4.5)
>>Channel Maximum Value	(5400,0112)	3	OB	Maximum valid sample value as limited by the acquisition equipment (see Section C.10.9.1.4.5)

>Waveform Bits Allocated	(5400,1004)	1	US	Size of each waveform data sample within the Waveform Data; See Section C.10.9.1.5
>Waveform Sample Interpretation	(5400,1006)	1	CS	Data representation of the waveform data points. See Section C.10.9.1.5.
>Waveform Padding Value	(5400,100A)	1C	OW	Value of waveform samples inserted in channels when input is absent or invalid. Required if acquisition equipment inserts padding. See Section C.10.9.1.6.
>Waveform Data	(5400,1010)	1	OW	Encoded data samples - channel multiplexed. See Section C.10.9.1.7
Waveform Data Display Scale	(003A,0230)	3	OW	The recommended time-based waveform data display scale in units of mm/s (see Section C.10.9.1.8).
5>>Referenced Waveform Channels	(0040,A0B0)	1		Identifier of the displayed channel, specified as a pair of values

DICONDE Digital Imaging and Communication in Non Destructive Evaluation

DICOM Digital Imaging and Communications in Medicine

B Fragebogen Stakeholder

In diesem Abschnitt wird der im Rahmen der Stakeholderanalyse erstellte Fragebogen dargestellt.

Stakeholderanalyse

Persönliche Daten

Vorname:	
Name:	
Abteilung:	
E-Mail:	

Tätigkeit

Erstellen, verarbeiten oder speichern Sie im Rahmen Ihrer Tätigkeit Daten die im Zusammenhang mit Bildgebenden Prüfverfahren entstanden sind?

- ◯ Ja
- ◯ Nein

Falls ja, beschreiben Sie bitte kurz.

Welche Software verwenden sie zurzeit zum erstellen, verarbeiten und speichern der erwähnten Daten?

DiNA 4.0

Arbeiten Sie zur Zeit im Rahmen des DiNA 4.0-Projektes?

- ◯ Ja
- ◯ Nein

Falls ja, welche Rolle haben Sie im Rahmen des Projektes inne?

Steht Ihre Tätigkeit in einer anderen Art und Weise mit dem Dina4-Projek in Verbindung?

- ◯ Ja
- ◯ Nein

Falls ja, beschreiben Sie bitte kurz.

DICONDE

Falls Sie selbst Software entwickeln, haben sie den DICONDE-Standard bereits in Ihrer Software implementiert, bzw. planen diesen zu implementieren?

Verwenden Sie bereits Software die den DICONDE-Standard implementieren oder planen Sie eine solche Software zu verwenden?

- ◯ Ja
- ◯ Nein

Kolophon

Dieses Dokument wurde mit der LaTeX-Vorlage für Abschlussarbeiten an der htw saar im Bereich Informatik/Mechatronik-Sensortechnik erstellt (Version 2.1). Die Vorlage wurde von Yves Hary und André Miede entwickelt (mit freundlicher Unterstützung von Thomas Kretschmer, Helmut G. Folz und Martina Lehser). Daten: (F)10.95 – (B)426.79135pt – (H)688.5567pt