Universität Leipzig

Fakultät für Mathematik und Informatik

Institut für Medizinische Informatik, Statistik und Epidemiologie

Synthetische Daten in der Medizin - Erzeugung künstlicher Patientendaten zur Unterstützung der klinischen Forschung in Deutschland

Bachelorarbeit

Leipzig, August 2020

vorgelegt von:
Marcus Rudolph

Studienrichtung:
Medizinische Informatik

Referent: Prof. Dr. Alfred Winter

Betreuer: Dr. Frank Meineke

Institut für Medizinische Informatik, Statistik und Epidemiologie (IMISE)

Inhaltsverzeichnis

Abkürzungsverzeichnis

BDSG	Bundesdatenschutzgesetz
CCDA	Consolidated Clinical Document Architecture (auch C-CDA)
CDC	Centers Disease Control and Prevention
CDA	Clinical Document Architecture
CEN	Europäische Komitee für Normung
CSV	Comma-separated values
DICOM	Digital Imaging and Communications in Medicine
DIMDI	Deutsches Institut für Medizinische Dokumentation und Information
DIN	Deutsches Institut für Normung
DSGVO	Datenschutz-Grundverordnung der Europäischen Union

EHR	Electronic health records
FHIR	Fast Healthcare Interoperability Resources
GMDS	Deutsche Gesellschaft für Medizinische Informatik, Biometrie und Epidemiologie e.V.
HL7	Health Level 7
HTTP	Hypertext Transfer Protocol
ICD	International Statistical Classification of Diseases and Related Health Problems
ID	Identifikator (bzw. Identifikationsnummer)
ISO	Internationale Organisation für Normung
KIS	Krankenhausinformationssystem
LOINC	Logical Observation Identifiers Names and Codes
MDR	Medical Device Regulation
MII	Medizininformatik-Initiative Deutschlands
MPG	Medizinproduktegesetz
NIH	National Institutes of Health
PACS	Picture Archiving and Communication System
PADARSER	Publicly Available Data Approach to the Realistic Synthetic EHR
PHI	Protected Health Information
PLZ	Postleitzahl
RS-EHR	Realistic Synthetic EHR
REST	Representational State Transfer
SNOMED	Systematisierte Nomenklatur der Medizin
SVN	Sozialversicherungsnummer
UCUM	Unified Code for Units of Measure

Zusammenfassung

In meiner Bachelorarbeit habe ich untersucht wie synthetische Daten in der Medizin, besonders für den Einsatz im deutschsprachigen Raum, erzeugt werden können. Ein Problem war die Lücke in der Verfügbarkeit hochwertiger Testdaten für die Medizininformatik. Es gab keine Möglichkeit syntaktisch korrekte (Test-)Datensätze für die Nutzung im medizinischen Umfeld (in Deutschland) nach Bedarf zu erzeugen und beliebig zu skalieren. Da Datenschutz besonders im medizinischen Umfeld eine große Rolle spielt, dürfen Datensätze nur anonymisiert weiterverwendet werden. Darunter „leidet" zum Teil die Softwareentwicklung, indem zusätzliche Hindernisse auftreten.

Ein Ausweg kann die Nutzung synthetischer Daten sein, die a priori keinen Datenschutzbestimmungen unterliegen. Um solche Daten zu generieren gibt es einige Ansätze, von denen viele leider nicht für z.B. die Erzeugung strukturierte Patientendaten anwendbar sind oder aber keine fertige Software liefern. Als derzeit beste Alternative hat sich dabei Synthea herausgestellt, da hier medizinische Daten (elektronische Gesundheitsakten) in den benötigten Standards, wie HL7 FHIR, erstellt werden können. Diese Software kann synthetische Patientendaten algorithmisch generieren und die Daten in aktuellen benötigten Formaten (wie beispielsweise FHIR R4 oder CCDA) ausgeben. Synthea wurde jedoch in den USA entwickelt und ist daher auf das US-amerikanische Gesundheitswesen ausgelegt. Daher musste das Programm für den Einsatz in Deutschland angepasst werden. In dieser Arbeit habe ich daher exemplarisch gezeigt wie solche Änderungen aussehen müssen und welche Daten ergänzt werden können, damit ein späterer vollumfänglicher Umbau der Software möglich ist. Schlussendlich ist es dadurch möglich künstliche Patientendaten zu generieren die den hiesigen Anforderungen entsprechen und realistische Eigenschaften besitzen (wie deutsche Namen, Adressen, etc.). Außerdem werden auch die Formate und Kodierungen erzeugt, die die deutsche Medizininformatik-Initiative (MII) in ihrem Kerndatensatz vorschlägt.

Anschließend wurde überprüft, ob die neu generierten Dateien fehlerfrei verarbeitet werden können. Dazu habe ich mit einem FHIR Testserver verifiziert, dass die Daten syntaktisch korrekt sind und damit in den gewünschten Formaten vorliegen. Die in dieser Arbeit gemachten Anpassungen bieten daher eine Lösung für das Problem der fehlenden (Test-)Daten, da es nun eine Möglichkeit gibt synthetische Datensätze für die Nutzung im medizinischen Umfeld in Deutschland zu erzeugen. Somit können zukünftig künstliche Patientendaten zur Unterstützung der klinischen Forschung in Deutschland beitragen.

1 Einleitung

1.1 Gegenstand und Motivation

Medizinische Anwendungssysteme operieren zwangsläufig auf hochsensiblen persönlichen Daten. Anwendungs- und Schnittstellen-Entwicklung benötigen jedoch Testdaten, auch um den zunehmenden Regulierungen des Entwicklungsprozesses für Medizinprodukte (und damit auch für die Software) gerecht zu werden. Daher besteht ein sehr hoher Bedarf an Personen- bzw. Patientendaten um unter anderem Software im Gesundheitswesen zu testen, zu demonstrieren oder um Personal zu schulen.

Auch für Testzwecke eingesetzte Daten, sofern auf reale Personen beziehbar, unterliegen weiterhin der DSGVO und können daher nicht uneingeschränkt, persönliche Daten sogar oft gar nicht benutzt werden(1, 2). Ein gängiges Verfahren zur Gewährleistung des Datenschutzes ist die Anonymisierung der Personenbezüge durch unkenntlich machen von Namen, Adressen, etc. (3). Diese und viele weitere Information über den Gesundheitszustand und die Behandlungen von Personen werden z.B. in den USA als PHI („protected health information“) gesammelt (4), dürfen aber nur anonymisiert weiter verwendet werden. Anonymisierte Patientendaten werden von einer Reihe von staatlichen oder kommerziellen Unternehmen (auch Versicherungen) oder klinischen Gruppen gekauft und verkauft (5). Allerdings sind selbst solche Testdatensätze aus externen Quellen, bei denen der Personenbezug entfernt wurde rechtlich schwer zu nutzen. Kommt es außerdem auf die Auswertung bestimmter persönlicher Kriterien an, wie z.B. die Verteilung von Patienten auf Orte oder Ähnliches, werden Tests nicht aussagekräftig sein, wenn der Bezug zum Wohnort entfernt wurden ist. Auch besteht ein zunehmendes Restrisiko, dass doch noch ein Rückschluss auf den Personenbezug möglich ist, was wiederum eine Verletzung des Datenschutzes darstellen würde. Gerade im medizinischen Umfeld lassen sich Patienten z.B. anhand der Kombination weniger Laborwerte re-identifizieren, da solch eine Kombination innerhalb eines Krankenhauses schon einzigartig sein könnte.

Eventuell dennoch verfügbare reale (anonymisierte) Datensätze hingegen sind statisch und nicht für das deutsche Gesundheitswesen angepasst, sie liegen also zum Teil nicht in den benötigten Standards bezüglich Formaten oder Kodierungen (wie z.B. ICD, SNOMED, LOINC) vor. Oder sie unterliegen wiederum Nutzungseinschränkungen. Wodurch es in der Realität schwer ist echte Datensätze zu finden, die man zum Testen bestimmter Software nutzen kann und die gleichzeitig eine gewisse klinische Aussagekraft haben. Des Weiteren muss man beachten, dass für viele Anwendungsfälle eine sehr große Datenmenge benötigt wird, wenn beispielsweise ein Datenbankmanagementsystem für Krankenhäuser getestet werden soll, sind auch Datensätze mit weniger als 10.000 Patienten nicht ausreichend für Last- oder Performancetests. Einen Datensatz dieser Größe aus echten (anonymisierten) Daten zu bekommen gestaltet sich dementsprechend schwierig.

Ersatzweise können künstlich erzeugte, sogenannte synthetische, Daten in verschiedenen Phasen der Entwicklung eingesetzt werden. Solche Daten sind maschinell generiert und stammen daher nicht aus realen Ereignissen. Sie werden auf verschiedenen Wegen erstellt und können je nach Ansatz eine große Ähnlichkeit zu echten Personendaten aufweisen (6). Mit den synthetischen Daten können beispielsweise Schnittstellen oder Modelle auch in Grenzbereichen (z.B. Fehler, Länge von Zeichenketten, o.Ä.) ausgetestet werden, die in den realen Einsatzszenarien nur selten entstehen (7).

Im Gegensatz zur Erzeugung künstlicher Daten von Grund auf können auch reale Datensätze algorithmisch und damit maschinell synthetisiert werden. Das bedeutet, dass Originaldaten (zum Beispiel sensible personenbezogene Daten) in eine synthetische Repräsentation überführt werden, die die Originaldaten bestmöglich repräsentiert und wesentliche Eigenschaften erhält (8, 9).

Ein synthetisierter Datensatz besteht dann aus Daten künstlicher Subjekte, diese wurden nicht durch direkte Messung erhoben, sondern sind das Resultat eines algorithmischen Mechanismus (welcher sich allerdings auf die Daten aus direkter Messung bezieht). Solche Verfahren basieren zum Beispiel auf probabilistischen Modellen, künstlichen neuronalen Netzwerken oder der multiplen Imputation. Letzteres ist beispielsweise ein statistisches Verfahren zur Ersetzung fehlender Werte in Datensätzen, bei welchem diese durch mehrere plausiblen Werte ersetzt werden. Dieser und andere Ansätze werden oft mit Hilfe von maschinellem Lernen durchgeführt und sind daher meist sehr aufwendig. Doch können diese Methoden so auch das Risiko der Re-Identifizierung von Datensubjekten erheblich senken. (9)

Durch synthetische Daten können Beschränkungen in der Nutzung von Datensätzen, die durch deren Sensibilität oder wegen gesetzlichen Regulierungen nicht genutzt werden dürfen, umgangen werden (7). Außerdem sind die Anwender nicht an bestimmte Bedingungen, die durch reale Daten bestimmt werden gebunden und Entwickler und Anwender sind durch den Einsatz von synthetischen Daten nahezu völlig frei in ihrer Nutzung und dem Datenaustausch (7). Somit könnten synthetische Patientendaten zum Testen (zum Beispiel von medizinischer Software) genutzt werden und bieten einen Ausweg für die oben genannte Problematik. Für die Generierung syntaktisch und strukturell plausibler Patientendatensätze, z.B. in Form von FHIR Ressourcen (10) existieren frei verfügbare Generatoren. Ein Beispiel für einen open-source Generator, der u.a. vollständig synthetische Patientendatensätze erzeugen kann, ist Synthea™ (5).

Synthea simuliert den kompletten Lebenslauf eines virtuellen Patienten und weist diesem mögliche Erkrankungen bzw. Vorfälle auf Basis von publizierten oder geschätzten Wahrscheinlichkeitsverteilungen zu. Dabei werden laut den Entwicklern (MITRE Corporation) mögliche Erkrankungen in Modulen beschrieben, die den Krankheitsverlauf plausibel darstellen. Es werden u.a. Medikationen, Diagnosen und Befunde erzeugt, auf Basis von Inzidenzen und Prävalenzen (z.B. zu Alter, Größe, Geschlecht oder Einkommen). Die Wahrscheinlichkeitsverteilungen in den Modulen werden von Statistiken aus Daten der realen Welt, die vom CDC, dem NIH und anderen Forschungsquellen

gesammelt wurden, zur Verfügung gestellt. Ziel ist es, syntaktisch und semantisch annotierte, synthetische und realistische (aber nicht echte) Patientendaten und zugehörige Gesundheitsakten auszugeben, die formal alle Aspekte von Daten des Gesundheitswesens abdecken. (11)

Solche Generatoren sind flexibel und können in verschiedenen Anwendungsfällen genutzt werden. Die resultierenden (Patienten-)Daten sind dann frei von Kosten-, Datenschutz- und Sicherheitsbeschränkungen und können u.a. im HL7-Standard FHIR ausgegeben werden. Diese Daten können ohne Einschränkung für eine Vielzahl von Verwendungszwecken in Wissenschaft, Forschung und Industrie eingesetzt werden (5). Derzeit gibt es über 90 verschiedene solcher Module, mit denen plausible Krankengeschichten von Patienten simuliert werden können. Wobei diese auch fortlaufend von der nutzenden Community über ein öffentliches Web-Frontend (12) erweitert und ergänzt werden können.

Allerdings ist dieser Generator in den USA entstanden und ist deshalb auf das amerikanische Gesundheitswesen und US-amerikanische demographische Gegebenheiten ausgelegt. Was heißt, dass die erzeugten Patienten englische Namen, Adressen, Telefonnummern et cetera haben. Auch die Kodierungen von zum Beispiel Laborwerten oder Medikationen entsprechen US-amerikanischen Standards (z.B. Nutzung der ICD9 in den USA statt der ICD-10-GM in Deutschland).

Auch und gerade im Hinblick auf die deutsche Medizininformatik-Initiative (13) existiert derzeit eine gravierende Lücke in der Verfügbarkeit hochwertiger Testdaten. Daher müsste ein solches Systeme für das deutsche Gesundheitswesen angepasst bzw. erweitert werden, damit die erzeugten Ressourcen im richtigen Format und mit den gewünschten Standards/Kodierungen vorliegen. Anschließend können generierte synthetische Daten auch im deutschsprachigen Raum genutzt werden. Ein öffentlich verfügbarer, flexibler Generator ist hierbei der Bereitstellung statischer Datensätze vorzuziehen und die erzeugten Daten können dann für Interoperabilitätstests, Lehre, Demonstratoren oder Performance Tests genutzt werden.

1.1.1 Gegenstand

- Med. Anwendungssysteme benötigen viele Testdaten in der Entwicklung
- Reale Datensätze sind schwer zu beschaffen und können nicht uneingeschränkt genutzt werden, auch eine Anonymisierung der Daten ist oft nicht ausreichend
- Einen Ausweg bieten synthetische Daten, die auch für die Entwicklung medizinischer Software interessant sind
- Es existieren Generatoren für synthetische Daten, die unterschiedliche Ansätze verfolgen und demzufolge verschiedene Ergebnisse liefern
- In dieser Arbeit sollen synthetische Daten in der Medizin betrachtet werden, insbesondere deren Erzeugung und Nutzung

1.1.2 Problematik

- Testdaten, die aus realen Daten abgeleitet werden unterliegen auch der DSGVO, die Anonymisierung der Personenbezüge ist oft nicht ausreichend
- Datensätze für Software-Tests, Demonstrationen, usw. unterliegen weiteren Restriktionen
- Entwickler benötigen also Alternativen, wie z.B. synthetische Daten
- Verfügbare Generatoren für künstliche (Patienten-)Daten müssen aber erst an die deutschen Gegebenheiten angepasst werden
- Daher die aktuell bestehende Lücke in der Versorgung mit (deutschen) Testdatensätzen im medizinischen Umfeld

1.1.3 Motivation

- Synthetische Daten können die Entwicklung von Software vorantreiben
- Gerade im medizinischen Bereich könnte so das Testen und Demonstrieren neuer Software stark vereinfacht werden
- Es soll ein Überblick über synthetische Daten gegeben und ein Generator für medizinischen Datensätze betrachtet werden
- Dieser soll dann für den deutschen Gebrauch angepasst werden
- Dadurch kann der Einsatz synthetischer Daten (im gewünschten Datenstandard) besonders die Arbeit der deutschen Medizininformatik erleichtern
- Somit kann die klinische Forschung weiter unterstützt werden

1.2 Problemstellung

Ein bestehendes Problem ist das mehr Daten zum Testen bzw. Entwickeln von Software benötigt wird als vorhanden ist. Es besteht hier also eine Lücke in der Verfügbarkeit hochwertiger Testdaten für die Medizininformatik. Ein Ausweg kann die Nutzung synthetischer Daten sein, wie oben erwähnt. Jedoch gibt es eine Reihe von Generatoren für synthetische Daten, vor allem zur Erzeugung von Kundendaten für ein Unternehmen (14). Diese werden häufig genutzt, wenn es um das Testen von z.B. Datenbanken geht. Jedoch eignen sich die meisten davon kaum für den Einsatz im medizinischen Umfeld, sodass ein erstes Teilproblem darin besteht einen Generator für synthetische Daten zu finden, der sich für den Einsatz in der medizinischen Informatik überhaupt eignet. Und damit verbunden auch die Daten in einem gewünschten Format liefern kann.

Des Weiteren ist der Großteil frei verfügbarer Generatoren im englisch-sprachigen Raum entstanden und nutzt deshalb auch US-standards und -kodierungen, welche im deutschen Gesundheitswesen nicht oder nur selten benutzt werden. Außerdem können oftmals nur Patienten mit den im anglo-amerikanischen Raum üblichen Merkmalen und Namen erzeugt werden, was für Anwendungen im deutschen Sprachraum störend wirkt. Hier müssen also Anpassungen an einem solchen Generator vorgenommen werden, die besonders die Kodierungen in den erzeugten Daten an die hier gebräuchlichen Standards angleicht. Dazu können exemplarisch einige Änderungen an dem originalen Generator durchgeführt werden, um ihn so ins deutsche zu überführen.

Eine Anleitung was angeglichen werden muss um nutzbare synthetische Daten für den deutschsprachigen Raum zu erzeugen, könnte dann einen späteren systematischen vollumfänglichen Umbau unterstützen, der nicht Teil dieser Arbeit ist.

- Problem P1: Fehlende Testdaten im Bereich der Medizininformatik

  Es gibt aktuell keine Möglichkeit syntaktisch korrekte (Test-)Datensätze für die Nutzung im medizinischen Umfeld (in Deutschland) nach Bedarf zu erzeugen bzw. beliebig zu skalieren.

1.3 Zielsetzung

- Ziele zur Lösung von Problem P1:
  - Ziel Z1: Generatoren finden/vergleichen

    Es muss erst ein geeigneter Generator für den Einsatz in der Medizin gefunden werden, der den hiesigen Anforderungen gerecht wird.
  - Ziel Z2: Generator anpassen

    Im Anschluss müsste dieser Generator für synthetische Daten angepasst werden, um damit nützliche Daten für den Einsatz im deutschen Gesundheitswesen erzeugen zu können. Dabei sollte auch erläutert werden was man im Allgemeinen angleichen muss damit synthetische Daten mit deutschen Standards erzeugt werden können.
  - Ziel Z3: Überprüfen der Daten

    Am Schluss kann auch noch überprüft werden ob alle generierten Daten fehlerfrei verarbeitet werden können. Es könnte also eventuell getestet werden, ob die erzeugten Testdaten auch die gewünschten Eigenschaften besitzen.

1.4 Aufgabenstellung

- Aufgaben zu Ziel Z1:
  - Aufgabe A1.1: Ermitteln der Anforderungen an synthetische Daten im Gesundheitswesen (inkl. Recherche).
  - Aufgabe A1.2: Einen (open-source) Generator finden, der diesen Anforderungen gerecht wird und diesen installieren.
- Aufgaben zu Ziel Z2:
  - Aufgabe A2.1: Testdaten erzeugen und darauf notwendige Veränderungen herausarbeiten. (Was muss wie verändert werden?)
  - Aufgabe A2.2: Den in A1.2 gefundenen Generator dahingehend anpassen, soweit möglich (arbeiten am Quellcode).

    - Aufgabe A2.3: Schritte/Änderungen protokollieren die nötig sind um Generator aus dem US-raum für deutsche User nutzbar zu machen.
- Aufgaben zu Ziel Z3:
    - Aufgabe A3.1: Exemplarisch synthetische Daten erzeugen
    - Aufgabe A3.2: Überprüfen ob Daten korrekt verarbeitet werden können (können Daten fehlerfrei genutzt werden und sind syntaktisch korrekt?).
    - Aufgabe A3.3: Dokumentieren und Bereitstellen der hier entwickelten Lösung für die nutzende Community in geeigneter Form, falls möglich.

2 Grundlagen

2.1 Software in der Medizin

IT-Systeme und Software sind aus dem Gesundheitswesen und der modernen Medizin nicht mehr wegzudenken. Sei es unterstützend zur Dokumentation, in Bereichen der Diagnostik und Therapie oder zum Verarbeiten von Informationen allgemein. Dabei hat die medizinische Informatik diverse Anwendungsfelder und wird von verschiedensten Personen(-gruppen) im Gesundheitswesen genutzt. Die Deutsche Gesellschaft für Medizinische Informatik, Biometrie und Epidemiologie e.V. (GMDS) definiert die medizinische Informatik wir folgt: „Die Medizinische Informatik ist die Wissenschaft der systematischen Erschließung, Verwaltung, Aufbewahrung, Verarbeitung und Bereitstellung von Daten, Informationen und Wissen in der Medizin und im Gesundheitswesen.“ (15). Die medizinische Informatik hat dabei das Ziel zur „[…] Gestaltung der bestmöglichen Gesundheitsversorgung beizutragen.“ (15). Und zwar indem unter Anderem Ärzte sowie Patienten unterstützt und Prozesse optimiert werden und indem neues Wissen im Gesundheitswesen erschlossen wird (15). Aufgrund der besonderen der Bedeutung der Medizin, muss an Software in der Medizin also auch ein besonderer Qualitätsanspruch gestellt werden. Das zeigt sich unter anderem darin, dass Programme die einen Einsatzzweck im medizinischen Bereich haben als Medizinprodukte gelten und dementsprechend zugelassen werden müssen.

Ab 1995 regelte in Deutschland das Medizinproduktegesetz (MPG) dazu näheres (16). Seit 2017 gilt hierfür die Medical Device Regulation (MDR), also die europäische Medizinprodukte-Verordnung. Wichtige Ziele dieser Gesetze sind u.a. die Sicherheit und Leistung von Medizinprodukten zu gewährleisten und den Schutz von Anwendern und Patienten sicherzustellen, außerdem werden mit der MDR andere EU-Richtlinien zusammengeführt. Jedoch ist die Medical Device Regulation noch nicht umgesetzt wurden, denn der „Geltungsbeginn der Verordnung wurde auf den 26. Mai 2021 verschoben“ (17). Stark vereinfacht ausgedrückt gelten als Medizinprodukt gemäß den Regelungen

diverse Produkte, welche zur medizinischen Anwendung bestimmt sind. Entscheidend ist hierbei die Zweckbestimmung des Herstellers. Sollte also beispielsweise eine Software primär für die Behandlung einer Krankheit; die Erkennung einer Verletzung oder die Untersuchung eines physiologischen Vorgangs vorgesehen sein, so gilt sie als Medizinprodukt. (18, 19)

Zu unterscheiden ist noch ob es sich um Betriebssoftware (Embedded Software) oder Applikationssoftware (Stand-Alone-Software) handelt. Betriebssoftware ist Teil eines Medizinprodukts und damit als Teil des Ganzen zu sehen. Hier erfolgt die Zulassung nur gemeinsam mit dem Produkt und eine Konformitätsbewertung der Software muss nicht gesondert erfolgen. Applilationssoftware hingegen gilt als eigenständiges Medizinprodukt (wenn sie die genannte Zweckbestimmung erfüllt) und muss dementsprechend klassifiziert werden. Diese Software muss dann also alle nötigen Bestimmungen erfüllen und zertifiziert sein, bevor sie in den Handel kommt. Diese Bestimmungen wurden dann mit dem MDR weiterhin verschärft. So kam es unter Anderem zu „Konkretisierung der Anforderungen an die klinische Bewertung, […] Verschärfung der Bestimmungen über die Marktüberwachung […] und Verbesserung der Identifizierung und Rückverfolgbarkeit von Produkten durch Einführung einer eindeutigen Produktidentifizierungsnummer (Unique Device Identification, UDI)“ (17). Außerdem gibt es nun auch neue Klassifizierungsregeln für medizinische Software und insgesamt höhere Anforderungen an ein Medizinprodukt, bevor es auf den Markt kommen kann. Allerdings gab es diesbezüglich auch Kritik, da durch die verschärften Voraussetzungen höhere Kosten auf die Gesundheitssysteme zukommen. Und innovative Produkte (auch Software) vermehrt getestet werden muss und Patienten somit womöglich länger auf neue, vielversprechende Behandlungsmöglichkeiten warten müssen. (17, 19)

Software in der Medizin sollte dann vor allem eines: funktionieren. Viele Programme in Krankenhäusern laufen im 24h-Betrieb und Ärzte müssen sich auf ihre Software verlassen können, wenn sie Entscheidungen treffen. Deshalb ist Sicherheit auch eine der obersten Maxime für Entwickler von Medizinprodukten. Um das bei medizinischer Software also zu gewährleisten sind neben der Einhaltung diverser Richtlinien und Bestimmungen, auch ein gutes Projektmanagement und umfassende Tests bei der Entwicklung notwendig. (20)

2.2 Software-Tests und Testdaten

Aus den oben genannten Gründen sind Regulierungen und die damit verbunden Software-Tests wichtig um vor allem Sicherheit zu garantieren. Des Weiteren können gut durchgeführte Tests die Wartungskosten von Software senken oder auch die Qualität von Anwendungen verbessern. Und Programme von denen möglicherweise Menschenleben abhängen werden nie in Betrieb genommen, wenn sie nicht vorher streng getestet wurden und alle Normen erfüllen.

Nach Parrington und Roper versucht man beim Testen von Software im Allgemeinen Fehler im Programm zu finden und zu korrigieren und das so früh wie möglich. Denn je später ein Fehler in der Entwicklung gefunden wird, desto teurer kann er werden. Ein Fehler ist allgemein die Nichterfüllung einer Anforderung oder eine Abweichung vom geforderten Verhalten eines Systems. „Qualität hat ihren Preis" heißt es oft und das gilt auch für Software. Denn auch wenn das Testen und die Qualitätssicherung sowohl viel Zeit als auch Geld kosten, sollten wirtschaftliche Aspekte eine untergeordnete Rolle spielen, wenn es um lebenswichtige Systeme geht. (21)

Das Testen größerer Anwendungen wird in mehrere Phasen unterteilt. Wobei es je nach Entwicklungsschema verschiedene Möglichkeiten gibt die Tests zu klassifizieren bzw. zu unterscheiden. Meist unterteilt man hierbei in Teststufen, die gemäß dem Fortschritt in der Entwicklung einer Software getrennt werden (vgl. V-Modell der Softwareentwicklung). Zu nennen sind hierbei dann Modultest (Unittest), Integrationstest, Systemtest und zuletzt der Abnahmetest, bevor ein System in Betrieb genommen wird. Man testet also zumeist „Bottom-up" (von unten nach oben). (22)

So unterscheidet Witte dabei die Tests bzw. Testphasen wie folgt: Unittests überprüfen die Korrektheit einzelner Module, Klassen, oder Ähnliches. Es werden also Einzelteile auf korrekte Funktion getestet. Da diese oft noch eine geringe Komplexität haben, können diese mit vergleichsweise wenigen Testfällen umfassend geprüft werden. Ein Integrationstest überprüft dann quasi das Zusammenwirken der einzelnen voneinander abhängigen Teile, die isoliert fehlerfrei laufen. Hier werden dann Datenaustausch und Schnittstellen getestet und Testszenarien erstellt, welche das Zusammenspiel betroffener Komponenten aufzeigen. Der Systemtest „ist die Teststufe, bei der das gesamte System gegen die gesamten Anforderungen (funktionale und nicht funktionale Anforderungen) getestet wird." (22). Es werden dann Testdaten eingesetzt und es sollte eine Testumgebung genutzt werden, die der Produktivumgebung des Kunden/Nutzers ähnlich ist. Beim Systemtest wird eine Software dann auch auf ihre funktionalen und nichtfunktionalen Qualitätsmerkmale überprüft. Zu den funktionalen gehören z.B.: Interoperabilität, Richtigkeit oder Sicherheit. Nicht-funktionale Tests sind u.a.: Performancetest, Lasttest, Stresstest oder ein Test auf Stabilität. Diese 3 Teststufen dienen der Verifikation der Software (ob ein Programm auch entsprechend den Vorgaben funktioniert), während ein Abnahmetest (auch User Acceptance Test) dann die Software validieren soll (also ob sich die Software für den geplanten Einsatzzweck eignet). Der Abnahmetest ist die Überprüfung des fertigen Programms durch den Endanwender oder Auftraggeber. Hier geht es nicht darum Fehler zu finden, diese sollten bereits behoben wurden sein, sondern dem Kunden Vertrauen in die Software zu geben. Testfälle werden vom Abnehmenden bestimmt und erst wenn hier alles erfolgreich ist, ist die Softwareentwicklung abgeschlossen und ein Produkt kann eingesetzt werden. Darüber hinaus sei noch zu erwähnen, dass gesetzliche Bestimmungen oder andere Regulierungen bei der Abnahme umgesetzt werden müssen, auch ohne explizit im Vertrag zu stehen. (22)

Ein Software-Test benötigt dann neben dem Testmanagements auch Testdaten, die die späteren Eingaben in das Programm simulieren sollen. Albrecht-Zölch definiert Testdaten zum Beispiel so:

> Testdaten bezeichnen alle Daten, die in die Testumgebung eingegeben (Eingabe- und Zustandswerte für ein Testobjekt) oder aus ihr ausgelesen werden (Sollwerte, Istwerte) sowie alle Daten, die in der Testumgebung vorhanden sind bzw. verwendet werden (z.B. in einer Datenbank, aber auch Umgebungsdaten wie Konfigurationsdateien, Ports usw.). Testdaten beeinflussen die Ausführung der zu testenden Komponente bzw. des zu testenden Systems oder werden dadurch beeinflusst. (23)

Genauer kann man Testdaten dann noch in Primär- und Sekundärdaten unterteilen. Die primären Testdaten sind die Eingabedaten, also diejenigen die Verarbeitet werden sollen. Während sekundäre Testdaten die Daten bezeichnet, die in der Testumgebung vorliegen müssen (z.B. Datenbanken oder Konfigurationsdateien). (23)

In dieser Arbeit sind mit (Test-)Daten immer Primärdaten gemeint. (Jedoch können die erzeugten Daten dann für mehr als „nur“ Testzwecke genutzt werden.) Auch interessant ist der Fakt, dass ca. 30 bis 50% des Aufwands eines Software-Tests auf das Erzeugen, sowie Pflegen der Testdaten fällt (23). Testdaten lassen sich dabei nach Albrecht-Zölch auf zwei Arten erzeugen: Entweder aus realen Daten oder auf synthetischem Wege. Die Daten aus realen Erhebungen müssen dann auf Grund von Datenschutzbestimmungen anonymisiert oder zumindest pseudonymisiert werden, synthetische Daten sind dagegen künstlich erzeugt und damit frei von diesen Regulierungen. Zu den Vorteilen von anonymisierten Daten gehört zum Beispiel, dass die Daten dann trivialerweise realistisch sind. Außerdem erhält man, wenn ein angemessener Bestand an Produktionsdaten besteht (also Daten, die von z.B. realen Kunden einer schon laufenden Anwendung stammen), das in der Regel „richtige“ Volumen an Testdaten für beispielsweise Performance- oder Laufzeitmessungen. (23)

Ein Nachteil von anonymisierten Daten ist zum Beispiel der relativ hohe initiale Aufwand um die realen Daten überhaupt (sicher) zu anonymisieren. Auch kann es sein, dass nicht alle möglichen Fälle abgebildet sind, wenn Realdaten diese nicht enthalten. Das heißt Grenzfälle zu testen gestaltet sich teilweise schwierig. Darüber hinaus ist es in einigen Situationen auch schwer ausreichend anonymisierte Daten zu erhalten, da schon die benötigten Realdaten rar sind. Der Datenbestand ist manchmal aber auch größer als benötigt, wenn z.B. die gesamten Produktionsdaten in Testdaten überführt werden. Daraus folgen wiederum hoher Speicherplatzbedarf und längere Laufzeiten beim Testen. Auch wenn Speicherplatz heutzutage nicht zu größten Kostenfaktoren gehört, ist die Zeit meistens ein sehr begrenzender Faktor. Testdaten aus realen Beständen müssen wie schon erwähnt erst kopiert, anonymisiert und dann ggf. ergänzt oder bereinigt werden. Dieser hohe Managementaufwand spricht ebenfalls gegen das Verwenden von anonymisierten Echtdaten zu Testzwecken. (23)

2.3 Datenschutz und Anonymisierung

Da Anwendungssysteme in der Medizin auf sehr sensiblen und persönlichen Daten operieren, müssen solche Daten auch entsprechend geschützt werden. Aber auch in der Wirtschaft spielt Datenschutz eine immer größere Rolle. So gibt Albrecht-Zölch auf die Frage wann echte Daten als Testdaten verwendet werden können die Antwort: „Genau genommen gar nicht. […] Aus Gründen des Datenschutzes sollten überhaupt keine echten Daten in Testumgebungen verwendet werden, wenn sie sensible (personenbezogene oder personenbeziehbare) Daten enthalten. Daher müssen die aus der Produktionsumgebung kopierten Daten bearbeitet werden (z.B. anonymisiert).“ (23).

2.3.1 Datenschutz nach DSGVO und Co.

Mit dem Inkrafttreten der Datenschutz-Grundverordnung (DSGVO) und dem ergänzendem Bundesdatenschutzgesetz (BDSG) im Jahr 2018 kam es zu weitreichenden Änderungen im Datenschutz. Das maßgebliche Ziel des Datenschutzes ist es, eine Verletzung der Persönlichkeitsrechte des einzelnen zu verhindern. Vor dieser Neuerung des Datenschutzes wurde dieses Thema in Deutschland einerseits durch das „[…] grundrechtlich garantierten Recht auf informationelle Selbstbestimmung [geregelt], dessen Ausgestaltung maßgeblich durch das Bundesverfassungsgericht geprägt wurde.“ (2). Andererseits gaben auch frühere EU-Datenschutzrichtlinien den Rahmen vor. Wobei ursprünglich der Schwerpunkt auf der Abwehr staatlicher Eingriffe in die Grundrechte lag, wurde später auch der Schutz der Persönlichkeitsrechte einzelner (aus wirtschaftlichem Interesse) wichtiger. Das ist in der DSGVO und dem BDSG genauso bestärkt worden, wie die Regulierung von IT-Systemen. Es wird immer wieder die Bedeutung von datenschutzfreundlichen IT-Systemen betont, die sowohl Datenminimierung (das Sammeln von möglichst wenig Daten), als auch datenschutzfreundliche Verarbeitung gewährleisten sollen. Hierfür geben einzelne Artikel der DSGVO einen (technischen und organisatorischen) Rahmen, wie IT-Systeme funktionieren können bzw. müssen. (2)

Gerade heutzutage, wo viele Informationen online verarbeitet oder gespeichert werden und es Unternehmen gibt, die Profit durch den Verkauf persönlicher Daten fremder Menschen machen, ist Datenschutz unentbehrlich. Aber auch nur für Testzwecke eingesetzte Daten unterliegen weiterhin der DSGVO und können daher nicht uneingeschränkt genutzt werden, wenn Sie personenbezogene Daten enthalten. Um den Begriff „personenbezogene Daten“ zu konkretisieren, nutze ich die Definition aus Artikel 4 der DSGVO:

> Im Sinne dieser Verordnung bezeichnet der Ausdruck […] "personenbezogene Daten" alle Informationen, die sich auf eine identifizierte oder identifizierbare natürliche Person (im Folgenden "betroffene Person") beziehen; als identifizierbar wird eine natürliche Person angesehen, die direkt oder indirekt, insbesondere mittels Zuordnung zu einer Kennung wie einem Namen, zu einer Kennnummer, zu Standortdaten, zu einer Online-Kennung oder zu einem oder mehreren besonderen Merkmalen, die Ausdruck der physischen, physiologischen, genetischen, psychischen, wirtschaftlichen, kulturellen oder sozialen Identität dieser natürlichen Person sind, identifiziert werden kann[…]. (2)

Das zeigt, dass z.B. im Bereich der Medizin fast alle gesammelten Daten eines Patienten ‚personenbezogene Daten' gemäß der DSGVO sind. Dazu gibt es noch die weiter spezifizierten ‚Gesundheitsdaten'. Das sind dann laut Art.4 „[…] personenbezogene Daten, die sich auf die körperliche oder geistige Gesundheit einer natürlichen Person, einschließlich der Erbringung von Gesundheitsdienstleistungen, beziehen und aus denen Informationen über deren Gesundheitszustand hervorgehen […]." (2).

Laut Gesetzestext zählen diese Gesundheitsdaten (wie auch beispielsweise genetischen Daten, biometrischen Daten oder Daten zur sexuellen Orientierung einer natürlichen Person) zu besonderen personenbezogenen Daten. Deren Verarbeitung ist laut Artikel 9 Abs. 1 DSGVO untersagt. Ausnahmen finden sich in Abs. 2 des Artikels, wonach z.B. die Verarbeitung solcher Daten erlaubt ist, sofern eine ausdrückliche und freiwillige Einverständniserklärung der betreffenden Person erfolgt ist. Unter Anderem ist die Verwendung auch erlaubt: zum Schutz lebenswichtiger Interessen, wenn eine Person z.B. körperlich nicht in der Lage ist ihr Einverständnis zu erteilen oder aus Gründen des öffentlichen Interesses (im Bereich der öffentlichen Gesundheit). (2)

2.3.2 Anonymisierung, Pseudonymisierung und Deanonymisierung

Also auch und gerade in der Medizin gilt die DSGVO um den Datenschutz zu gewährleisten. Diese personenbezogenen Daten dürfen dann nur sehr eingeschränkt genutzt werden. Hier könnte die Anonymisierung oder Pseudonymisierung Abhilfe schaffen. So bezeichnet der Ausdruck Pseudonymisierung:

> […] die Verarbeitung personenbezogener Daten in einer Weise, dass die personenbezogenen Daten ohne Hinzuziehung zusätzlicher Informationen nicht mehr einer spezifischen betroffenen Person zugeordnet werden können, sofern diese zusätzlichen Informationen gesondert aufbewahrt werden und technischen und organisatorischen Maßnahmen unterliegen, die gewährleisten, dass die personenbezogenen Daten nicht einer identifizierten oder identifizierbaren natürlichen Person zugewiesen werden. (2)

Es werden hier also Daten teilweise durch ein Pseudonym ausgetauscht, damit diese (ohne Zusatzinformationen) keiner Person zugeordnet werden können. Ein Beispiel hierfür wäre das Verarbeiten von medizinischen Untersuchungen in Laboren. Die Labore arbeiten, nach Anfrage eines Arztes, nur mit Pseudonymen und nur durch die Arztpraxis kann dann das Ergebnis wieder einer eindeutigen Person zugeordnet werden (24).

Eine vollständige Anonymisierung ist dagegen die permanente Veränderung personenbezogener Daten, sodass diese keinen Personen mehr zugeordnet werden können (2, 24). Bei der Pseudonymisierung bleiben Personenbezüge also erhalten (wenn auch verborgen), während bei einer Anonymisierung der Personenbezug verloren gehen sollte. Allerdings gehen bei einer Anonymisierung mit dem Personenbezug auch einige Informationen verloren, welche für statistische Auswertungen interessant sein könnten. In §71 BDSG wird dann noch ergänzt: „[…] Personenbezogene Daten sind zum frühestmöglichen Zeitpunkt zu anonymisieren oder zu pseudonymisieren, soweit dies nach dem Verarbeitungszweck möglich ist. […]" (2). Ob dies immer so umgesetzt wird ist fraglich. Abschließend sei auch noch erwähnt, dass die DSGVO „nicht für anonyme Informationen gelten, d. h. für Informationen, die sich nicht auf eine identifizierte oder

identifizierbare natürliche Person beziehen, oder personenbezogene Daten, die in einer Weise anonymisiert worden sind, dass die betroffene Person nicht oder nicht mehr identifiziert werden kann.“ (2). Das heißt, dass die obigen Bestimmungen und Restriktionen nicht auf Datensätze angewendet werden müssen, die vollständig anonymisiert sind.

Originaldaten können dann auf verschiedenen Wegen anonymisiert bzw. pseudonymisiert werden. Auch durch das zunehmende Wachstum der Datenmengen, müssen in Zukunft „[…] verstärkt Verfahren gefunden werden, mit welchen personenbeziehbare Daten anonymisiert werden können, aber der Nutzen der Daten für Analysen erhalten bleibt“ (8). Eine Möglichkeit zum Anonymisieren eines Datensatzes ist das Verrauschen der Daten (also das zufügen von zufälligen Werten) und das Entfernen des Personenbezuges, da ohne Personenbezug die strengen Vorgaben aus dem Datenschutz entfallen. Dieser Vorgang reduziert jedoch die Datenqualität und die statistische Aussagekraft. Außerdem muss sichergestellt sein, dass es nicht möglich ist eine Person aus dem Datensatz zu identifizieren. Zum Anonymisieren von Daten gibt es mittlerweile einige Tools (auch open-source Software) die je nach Anwendungszweck hohen Datenschutz bieten und eine Auswertung der Daten zulassen. Ein Beispiel wäre das ARX Data Anonymization Tool (25), von der Charité Berlin entwickelt „zur Anonymisierung von Patienten- oder Probandendaten“ (25). Auch Datenbankenmanagementsysteme bieten bereits Module zur Anonymisierung an, bei denen die Daten direkt maskiert werden.

Eine andere Möglichkeit zum Anonymisieren von Daten ist die Synthetisierung. Drechsler beschreibt dies wie folgt: „Bei der Daten-Synthetisierung handelt es sich um eine Methode, mit der eine ‚künstliche‘ Repräsentation eines Originaldatensatzes erstellt werden kann. Hierzu wird ein Modell entwickelt, das die Originaldaten so gut wie möglich erklärt. Aus diesem Modell werden neue Daten generiert, die wichtige statistische Eigenschaften des Originaldatensatzes erhalten. Der synthetische Datensatz besteht nicht aus Daten natürlicher Personen, sondern aus Daten synthetischer Einheiten.“ (8). Solche Methoden sind teilweise sehr aufwendig und kompliziert, sie lohnen sich also nicht für jeden Anwendungszweck.

Als Re-Identifizierung (oder auch Deanonymisierung) bezeichnet man dann die Aufhebung der durchgeführten Anonymisierung. Dabei wird meist versucht, die vorhandenen Daten so zu kombinieren, dass diese mit einer sehr großen Wahrscheinlichkeit nur noch auf eine Person zutreffen. Je mehr zusätzliche Daten gesammelt werden können, desto wahrscheinlicher ist eine Deanonymisierung. Eine vollständige und „sichere“ Anonymisierung ist daher schwer zu erreichen, zumindest wenn eine gewisse Aussagekraft des Datensatzes erhalten bleiben soll. Wie oben erwähnt wurde findet die DSGVO auf anonyme Daten keine Anwendung. Ist aber eine Re-Identifikation möglich, beispielsweise unter Zuhilfenahme zusätzlicher Informationen, so findet dann möglicherweise doch eine Verletzung des Datenschutzes statt. Der Gesetzestext ist dabei nicht eindeutig. Immer wieder werden Ausnahmen benannt, falls Bestimmungen nur „mit einem unverhältnismäßig großen Aufwand“ (2) umzusetzen sind.

Auch wenn solche Formulierungen recht schwammig sind, kann es zu rechtlichen Konsequenzen kommen falls ein Teil eines anonymisierten Datensatzes deanonymisiert werden kann. Und dies geschieht in der Realität häufiger als erwartet.

Ein Beispiel dafür ist ein Fall aus 2006, in dem (scheinbar) anonymisierte Daten von Netflix veröffentlicht wurden sind und wenig später deanonymisiert werden konnten. Das als Online-Videothek etablierte Unternehmen hatte im Rahmen einer Ausschreibung 100.480.507 Datensätze in anonymisierter Form veröffentlicht. Hauptsächlich waren das die Bewertungen von Nutzern zu bestimmten Filmen. Anonymisiert hieß in dem Fall, dass jeder Bezug zum Nutzer hinter einer Bewertung entfernt wurden ist. (26)

Narayanan et al. untersuchten dann in ihrer Arbeit „Robust De-anonymization of Large Spare Datasets“ (27) ob und wie die veröffentlichten Daten genutzt werden können, um Nutzer zu re-identifizieren. Zuerst haben die Autoren den Datensatz von Netflix analysiert und anschließend zusätzliche Informationen (in Form von Filmbewertungen) von öffentlichen Datenbanken, vor allem aus der IMDB (Internet Movie DataBase) benutzt. Ohne näher auf den Algorithmus eingehen zu wollen, wurde es durch die zusätzlichen Informationen ermöglicht einen Großteil des Datensatzes zu deanonymisieren. „Die Schlussfolgerung der Autoren ergab, dass schon mit acht Bewertungen, von denen sogar zwei falsch sein dürfen, und Bewertungszeiträumen die bis zu 14 Tage auseinanderliegen, 99% der Datensätze eindeutig identifiziert werden können. Für 68% reichen bereits zwei Bewertungen mit einer Abweichung von weniger als drei Tagen.“ (26). Das Ergebnis zeigte, dass auch mit wenigen zusätzlichen Informationen und vergleichsweise geringem Aufwand, Datensätze deanonymisiert werden können und dadurch die Privatsphäre von Nutzern kompromittiert wird. Auch wenn Datensätze also scheinbar anonymisiert veröffentlicht werden, müssen sie nicht zwangsläufig anonym bleiben. (26, 27)

2.4 Synthetische Daten

Die Nachteile bzw. Schwierigkeiten beim Testen oder Arbeiten mit anonymisierten Daten lassen synthetische Testdaten noch attraktiver erscheinen. Diese sind nämlich frei von jeglichen Datenschutz Richtlinien und benötigen keine zusätzliche Bearbeitung um verwendet werden zu dürfen. Solche synthetischen Daten sind maschinell generiert und stammen daher nicht aus realen Ereignissen. Sie können allerdings je nach Ansatz eine große Ähnlichkeit zu echten Personen- bzw. Patientendaten aufweisen. Mit synthetischen Daten können dann auch z.B. Schnittstellen oder Modelle in Grenzbereichen ausgetestet werden, die in der Realität nur selten entstehen. Es können Beschränkungen in der Nutzung von Datensätzen umgangen werden und Entwickler, sowie Anwender, sind durch synthetischen Daten flexibler in der Daten-Nutzung und dem Datenaustausch. (5, 7)

Synthetische Daten sind also im Allgemeinen künstlich generierte Daten, die Originaldaten bestmöglich nachstellen sollen und syntaktisch korrekt vorliegen müssen. Semantisch bzw. inhaltlich können solche Daten ebenfalls sehr nah an der Realität liegen oder aber große Abweichungen haben können, was mitunter auch gewollt ist. Eingesetzt werden sie an mehreren Stellen der Software-Entwicklung und den unterschiedlichsten Domänen. Für das Erzeugen von allgemeinen Testdaten (für beispielsweise Datenbanken) gibt es mittlerweile einige Tools. Diese sind jedoch nicht für alle Anwendungszwecke geeignet. Mitunter werden sich aber auch einfach plausible Daten ausgedacht, also frei erfunden. Solche Daten stellen auch synthetische Daten dar, auf diese Weise können aber keine größeren Datensätze sinnvoll erstellt werden.

Auch in der Medizin kommt es dann zum Einsatz von Testdaten, wie z.B. zur Softwareentwicklung oder Demonstrationen. Großer Bedarf besteht auch bei den Patientendaten, die im Original logischerweise geschützt sind. Gerade auf diesem Gebiet gibt es vergleichsweise wenig Arbeiten, die sich mit der Erzeugung synthetischer Patientendaten auseinandersetzen. So schreiben Dube und Gallagher in einem ihrer Paper:

> The problem of generating synthetic data has been widely investigated in many domains. Despite the chronic limitations on access to the EHR for secondary use due to privacy concerns, there are very few research efforts to-date directed on developing promising and low cost approaches to generating synthetic EHRs for secondary uses. Only about 5 major works have appeared in literature during the past 12 years compared to more than 60 works in other domains. (28)

Patientendaten bzw. die entsprechenden Gesundheitsakten werden im englischen auch als *electronic health record* (EHR) bezeichnet und gespeichert. Darunter werden alle Befunde, Laborergebnisse, Behandlungen, Beurteilungen, usw. bezüglich eines Patienten dokumentiert. EHRs sind quasi die digitalisierte Gesundheitshistorie der Patienten und ermöglichen eine effizientere Arbeit und Behandlung. Diese elektronischen Gesundheitsakten sind für autorisierte Personen (in der Regel die behandelnden Ärzte, Labore oder Ähnliches) zugänglich, unterliegen aber neben einer ärztlichen Schweigepflicht auch dem Datenschutz und können somit nicht einfach weitergegeben werden. (28, 29)

Gerade für die Entwicklung von Software, beispielsweise zur Patientenverwaltung, wären aber diese Daten interessant und hilfreich. Daher gibt es unter anderem *realistic synthetic EHRs* (RS-EHRs), die die benötigten elektronischen Patientenakten simulieren sollen (5). RS-EHRs sind synthetische Patientendaten bzw. künstliche Gesundheitsakten, welche in der Regel maschinell generiert werden. Sie sind Gegenstand einiger Forschungsarbeiten und können wie bereits erwähnt große Vorteile in der Softwareentwicklung bringen. Ein Beispiel für eine Software die RS-EHR erzeugen kann ist das Programm Synthea (5, 11) worauf ich später genauer eingehen werden (Kapitel 2.6).

Synthetische Daten können dann, je nach Anwendungszweck, auf verschiedenen Wegen erzeugt werden. Man kann zum Beispiel zwischen synthetisierten Daten und „simulierten" synthetischen Daten unterscheiden. Synthetisieren ist das erzeugen künstlicher Datensätze, ausgehend von realen Daten und daraus gewonnenen Modellen.

Das bedeutet, dass Originaldaten in eine synthetische Repräsentation überführt werden, die die Originaldaten bestmöglich widerspiegelt und wesentliche statistische Eigenschaften erhält (8, 9). Solche Verfahren sind Datenbasiert und analysieren Produktivdaten um daraus einen vergleichbaren Datensatz erzeugen zu können. Diese Methoden können zum Beispiel auf der multiplen Imputation basieren, einem mathematisch-statistischen Verfahren zur Ersetzung fehlender Werte in Datensätzen, bei welchem diese durch mehrere plausiblen Werte ersetzt werden (9). Andere Ansätze basieren z.B. auf künstlichen neuronalen Netzen oder probabilistischen Modellen. Hier benötigt man jedoch immer noch Echtdaten, die entsprechend sorgfältig behandelt werden müssen. Diese Ansätze werden mit Hilfe von Machine Learning durchgeführt und sind daher aufwendig, da komplexe Modelle trainiert und zum Teil überwacht werden müssen (30). Dabei werden zum Trainieren oder zur Validierung auch echte Patientendaten gebraucht, was wieder andere Probleme mit sich bringt. Das Prinzip ist immer recht ähnlich: es wird eine Reihe von realen EHR-Daten genommen, damit ein Modell angefertigt und angepasst, um somit dann neue synthetische EHR-Daten aus dem erlernten Modell zu generieren (30). Durch das Lernen aus echten EHR-Stichproben wird erwartet, dass das Modell relevante statistische Eigenschaften der Daten extrahieren kann (30).

Andere Ansätze erzeugen vollständig synthetische Daten rein algorithmisch. Wie z.B. PADARSER („Publicly Available Data Approach to the Realistic Synthetic EHR"), bei dem die Autoren bereits veröffentlichte Daten und Informationen ausnutzen und daraus synthetische Datensätze (RS-EHRs) generieren (28). Es werden hierbei öffentlich verfügbare Daten wie Statistiken zur Gesundheit, klinische Richtlinien und medizinischer Kodierungsstandards gesammelt und daraus schlussendlich synthetische Patientendatensätze erzeugt. Aus den öffentlichen Datensätzen werden zuerst verschiedene Daten extrapoliert und diese dann in einem weiteren Schritt algorithmisch zu RS-EHRs verarbeitet. Eine Darstellung davon ist in Abbildung 1 zu sehen.

Der Algorithmus, welcher in dem Paper von Dube und Gallagher vorgestellt wird, läuft grob wie folgt ab: Zuerst werden „Patienten" gemäß den vorhandenen Statistiken erzeugt und demographisch verteilt. Jedem synthetischen Patienten wird nun iterativ eine Krankheit „injiziert", die aus öffentlich verfügbaren Daten extrapoliert wurde. Jede Krankheit ist mit einer bestimmten Richtlinie bzw. einem Protokoll verbunden, aus dem der klinische Arbeitsablauf zur Behandlung der Krankheit abgeleitet wurde. Der Ablauf bietet dann einen realistischen Kontext für klinische Ereignisse, die die Grundlage für die Generierung synthetischer Einträge in den RS-EHRs bilden. Für jeden künstlichen Eintrag werden des Weiteren geeignete Codes aus dem standardisierten medizinischen Codierungssystem beigefügt. Die RS-EHRs werden hierbei ohne Zugang zu echten EHR, also ohne originale personenbezogene Daten, gewonnen. (28)

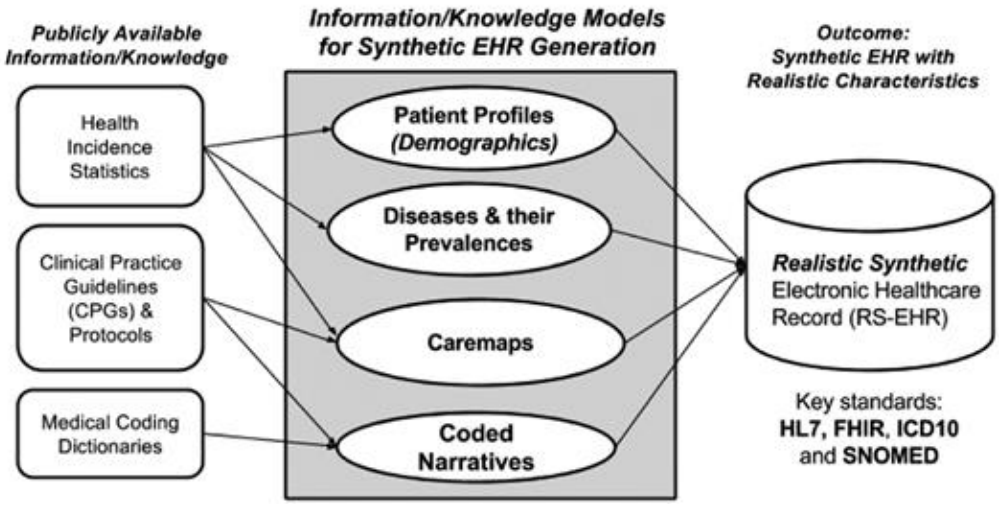


Abbildung 1: Schematische Darstellung des PADARSER-Framework (5)

Jedoch wird in der eben erwähnten Arbeit lediglich der PADARSER-Ansatz vorgestellt und der zu Grunde liegende Algorithmus (zu einem gewissen Level) erklärt. Eine Software zum Anwenden, mit der man selbst realistische synthetische Patientendaten generieren kann, wurde ursprünglich nicht veröffentlicht. Es gibt noch andere arbeiten, die sich mit dem generieren von synthetischen Daten im medizinischen Umfeld beschäftigen. Jedoch liegt bei den wenigsten davon das Hauptaugenmerk auf den, auch in Deutschland, benötigten (realistischen und synthetischen) Patientendaten. Weitere Ansätze sind dabei auch oft nicht genau genug beschrieben oder eine praktische Umsetzung der Algorithmen ist aktuell nicht frei verfügbar. Ein weiterer und vielversprechender Ansatz zur Erzeugung von synthetischer Patientendaten ist Synthea. Hier wird neben einem theoretischen Konzept auch ein Programm veröffentlicht, welches RE-EHRs algorithmisch erzeugen kann in dem das „Leben" von künstlichen Patienten simuliert wird (ebenfalls auf Basis von öffentlichen Statistiken und ohne das Originaldaten benötigt werden). Darauf gehe ich unter Kapitel 2.6 gesondert ein.

2.5 Medizinische Standards und Code-Systeme

Im medizinischen Umfeld kommt mittlerweile eine große Auswahl von verschiedensten Programmen und Anwendungen zum Einsatz. Von Bildgebungsverfahren oder Labor Untersuchungen bis hin zu statistischen Auswertungen werden jede Menge Daten in der Medizin erzeugt und gespeichert. Dies alles wird dokumentiert und in einer (meist elektronischen) Patientenakte gesammelt. Medizinische Dokumentation besteht im Allgemeinen aus dem Erfassen, Gewinnen, Speichern und vor allem Wiedergeben von medizinischen Informationen und spielt eine wichtige Rolle in der Krankenversorgung (18). Da dafür und anderweitig, allein innerhalb eines Krankenhauses, verschiedene Hard- und Software eingesetzt wird und diese meistens nicht alle vom selben Hersteller sind, könnten Daten in verschiedenen Formaten erzeugt werden. In der Regel läuft

jede IT-gestützten Behandlung über das Krankenhausinformationssystem (KIS) ab. Welches das „sozio-technische System aller Informationsverarbeitung, -übermittlung und -speicherung im Krankenhaus“ (18) ist.

Alle erzeugten Daten fließen dort ein und können daraus abgerufen werden. Besonders umfangreichere Daten bzw. Dateien können dann zusätzlich in einem sogenannten PACS (Picture Archiving and Communication System) gespeichert werden. Den Datentransfer zwischen unterschiedlichen Systemen dann über bilaterale Schnittstellen zu lösen, ist umständlich und erzeugt unnötigerweise zusätzliche Hindernisse. Unter anderem um die Effizienz der Patientenverwaltung zu steigern (und damit kosten einzusparen) wurden Kommunikationsstandards für das Gesundheitswesen entwickelt. (18, 31)

Um das Zusammenwirken medizinischer Geräte verschiedener Hersteller zu garantieren gibt es Normen, Standards und Ordnungssysteme, die für Interoperabilität zwischen den Anwendungen sorgen und dafür, dass alle Informationen richtig und einheitlich interpretiert werden können. Normen und Standards dienen der Sicherheit, senken den Aufwand der Qualitätssicherung und bieten Anwendern den Vorteil der Interoperabilität. Sie werden durch verschiedene Stellen auf verschiedenen Ebenen herausgegeben und überwacht. In Deutschland unter anderem durch das Deutsche Institut für Normung (DIN), auf europäischer Ebene vom Europäischen Komitee für Normung (CEN) und auf internationaler Ebene zum Beispiel von der Internationalen Organisation für Normung (ISO) oder andere Stellen (je nach Domäne). (32)

2.5.1 HL7

Standards in der Medizin sorgen also für Interoperabilität, korrekten Datenaustausch und steigern die Effizienz in medizinischen Einrichtungen. Eine Reihe wichtiger Kommunikations-Standards werden dabei von Health Level Seven (HL7) geliefert. HL7 ist eine Ansammlung von Standards im Gesundheitswesen, die von der non-profit Organisation HL7 International herausgegeben wurde. Auf der Seite des HL7 Deutschland e.V. werden die Standards wie folgt beschrieben:

> Um die zwischen den Systemen notwendigen Schnittstellen zu standardisieren, steht mit HL7 eine wertvolle Hilfe bei der Implementierung zur Verfügung. HL7 wurde in den USA entwickelt und ist dort seit langer Zeit ein offizieller Standard. […] HL7 spezifiziert Kommunikationsinhalte und Austauschformate auf der Anwendungsebene. Im Schichtenmodell der Kommunikation zwischen offenen Systemen ist diese Ebene die siebte, was zum Namen HL7 geführt hat. […] Der speziell für das Gesundheitswesen entwickelte Kommunikationsstandard HL7 ermöglicht die Kommunikation zwischen nahezu allen Institutionen und Bereichen des Gesundheitswesens. Mit HL7 lassen sich alle wesentlichen Kommunikationsaufgaben abwickeln und die Effizienz der Kommunikationsvorgänge entscheidend verbessern. (33).

Es liegen also mehrere Standards vor, die zum Beispiel die Kommunikation zwischen Systemen innerhalb eines Krankenhauses regeln oder auch allgemein Struktur und Inhalt medizinischer Dokumente festlegen. So wird HL7 zum Beispiel auch für die Übermittlung patientenbezogener Nachrichten im Krankenhausinformationssystem (KIS)

genutzt, denn „Im Standard-Workflow des klinischen Bereichs basieren die Schnittstellen zwischen KIS und Fachabteilungssystem auf HL7." (31).

Im Allgemeinen funktioniert dies folgendermaßen: „HL7 beschreibt, zu welchen Ereignissen Nachrichten mit welchem Aufbau zwischen Anwendungsbausteinen im Gesundheitswesen […] ausgetauscht werden. HL7 geht davon aus, dass in einem Anwendungsbaustein A auf Grund des Eintretens eines Ereignisses eine Nachricht an einen anderen Anwendungsbaustein B gesendet wird […]." (18). Darüber hinaus gilt: „Der für die Nachricht verwendete Nachrichtentyp hängt vom Typ des eingetretenen Ereignisses ab, beschreibt den Aufbau der versendeten Nachricht und legt die Bedeutung der einzelnen Teile der Nachricht fest. Nach dem Eintreffen der Nachricht bestätigt der Anwendungsbaustein B den Erhalt der Nachricht durch eine Bestätigung, die an den Anwendungsbaustein A zurückgesendet wird." (18). Software Hersteller können dann zu ihren Produkten passende HL7-Schnittstellen anbieten und die Software somit einsatzfähig für die meisten KIS machen.

Ein wichtiger Teil der HL7 Standard-Familie ist dabei FHIR (Fast Healthcare Interoperable Resources). Dieser wird wie folgt beschrieben: „Der Standard unterstützt den Datenaustausch zwischen Softwaresystemen im Gesundheitswesen. Er vereinigt die Vorteile der etablierten HL7-Standard-Produktlinien Version 2, Version 3 und CDA mit jenen aktueller Web-Standards und legt einen starken Fokus auf eine einfache Implementierbarkeit." (34). So wurde FHIR entwickelt um z.B. den Datenaustausch zu vereinfachen und Interoperabilität schneller erreichen zu können. Auch soll FHIR neuere Trends unterstützen und es wird im Allgemeinen mehr Wert auf Mobilität und Transparenz gelegt. Daher hat man sich für REST-Architekturen und bekannte Technologien (z.B. XML, JSON, HTTPS, …) als Grundlage von HL7 FHIR entschieden. (34)

Der FHIR Standard besteht nach HL7 aus 3 Bestandteilen: Ressourcen, Referenzen und Profilen. Ressourcen sind „[…] kompakte, logisch diskrete Einheiten des Datenaustausches mit einem wohldefiniertem Verhalten und eindeutiger Semantik. Sie sind die kleinste Einheit der Übermittlung. Die 150 spezifizierten Resourcen [sic] decken das gesamte Spektrum des Gesundheitswesens ab." (34). Beispiele hierfür sind Patient, Practitioner oder Medication. Wobei diese Ressourcen dann strukturierte Daten enthalten, die laut HL7 „80% der üblichen Einsatzszenarien" (34) abdecken. Außerdem können sie noch aus textuellen Zusammenfassungen des Inhalts bestehen oder sogenannten Extensions, also Erweiterungen die zusätzliche Anwendungsfälle abdecken. Referenzen sind dann Verweise von einer Ressource auf eine andere, mit Hilfe von Links. „Dadurch verknüpfen sich die Informationseinheiten zu einem Netzwerk, das beispielsweise eine Medikamenten-Verordnung, einen Labor-Befund oder gar vollständige Patientenakte abbilden kann." (34). Profile geben dann an wie einzelne Systeme die FHIR Ressourcen nutzen, da diese beliebig kombiniert werden kann. Sie bilden „[…] das Regelwerk für die Definition eines Services. Sie erklären, welche Resourcen [sic] und Extensions ein System kommunizieren und speichern kann." (34). Solche Profile werden in der Regel von HL7 selbst definiert, können aber auch von anderen lokalen Benutzergruppen (z.B. HL7 Deutschland e.V.) angepasst werden. Außerdem können Unternehmen oder andere Organisationen sich eigene Profile für ihre Verwendungszwecke erstellen. (33, 34)

2.5.2 SNOMED

Damit neben dem technischen Austausch von Daten auch der Informationsaustausch funktioniert und es zu keinen Fehlinterpretationen kommt, gibt es in der Medizin Terminologie-Standards. Dadurch werden Begriffe eindeutig definiert und sind für alle einheitlich zu interpretieren, auch wenn sie sprachlich unterschiedlich beschrieben werden können.

Dabei ist als ein Standard, welcher internationale Anerkennung gefunden hat, die Systematisierte Nomenklatur der Medizin (SNOMED) zu nennen. Der ursprünglich als Nomenklatur entwickelte Terminologie-Standard dient „zur Indexierung medizinischer Sachverhalte“ (18). SNOMED gab bzw. gibt es in mehreren Versionen: die erste davon wurde 1975 in den USA vorgestellt, als Weiterentwicklung einer Nomenklatur der Pathologie. Wenig später entstand dann die Überarbeitete Fassung SNOMED II, welche dann auch zu einer deutschsprachigen Fassung führte. Ende der 90er Jahre kam dann die dritte und vorerst letzte Version (SNOMED III) heraus, die über 150.000 Begriffe der Medizin enthält. Daraus abgeleitet entstanden später SNOMED RT (related terms) mit über 200.000 Begriffen und SNOMED CT (clinical terms), welche als Ordnungssystem circa 300.000 „Konzepte“ besitzt und diese in Relation setzt und eindeutig kodiert. (18, 35)

SNOMED CT gibt dabei jedem Konzept einen eindeutigen Namen, einen Kode und eine textuelle Beschreibung bzw. Definition. Außerdem beinhaltet es neben den Konzepten auch noch so genannte Beschreibungen, diese sind „[…] die fachsprachlichen Terme, von denen einer oder mehrere je einem SNOMED CT-Konzept zugeordnet sind. Davon gibt es ca. 800.000.“ (35). SNOMED CT kodiert dabei Befunde, Verfahren, Eingriffe, Körperstrukturen, Substanzen oder biologische Erzeugnisse, etc.

Als Beispiel nennt Schulz in seiner Arbeit aus 2011 das Konzept 82272006, welches durch die Beschreibungen „Acute coryza", „Acute nasal catarrh", „Acute rhinitis" und „Common cold" charakterisiert wird. Außerdem sind Konzepte in SNOMED CT über circa 1.360.000 semantische Relationen verbunden, welche sowohl Oberbegriffsrelation abbilden als auch Definitionen. So steht z.B. 82272006 ("Common cold") in einer is-a-Beziehung mit 281794004 ("Viral upper respiratory tract infection"). Im Gegensatz zur International Statistical Classification of Diseases and Related Health Problems (ICD) beispielsweise, sind in SNOMED CT auch Überlappung von Klassen erlaubt und vorhanden. Außerdem verfolgt das SNOMED CT entwickelnde IHTSDO (International Health Terminology Standards Development Organisation) die Stärkung der Interoperabilität mit anderen semantischen Standards der Medizin, wie z.B. mit ICD-10 bzw. ICD-11 oder mit HL7 um „Lücken und Überschneidungen zwischen HL7- und IHTSDO-Standards zu beseitigen[…]“ (35). SNOMED CT kann als Terminologie System also klinische Sachverhalte exakt und sprachunabhängig kodieren und könnte so semantische Interoperabilität stärken. (35)

Auch die deutsche Medizininformatik-Initiative (MII) weist auf die Wichtigkeit einer solchen Nomenklatur hin. Denn „[…] nicht nur zum „Lückenschließen“ zwischen bestehenden Codiersystemen (Katalogen) und zur automatisierten „Übersetzung“ bereits

strukturiert vorliegender Information benötigt man SNOMED CT. Möchte man Freitexte wie z.B. Arztbriefe automatisiert analysieren, ist die treffende „Übersetzung" in computerlesbare Einheiten durch SNOMED CT gewährleistet." (36). Daher wurde 2020 im Zusammenhang mit der Initiative eine deutsche Lizenz erworben. (36)

2.5.3 LOINC

Ein weiterer wichtiger Standard ist LOINC (Logical Observation Identifiers Names and Codes). Dieses Code-System verschlüsselt insbesondere Labornachrichten zum eindeutigen versenden von Untersuchungsergebnissen. Dabei finden sich Angaben zu Gegenstand und Material einer Untersuchung, sowie zur Methode selbst. LOINC wurde in den USA vom Regenstrief Institute entwickelt und etabliert, mittlerweile gibt es aber auch (ca. 11.000) deutschsprachige LOINC-Bezeichner zur Verfügung. Das Deutsches Institut für Medizinische Dokumentation und Information (DIMDI) sowie das Bundesinstitut für Arzneimittel und Medizinprodukte (BfArM) fördern aktuell die Einführung des Kodierungssystems in Deutschland: „So werden Doppelarbeiten und Inkompatibilitäten mit anderen Standards vermieden und ein öffentlicher und sicherer Zugang gewährleistet. Das BfArM koordinierte auch die Qualitätssicherung und Zusammenfassung der deutschen Übersetzungen[…]" (37). Die LOINC Codes werden in der Datenbank RELMA (Regenstrief LOINC Mapping Assistant) bereitgestellt und jährlich aktualisiert. (18, 37)

Laut DIMDI ist LOINC „[…] eine Nomenklatur, eine Sammlung von universellen, weltweit eindeutigen Identifikatoren für medizinische Untersuchungsergebnisse. Ursprünglich vom Regenstrief Institute für Laboruntersuchungen entwickelt, wurde LOINC auf klinische und medizinisch-technische Untersuchungen, medizinische Dokumententypen und andere medizinische Parameter ausgeweitet." (37). LOINC löst dabei das Problem proprietären Codes bzw. Verschlüsselungen wie sie in einigen Laborinformationssystemen vorkommen. So fördert LOINC durch „[…] standardisierte Bezeichnungen den Austausch und das Zusammenführen von Untersuchungsergebnissen und kann für den Datenaustausch in Labor, Klinik und Praxis eingesetzt werden." (37). Die Datenbank besteht dabei aus zwei Teilen: einen Teil für Laboruntersuchungen (enthält unter anderem Kategorien der klinischen Chemie, Mikrobiologie, Toxikologie oder auch für Medikamente, Drogen, Allergene, etc.). Beispielsweise wird Kalium hier mit dem Code 2828-2 oder Cholesterin in Low Density Lipoprotein (LDL) mit 2089-1 kodiert. Und es gibt einen zweiten Teil für klinischen Untersuchungen, mit z.B. Einträgen zu Vitalzeichen, EKG, Ultraschall und anderer Bildgebung oder endoskopischen Untersuchungen. Außerdem wird die zur LOINC-Datenbank zugehörige Dokumentation vom Regenstrief Institute ebenfalls bereitgestellt. (37)

2.5.4 DICOM

Ein anderer wichtiger Kommunikationsstandard im Gesundheitswesen ist DICOM (Digital Imaging and Communication in Medicine). Der Standard stammt aus dem Bereich der Radiologie und ist spezialisiert auf die Übertragung, Verarbeitung und Speicherung von digitalen Bildern. Basierend auf dem Client-Server-Prinzip dient DICOM hauptsächlich der Kommunikation zwischen PACSs (Picture Archiving and Communication Systems) und Radiologie- bzw. Krankenhausinformationssystem oder zwischen dem KIS

und anderen Fachabteilungen eines Krankenhauses. Aufgrund der zunehmenden Bedeutung der Telemedizin wird DICOM aber auch zur Kommunikation und Bildaustausch zwischen verschiedenen Einrichtungen eingesetzt. (18, 31)

DICOM beinhaltet dabei Strukturinformationen über den Inhalt der Daten, Anweisungen wie mit den Daten verfahren werden soll und Protokolle für die Datenübertragung (18). Der Standard ist dabei in 15 Teile unterteilt wie z.B. „Media Storage & File Format" (definiert das Speichern von Bildformaten auf versch. Datenträgern und das Dateiformat, mit dem DICOM-Objekte verpackt werden) oder „Data Dictionary" (Listet alle gültigen Datenelemente auf und legt deren Kennzeichnung fest) (18). Es sei auch gesagt das DICOM nicht nur zum Senden der Bilder genutzt wird, sondern auch die Daten des zugrunde liegenden Auftrages übermittelt. Und „Im Gegensatz zu HL7 definiert DICOM nicht nur ein Nachrichtenformat, sondern koppelt dies auch eng an Austauschformate." (18).

2.5.5 ICD

Die ICD (International Statistical Classification of Diseases and Related Health Problems), zu Deutsch die Internationale statistische Klassifikation der Krankheiten und verwandter Gesundheitsprobleme, ist eine der bekanntesten Diagnoseklassifikation in der Medizin. Sie ist ein weltweit anerkanntes Ordnungssystem und wird heutzutage von der Weltgesundheitsorganisation WHO herausgegeben. Sie wird regelmäßig aktualisiert und damit erweitert, aktuell ist die 10. Revision im Einsatz (ICD-10). Außerdem findet eine zunehmende Weiterentwicklung zu Dokumentations- und Abrechnungszwecken, sowie für Statistiken seit 1948 statt (18). Wobei es auch regionale bzw. länderspezifische Ergänzungen gibt, wie z.B. die 10. Revision, German Modification (ICD-10-GM) in Deutschland. Sie ist „die amtliche Klassifikation zur Verschlüsselung von Diagnosen in der ambulanten und stationären Versorgung in Deutschland." (38). In den USA ist seit längerem die ICD-9-CM (clinical modification) üblich, da sie auf die dortigen klinischen Bedürfnisse angepasst wurde (18). Ab 2022 soll dann die nächste Version benutzt werden, so schreibt das dafür hierzulande zuständige DIMDI: „Die ICD-11 soll am 1. Januar 2022 in Kraft treten; erst nach einer flexiblen Übergangszeit von 5 Jahren sollen Todesursachen ausschließlich mit der ICD-11 kodiert werden. Über den konkreten Zeitpunkt einer Einführung der ICD-11 in Deutschland sind noch keine Aussagen möglich." (39).

Die ICD (10. Revision) ist im Allgemeinen wie folgt aufgebaut: Sie unterteilt sich hierarchisch in 21 Krankheitskapitel, wie z.B. „Kapitel X: Krankheiten des Atmungssystems". Darunter gliedert sie sich in insgesamt 261 Krankheitsgruppen (z.B. „J40-J47: Chronische Krankheiten der unteren Atemwege"), welche sich in 2.037 dreistellige Krankheitsklassen aufteilen (z.B. „J45 Asthma bronchiale"). Diese können sich dann wiederum in insgesamt 12.161 vierstellige Krankheitsklassen (Unterkategorien) unterteilen, wie z.B. „J45.1 Nichtallergisches Asthma bronchiale". So werden sämtliche Krankheiten systematisch klassifiziert und hierarchisch geordnet. Wobei die Bereiche „U00-U49" für spätere Erweiterungen reserviert sind und „U50-U99" für Forschungszwecke. (18)

2.6 Synthea

Wie bereits erwähnt ist Synthea ein vielversprechender Ansatz zur Generierung von synthetischer Patientendaten. Synthea steht unter Copyright der MITRE Corporation, der Quellcode aber ist online (mit der Apache-Lizenz 2.0) frei verfügbar (40). Hier wird neben einem theoretischen Konzept auch ein Programm veröffentlicht, welches simulierte elektronische Patientenakten algorithmisch erzeugen kann. Dies geschieht in dem das „Leben" von künstlichen Patienten simuliert wird. Der Generator erzeugt syntaktisch und strukturell plausible Patientendatensätze, z.B. in Form von FHIR Ressourcen und ist online gut dokumentiert. Solch ein Generator ist flexibel und kann in verschiedenen Anwendungsfällen genutzt werden. Die resultierenden (Patienten-)Daten sind frei von Kosten-, Datenschutz- und Sicherheitsbeschränkungen und können u.a. im HL7-Standard FHIR ausgegeben werden. Sie können dann ohne Einschränkung eingesetzt werden. Hier möchte ich also kurz auf die Funktionen von Synthea eingehen, weil mit einer Software wie dieser das eingangs erwähnte Problem gelöst werden könnte.

Das 2017 von Walonoski et al. vorgestellte Synthea (5) liefert eine Methode und ein System für die RS-EHR-Generierung. Es garantiert die vollständig synthetische Ausgabe von Patientendaten, indem nur öffentlich verfügbare Informationen und Gesundheitsstatistiken als Eingaben genutzt werden. Diese Methode erzeugt Daten basierend auf Modellen des klinischen Arbeitsablaufs und verschiedener Krankheitsverläufe, welche bei Bedarf leicht verändert und verfeinert werden können. Außerdem wird hierbei das gesamte Leben eines Patienten abdeckt, anstatt sich auf ein Gesundheitsproblem oder eine Krankheit zu konzentrieren. Mit Synthea können die zu erzeugenden Daten auch nach Bedarf skaliert werden, je nach Einsatzzweck. (5)

Das Framework für den von Synthea genutzten Prozess zur Erzeugung synthetischer Daten basiert auf PADARSER, wobei ausschließlich öffentlich verfügbare Daten (aus den USA) genutzt wurden. Das PADARSER-Framework greift, wie bereits erwähnt, auf öffentlich verfügbare Datensätze zu und kombiniert diese um die synthetische EHR zu erzeugen. Abbildung 2 zeigt das PADARSER-Framework wie es in Synthea realisiert wurde. Der Realismus wird durch regionale Datensätze und das Nutzen klinischer Arbeitsabläufe verbessert. Die resultierenden synthetischen Patientendaten sind so für viele Verwendungszwecke verfügbar, ohne die mit der re-Identifizierung anonymisierter Daten verbundenen Gefahren. Zu Beginn konnten die 10 wichtigsten Gründe, warum Patienten ihren Hausarzt aufsuchen und die 10 wichtigsten chronischen Erkrankungen in den USA simuliert werden. Mittlerweile gibt es über 90 Module und dadurch können natürlich weitaus mehr Krankheiten generiert werden. Über eine Million Patientenakten sind so in Standardformaten wie HL7-FHIR oder C-CDA (Consolidated-Clinical Document Architecture) frei verfügbar. (5)

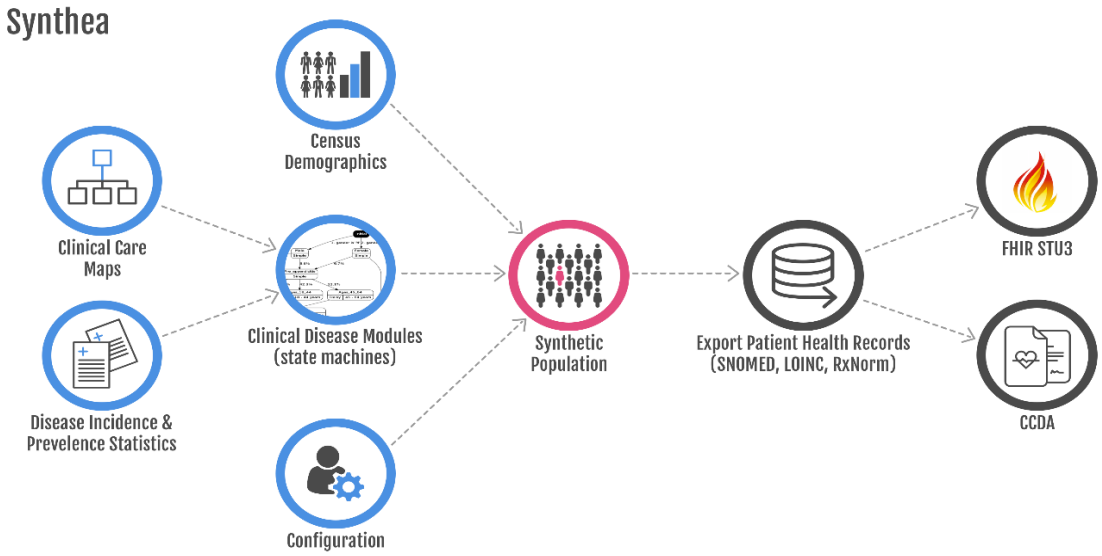


Abbildung 2: Syntheas allgemeine funktionsweise (41)

Mithilfe von Statistiken werden also Modelle für das Fortschreiten und die Behandlung von Krankheiten in einem Modul-Framework erstellt, das diese Modelle als Zustands-übergangsmaschinen im JSON-Format kodiert. Die allgemeine Programmlogik von Synthea ist schematisch in Abbildung 2 zu erkennen.

Laut den Entwicklern berechnet jedes Modul dabei in der synthetischen Welt Zustand-sübergänge (falls vorhanden) für jede Person. Zeitschritte sind konfigurierbar und standardmäßig auf 7 Tage eingestellt. Jeder Zustand oder Übergang in einem Modul kann dann Zustandsbeginn, Begegnungen, Verschreibungen von Medikamenten oder andere klinische Ereignisse auslösen. Ein Beispiel wäre das Modul für Ohrenentzündungen bei Kindern. Hier bekommen Kinder mit unterschiedlichen Wahrscheinlichkeiten (je nach Alter) eine Ohrenentzündungen, werden dann durch eine Begegnung mit einem Arzt diagnostiziert und erhalten entweder ein Antibiotikum, ein Schmerzmittel oder beides. Dieses Beispiel zeigt verschiedene Arten von Zuständen und Übergängen (siehe Abbildung 3). (5)

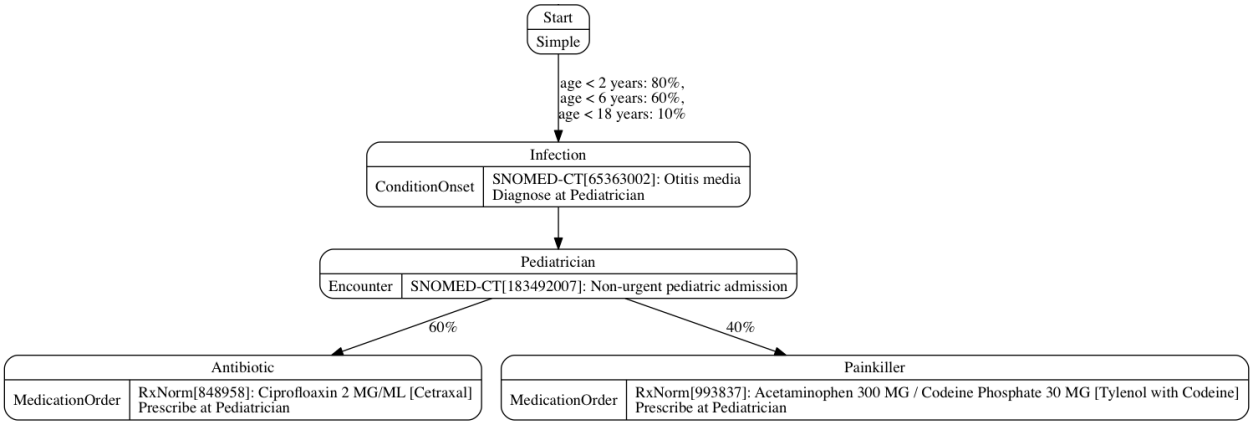


Abbildung 3: Teil des Moduls für Mittelohrentzündung (42)

Synthea besitzt 7 Arten von Steuerzuständen: Initial (Beginn eines Moduls), Terminal (Ende eines Moduls), Simple (um den Fluss zu steuern oder Lesbarkeit zu verbessern), Guard (gewissermaßen als Filter), Delay (pausiert das Modul um Zeit zu simulieren), SetAttribute (legt bzgl. eines Patienten ein Attribut-Wert-Paar fest) und Counter (Zähler für bestimmte Attribute) (5). Dazu enthält Synthea derzeit 11 Arten von klinischen Zuständen: Encounter, ConditionOnset, ConditionEnd, MedicationOrder, MedicationEnd, CarePlanStart, CarePlanEnd, Procedure, Observation, Symptom, und Death (5). Die Details der klinischen Zustände enthalten häufig medizinische Terminologie Codes und simulieren die medizinische Versorgung eines Patienten. Außerdem gibt es noch 4 Arten von Übergängen: direct, distributed, conditional und complex, welche den Ablauf und die Übergänge in den Modulen steuern (5).

Synthea wurde als Open-Source-Community-Projekt gegründet und damit erzeugte Patientendaten können für eine Vielzahl von sekundären Verwendungszwecken genutzt werden. Die Software selbst kann geändert oder in andere Projekte integriert werden, so Walonoski (5). Die Erfinder von Synthea sind also offen für die Weiterentwicklung Ihrer Software und hoffen so weitere Krankheitsmodule hinzufügen zu können und den Realismus der erzeugten Daten zukünftig zu steigern. Dafür gibt es sogar einen online Module-Builder (siehe Abbildung 4) mit dem eigene Module einfach per Drag and Drop zusammengestellt und bearbeitet werden können (12). In dem Paper werden am Ende auch zukünftige Arbeiten vorgeschlagen, wie evtl. das Erzeugen synthetischen Bilder (Röntgen- und Ultraschallbilder) oder die Modellierung von Organsystemen und -funktionen (zur Erzeugung von Herzfrequenzüberwachungsströmen, usw.). Allerdings sind die erzeugten synthetischen Daten nicht geeignet für die Erforschung von Krankheiten oder für klinische Entdeckungen, so die Autoren. (5)

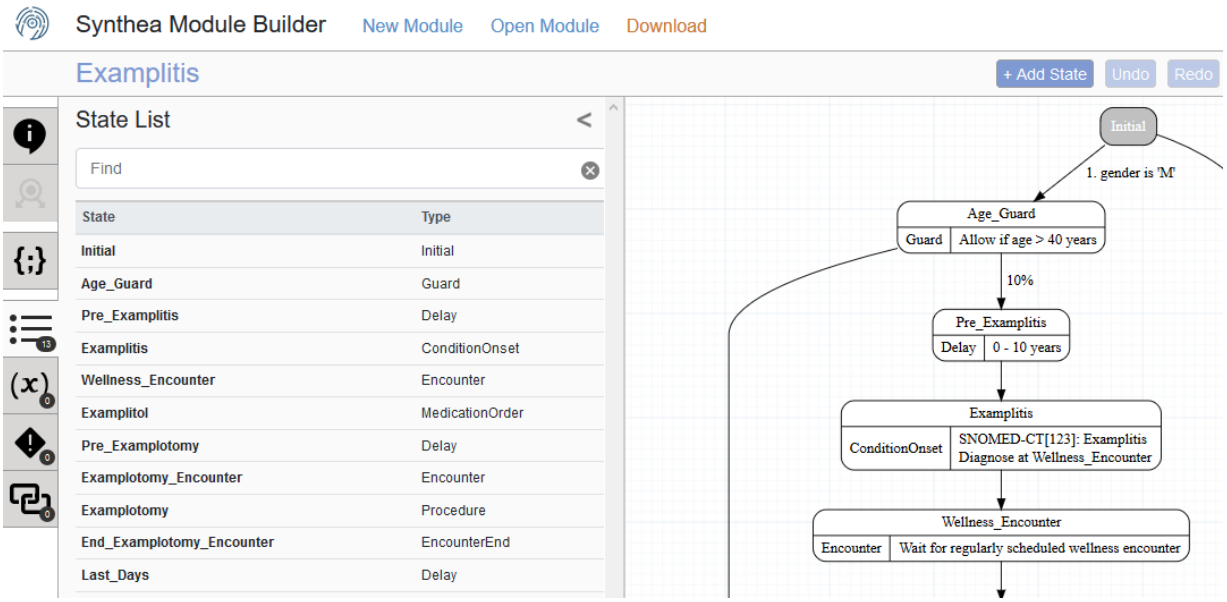


Abbildung 4: Ausschnitt des Synthea Module Builder (12)

Das Ziel realistische Daten für Software-Tests im Gesundheitswesen, Systemimplementierung oder für Schulungen zu generieren wird durch eine Software wie Synthea weitestgehend erreicht. Mittlerweile gibt es auch Studien, die die Qualität solcher Generatoren bewerten um den Realismus der erzeugten Patientendaten zu überprüfen.

In einer Arbeit von Chen et al. aus dem Jahr 2019 (6) werden die mit Hilfe des Open-Source-Generators Synthea erzeugte Daten mit realen Statistiken verglichen. Dabei wurden repräsentativ 1,2 Millionen Patienten aus Massachusetts benutzt, die von Synthea generiert wurde (der SyntheticMass Datensatz (43)). Für vier Merkmale wurden dann die Werte der „Patienten" mit den realen Daten der Bevölkerung von Massachusetts und den USA verglichen. Es wurde Darmkrebs-Screening, 30-Tage-Mortalität bei chronisch obstruktiver Lungenerkrankung (COPD), Komplikationsrate nach Hüft- oder Knieersatz und Kontrolle des Bluthochdrucks untersucht. Von der Gesamtbevölkerung von Synthea waren 394.476 für die Messung zur Darmkrebsvorsorge geeignet und 248.433 (63%) wurden als konform angesehen. Verglichen mit den öffentlich gemeldeten Massachusetts- und nationalen Raten von 77,3% bzw. 69,8% ist dieser Wert relativ genau. Bzgl. Der 30-Tage-Mortalität bei COPD starben von den 409 in Frage kommenden Patienten 0,7% innerhalb der Zeit nach der COPD-Exazerbation, gegenüber 7% in Massachusetts und 8% auf nationaler Ebene. Allerdings gab es dann einige größere Abweichungen bei den Daten bzgl. Komplikationen nach einem Hüft- oder Knieersatz-OP. Hier gab es bei keinem der Patienten von Synthea Komplikationen, während es in der realen Welt in Massachusetts 2,9% und in den USA national 2,8% waren. Ähnliches gilt für die Bluthochdruck Kontrolle nach Diagnose von Hypertonie. Dort hatte kein synthetischer Patient eine Kontrolle nach der Diagnose während es in Massachusetts: 74,52% und national: 69,7% waren. Die Ergebnisse zeigen also, dass Synthea bei der Modellierung der Demografie und der Wahrscheinlichkeiten von durchschnittlichen Gesundheitsvorsoge-Maßnahmen relativ zuverlässig ist. Die Möglichkeiten zur Modellierung heterogener Gesundheitsereignisse nach den Behandlungen sind jedoch begrenzt. Synthea und andere synthetische Patientengeneratoren modellieren derzeit keine Abweichungen in der Pflege und die potenziellen Ergebnisse, die sich daraus ergeben könnten. Um einen realistischeren Datensatz auszugeben, sollten die Generatoren weitere Faktoren nach den Behandlungen berücksichtigen (wenn Kliniker beispielsweise von der Standardpraxis abweichen), schlagen die Autoren vor. (6)

Und tatsächlich sind inzwischen einige der vorgeschlagenen Verbesserungen umgesetzt worden. Sowohl aus der Studie, als auch von den Synthea Entwicklern selbst, wurden Ideen umgesetzt und die Funktion der Software erweitert. So besteht mittlerweile die Möglichkeit physiologische Simulationen zu nutzen, um einige Vitalfunktionen und deren Werte zu generieren. Auch kann eingestellt werden, dass es zu „Fehlern" bei Größen-, Gewichts- und BMI-Werten für Personen unter 20 Jahren kommt oder Tod durch natürliche Ursachen vorkommt. Außerdem kann konfiguriert werden wie Patienten einen Gesundheitsdienstleister auswählen (nach Nähe, Qualität oder zufällig) oder ob es zu einer Vernachlässigung der Pflege kommt. Das meint, dass Patienten be-

stimmte Behandlungen nicht wahrnehmen und es ggf. zum „Tod durch fehlende Maßnahmen“ kommen kann, zum Beispiel wenn ein Patient nicht versichert oder eine Behandlung anderweitig unerschwinglich ist. Diese Neuerungen sorgen für zusätzliche Funktionen und de facto auch für mehr Realismus, wobei Synthea immer noch stetig weiterentwickelt wird.

3 Lösungsansatz

Die Hauptproblematik hier besteht also in den fehlende Testdaten im Bereich der Medizininformatik, wie Anfangs festgestellt. Es gibt aktuell keine Möglichkeit syntaktisch korrekte (Test-)Datensätze für die Nutzung im medizinischen Umfeld (in Deutschland) nach Bedarf automatisiert zu erzeugen und zu skalieren. Das Ziel besteht daher darin, eine (open-source) Software zu finden oder zu entwickeln die dazu im Stande ist. Dabei sind synthetisch generierte Daten deutlich praktischer als anonymisierte oder pseudonymisierte reale Datensätze. Und somit kann die Erzeugung künstlicher Patientendaten die klinische Forschung in Deutschland unterstützen und den Arbeitsaufwand vieler Entwickler in diesem Bereich verringern.

Um die derzeit bestehende Lücke in der Versorgung mit medizinischen Testdatensätzen zu schließen, bedarf es im Idealfall einer Software welche synthetische Daten erzeugen kann, die sich für die Nutzung im klinischen Umfeld eignen. So eine Software soll in dieser Arbeit gefunden und wenn nötig angepasst werden. Doch welche Daten müssen erzeugt werden können und welche Anpassung sind dafür notwendig?

Patientendaten sind personenbezogenen Informationen eines Patienten, die in einer medizinischen Einrichtung verarbeitet und gespeichert werden. Das meint sowohl Stammdaten als auch Informationen zur Diagnose und Behandlung. Solche Daten und Dateien sollten dann natürlich aus Gründen der Interoperabilität in einem allgemein anerkannten Format vorliegen. Die deutsche Medizininformatik-Initiative (MII) hat das Ziel, dass „jeder Arzt, jeder Patient und jeder Forscher in Zukunft Zugang zu den für ihn erforderlichen Informationen hat. Dies führt zu passgenaueren Diagnose- und Behandlungsentscheidungen, schafft neue Erkenntnisse für die wirksame und nachhaltige Bekämpfung von Krankheiten und trägt dazu bei, die Versorgung noch besser zu machen.“ (13). Dort werden auch die Weichen gestellt für die Zusammenarbeit der Kliniken und Einrichtungen indem ein Kerndatensatz entworfen wird, der gewisse Standards festlegt, auf die sich die Beteiligten geeinigt haben. In diesen Standards sollen am Ende auch die hier erzeugten Daten vorliegen, um eine Nutzbarkeit der Software zu gewährleisten. Es wurde beschlossen, dass „HL7 FHIR (Fast Healthcare Interoperability Resources) der Standard für die technische Repräsentation des MII-Kerndatensatzes ist.“ (44). Des Weiteren wurde in einer Formulierung dieses Kerndatensatzes festgehalten, dass „SNOMED CT als geeignete Nomenklatur für die Codierung verschiedener Datenelemente“ (45) vorgeschlagen wird. Außerdem sind im Modul der Laborbefunde LOINC und UCUM (also Laborparameter und Einheiten) „für die Strukturierung

und Codierung“ (45) vorgesehen. „UCUM (The Unified Code for Units of Measure) definiert eine international einheitliche maschinenlesbare Wiedergabe von Maßeinheiten“ (45). Außerdem sollen auch die ICD-10-GM, sowie der Operationen- und Prozedurenschlüssel (OPS) als Kodierung verwendet werden. (45)

Die zu erzeugenden Patentendaten sollen also im HL7 FHIR Format vorliegen und im besten Fall SNOMED CT, LOINC, UCUM, etc. als Kodierungen verwenden. Da es äußerst aufwendig wäre einen Generator von Grund auf zu programmieren, der syntaktisch korrekte und inhaltliche plausible Patientendaten erzeugen kann, ist eine Recherche ob es so etwas (oder etwas Ähnliches) bereits gibt, sinnvoll. Im Anschluss muss eine potenziell Software Lösung evtl. noch an die benötigten Gegebenheiten angepasst werden. Die notwendigen Anpassungen müssen dabei von dem derzeitigen Stand und den Funktionen der Software abhängig gemacht werden. Damit schlussendlich die Problematik der fehlenden Testdatensätze in der deutschen Medizininformatik gelöst werden kann.

4 Ausführung der Lösung

Zuerst möchte ich noch einmal kurz erläutern, warum Synthea der geeignetste Generator ist und welche Vorteile es bietet bzw. warum es keine Alternative gibt. Im Anschluss werde ich darauf eingehen, welche Änderungen vorgenommen werden, damit plausible Datensätze für den deutschsprachigen Raum erzeugt werden können.

4.1 Auswahl des Generators

Als Erstes habe ich also recherchiert welche Programme es gibt, die bei der Lösung des Problems helfen können. Da echte (anonymisierte) Datensätze diverse Nachteile haben, wie das sie z.B. aufwendig zu erhalten sind, in der Regel erst anonymisiert oder pseudonymisiert werden müssen, nicht beliebig skalierbar und teilweise nicht variabel genug sind, habe ich mich schon bei der Suche auf synthetische Daten konzentriert. Besser gesagt auf Software die diese erzeugen kann. Denn synthetische Daten sind wie schon erwähnt frei von jeglichen Datenschutz Richtlinien und benötigen keine zusätzliche Bearbeitung um verwendet werden zu dürfen. Außerdem besteht bei diesen Daten trivialerweise kein Risiko das sie re-identifiziert werden können, wie dies bei anonymisierten Datensätzen der Fall sein kann. Synthetische Daten sind maschinell generiert und damit auch beliebig skalierbar und bestenfalls anpassbar. Sie können dann auch eine große Ähnlichkeit zu echten Personen- bzw. Patientendaten aufweisen. Dadurch sind Entwickler und Anwender schlussendlich einfach freier in der Daten-Nutzung und dem Datenaustausch.

Wenn man zum Beispiel in PubMed (46), einer freien englischsprachigen Meta-Datenbank mit Bezug zu medizinischen Artikeln, nach Stichworten wie „synthetic patient data“ sucht, findet man einige Ansätze zur Erzeugung Synthetischer Patientendaten

oder Auswertungen dieser Methoden. Leider stellte sich heraus, dass viele davon eben nur Ansätze, Frameworks oder theoretische Konzepte sind. Also an fertigen Programmen zum Erzeugen künstlicher Daten mangelt es. Erst recht, wenn nach Patientendaten (zum Beispiel als EHR) gesucht wird. Auch bei der Suche nach „synthetic data generator" finden sich zwar Generatoren, welche zum Teil auch schon benutzbar sind, jedoch erzeugen diese keine Patientendaten, sondern beispielsweise Sensordaten oder Ähnliches. Auch bei der Recherche mit vergleichbaren Suchbegriffen und in anderen Meta-Datenbanken und Bibliotheken (wie bei Google-Scholar oder der Universitäts-Bibliothek) waren Generatoren für die Erzeugung synthetischer Patientendaten rar.

Die einzige vielversprechende Software Lösung die synthetische Patientendaten erzeugen kann war Synthea. Denn in der Arbeit von Walonoski et al. (5) wurde sowohl ein Ansatz erklärt, als auch ein fertiges Programm mitgeliefert. Das Projekt Synthea ist öffentlich und kostenlos nutzbar. Das war auch ein notwendiges Einschlusskriterium, denn Software oder Dienstleistungen, die zum Kauf angeboten werden und ggf. die gesuchten Patientendaten generieren, werden in dieser Arbeit per se nicht berücksichtigt. Außerdem war Synthea zum Zeitpunkt der Veröffentlichung schon betriebsbereit und wurde seitdem (zum Teil auch von der nutzenden Community) weiterentwickelt. Man hätte auch einen anderen vorgeschlagenen Ansatz zur Erzeugung künstlicher Datensätze benutzen und darauf basierend ein Programm schreiben können, da diese zum Teil ausgiebig evaluiert wurden. Es gibt zum Beispiel Wahrscheinlichkeitsmodelle, klassifikationsbasierte Imputationsmethoden und generative konträre neuronale Netze (engl.: „generative adversarial neural networks", GANs) zum Generieren von realistischen synthetischen Patientendaten (30). Jedoch sind diese Methoden teilweise sehr komplex und machine learning basiert, was bedeutet das Modelle trainiert werden müssen. Das Trainieren von solchen Modellen kann (Zeit-)aufwendig sein, muss manchmal überwacht werden und benötigt in jedem Fall ausreichend reale Daten als Grundlage. Was schon zu einem Problem führt, da passende reale Daten ja schwierig zu erhalten sind. Außerdem sind die entstandenen Modelle meist nur auf den zugrundeliegenden Datensatz trainiert, ist dieser beispielsweise nicht repräsentativ sind es die generierten synthetischen Daten auch nicht. Insgesamt sind diese Methoden vielversprechend und können realistische synthetische Patientendaten erzeugen, aber darauf basierend eine Software zu entwickeln (welche flexible und skalierbare Patientendatensätze generiert) wäre kompliziert und der Nutzen würde kaum den Aufwand rechtfertigen, wenn es schon ein Programm wie Synthea gibt.

Denn Synthea kann realistische synthetische Patendaten erzeugen, ohne das reale Datensätze vorausgesetzt werden. Lediglich veröffentlichte Statistiken, klinische Arbeitsabläufe und Krankheitsverläufe werden benötigt, welche im Normalfall nicht dem Datenschutz oder Ähnlichem unterliegen. Bei Synthea müssen keine Modelle trainiert werden, die Software ist öffentlich zugänglich und leicht zu bedienen. Die erzeugte „Patienten" sind dennoch realistisch und können für Softwaretests, Demonstrationen, Schulungen, etc. ohne Weiteres verwendet werden. Der Realismus kommt von den

verwendeten Statistiken, die dem Programm zugrunde liegen, welche für die USA umfangreich verfügbar sind. Allerdings wurden auch einige Wahrscheinlichkeiten geschätzt oder Sachverhalte vereinfacht, weshalb die von Synthea erzeugten Patentendaten keine medizinische Aussagekraft besitzen.

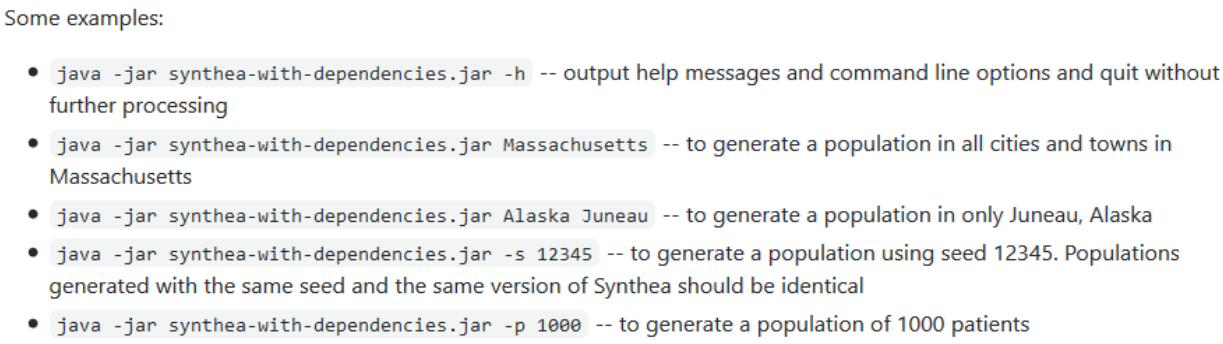
Da Synthea darüber hinaus die EHR im HL7 FHIR Format erzeugen kann (neben vielen anderen möglichen Formaten) und die Krankheiten und Untersuchungen mit SNOMED CT kodiert sind, entspricht dies genau dem was zum Beispiel die deutsche Medizininformatik-Initiative als geeignete Standards vorschlägt. Auch sind die Laboruntersuchgen bzw. die Einheiten in LOINC und UCUM kodiert. Summa summarum gibt es also aktuell wenig bis nichts was künstliche Patientendaten so einfach, flexibel und gleichzeitig so umfangreich generieren kann wie Synthea. Außerdem werden bereits die Standards genutzt und die Formate erzeugt, in denen die Daten am Ende vorliegen sollen. Da der Quelltext öffentlich verfügbar und gut dokumentiert ist, ist dieser Generator die beste Möglichkeit syntaktisch korrekte Datensätze für die Nutzung im medizinischen Umfeld nach Bedarf zu erzeugen. Da das Programm allerdings in den USA entwickelt wurde und die generierten Datensätze die anglo-amerikanische Bevölkerung repräsentieren, bedarf es einiger Anpassungen um die Software sinnvoll im deutschsprachigen Raum einsetzen zu können.

4.2 Anpassung der Software

Nachdem Synthea also als Generator feststand, musste dieser noch angepasst werden. Dabei waren mehrere Dinge auffällig „amerikanisch“ und einige Änderungen wurden auf Grund der verschiedenen Gesundheitssysteme durchgeführt, um die „deutschen Patienten“ realistischer erscheinen zu lassen. Als nächstes werde ich kurz erklären wie die Software im Allgemeinen aufgebaut ist und wie man Synthea benutzt. Um anschließend die notwendigen Veränderungen aufzuzeigen und zu beschreiben was ich bearbeitet habe, damit nutzbare Daten für den deutschsprachigen Raum erzeugt werden können.

Synthea wird als Konsolenprogramm ausgeführt, eine graphische Benutzeroberfläche gibt es nicht. Unter https://github.com/synthetichealth/synthea/wiki/Basic-Setup-and-Running gibt es eine einfache Anleitung zum Ausführen des Programms: Nach einem Download einer .jar Verzeichnis-Datei öffnet man die Eingabekonsole und geht in den Ordner, in den die Datei heruntergeladen wurde. Als nächstes gibt man dort den Befehl „java -jar synthea-with-dependencies.jar“ ein, Synthea wird ausgeführt und ein Patient wird mit den Standard Einstellungen generiert. Dabei wird immer ein FHIR Bundle erzeugt, also in einer Datei befinden sich neben dem Patienten selbst alle seine Beschwerden, Untersuchungen, Behandlungen, etc. Also de facto die gesamte Gesundheitshistorie, auch wenn manchmal nur von dem Patienten die Rede ist. Darüber hinaus kann man die Ausgabe von Synthea noch beeinflussen mit einigen Parametern in der Eingabe. Also zum Beispiel hinter dem Befehl [-g gender] für das Geschlecht der Patienten oder [state [city]] für Patienten aus einer bestimmten Stadt, in einem bestimmten Staat (der USA). (47)

Abbildung 5 zeigt dazu einige Beispiele. Darüber hinaus gibt es noch eine weitere Möglichkeit Synthea herunterzuladen und ausführen. Der „Developer Setup“ ist ausführlich unter https://github.com/synthetichealth/synthea/wiki/Developer-Setup-and-Running beschrieben (48). Diese Methode umfasst ein paar Schritte mehr, allerdings hat man am Ende auch mehr Freiheiten, da hier der source code vorhanden ist und bearbeitet werden kann. Die fertigen EHRs liegen nach dem Generieren im Output Ordner, den Synthea besitzt.


Some examples:

- `java -jar synthea-with-dependencies.jar -h` -- output help messages and command line options and quit without further processing
- `java -jar synthea-with-dependencies.jar Massachusetts` -- to generate a population in all cities and towns in Massachusetts
- `java -jar synthea-with-dependencies.jar Alaska Juneau` -- to generate a population in only Juneau, Alaska
- `java -jar synthea-with-dependencies.jar -s 12345` -- to generate a population using seed 12345. Populations generated with the same seed and the same version of Synthea should be identical
- `java -jar synthea-with-dependencies.jar -p 1000` -- to generate a population of 1000 patients

Abbildung 5: Einige Beispiele für Eingabeparameter in Synthea (47)

Intern besteht Synthea einerseits, neben einigen für den Build erforderlichen Dateien, aus den Java Klassen und JSON Dateien unter …\Synthea\synthea\src\main\java\... je nachdem wo das Verzeichnis gespeichert wurden. Dort wird die Logik berechnet, es werden unter anderem Patienten oder Ärzte generiert, die Module (Krankheiten) werden durchlaufen und am Ende werden die Daten je nach Konfiguration in die entsprechenden Formate (z.B. CCDA, FHIR STU3 oder FHIR R4) gewandelt und ausgegeben. Dafür sind die Java Dateien nochmal in Unterordner aufgeteilt, je nach Funktion. So findet man z.B. unter …\Synthea\synthea\src\main\java\org\mitre\synthea\export eine Klasse für jedes mögliche Dateiformat der erzeugten EHR, oder in dem Ordner …\Synthea\synthea\src\main\java\org\mitre\synthea\engine die Klassen, die das „Leben“ der Patienten simulieren.

Anderseits beinhaltet Synthea unter dem Pfad …\Synthea\synthea\src\main\resources\... die zugrunde liegenden Ressourcen. Also die Statistiken für die künstliche Bevölkerung (meist als CSV Dateien), die Krankheitsmodule auf die zugegriffen wird (inkl. deren Wahrscheinlichkeiten) und die synthea.properties Datei, mit der man einfache Konfigurationen vornehmen kann.

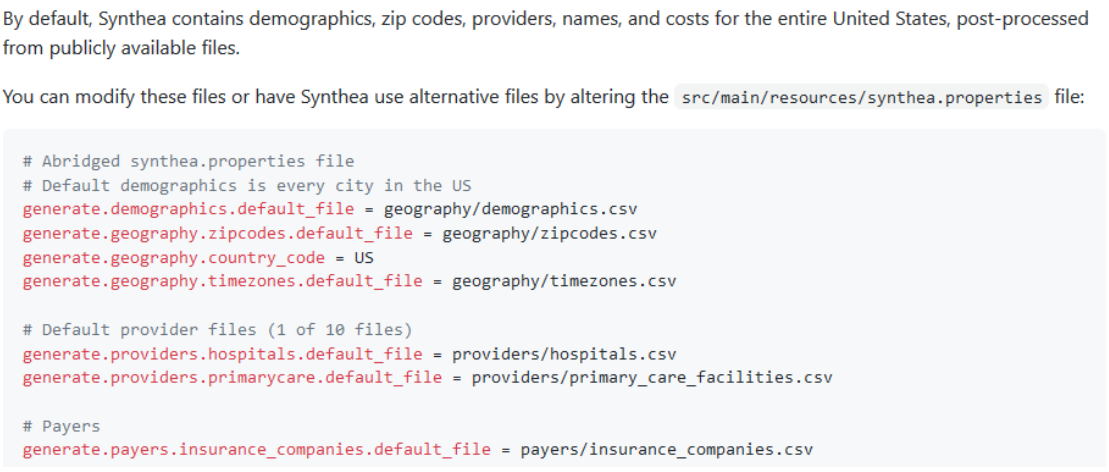
4.2.1 Konfigurationsdatei

Einige Funktionen können ganz einfach mithilfe einer Einstellungsdatei konfiguriert werden. Die Datei synthea.properties lässt sich im resources Ordner finden und kann mit einem Text Editor geöffnet und bearbeitet werden. Dabei kann zum Beispiel das Format angepasst werden, in welchem die EHR erzeugt werden. Dabei sind neben dem standardmäßig eingestelltem FHIR R4 noch CCDA, FHIR STU3 und DSTU2, CSV und ein einfaches Textdokument auswählbar. Es kann eingestellt werden ob für z.B. Krankenhausinformationen oder für die behandelnde Ärzte eine extra Datei erstellt werden soll. Darüber hinaus kann der Ausgabeordner verändert werden und es besteht die

Möglichkeit eine Datenbank zu erstellen, die in einer Datei die Ergebnisse werden zwischen den Läufen oder jedes einzelnen Laufes speichert. Diese kann dann auch mittels SQL abgefragt werden und die Ergebnisse daraus werden in einem zusätzlichen Textdokument gespeichert. Außerdem können in der properties Datei auch die CSV-Dateien mit den Statistiken zugewiesen, sowie andere Wahrscheinlichkeiten angepasst werden. Das bedeutet, dass statt den Standard Dateien mit den demographischen Daten der USA, andere Dateien verwendet werden können, um Patienten aus anderen geografischen Regionen zu erzeugen. Die Datei ist ausreichend kommentiert und auf der Wiki Seite des Synthea GitHub (49) findet sich eine Übersicht mit möglichen Änderungen für diese Datei. Zusätzlich habe ich dort noch eine Zeile geändert, nämlich den Country-Code, also das Länderkürzel. Diesen habe ich auf DE für Deutschland gesetzt: „generate.geography.country_code = DE“. Weitere Konfigurationen der Datei sin dem Nutzer überlassen und anhängig vom jeweiligen Anwendungszweck.

4.2.2 Demographie

Da natürlich in Deutschland lebenden Personen respektive Patienten erzeugt werden sollen, müssen die CSV Dateien angepasst oder ausgetauscht werden (siehe Abbildung 6). Dazu werden im …\resources\geography Ordner folgende Dateien angepasst: demographics.csv, zipcodes.csv und timezones.csv. Hierbei können diese durch neue Dateien ersetzt und in synthea.properties zugewiesen werden oder die bestehenden CSV Dateien werden angepasst. Hierbei würde ich der Einfachheit halber letzteres empfehlen. Denn beim Ausführen von Synthea werden diverse Tests gestartet um Fehler zu entdecken. Diese Java Tests benötigen an mehreren Stellen Städte oder Staaten aus den USA, auf die zurückgegriffen werden muss und geben Fehler aus, falls diese in der Datei fehlen. Das heißt wenn die CSV Dateien komplett ausgetauscht werden, müssen auch die Java Tests angepasst oder ausgetauscht werden. Ich habe also die bestehenden CSV Dateien für Städte, Postleitzahlen und Zeitzonen um deutsche Orte ergänzt. Durch Eingabe des entsprechenden Parameters (beim Ausführen von Synthea) lässt sich dann die Herkunft der Patienten anpassen. Dasselbe gilt auch für die Postleitzahlen- und die Zeitzonen Datei, sowie später für die Krankenhäuser und Versicherungen, die bei denen ebenfalls Daten hinzugefügt werden müssen.



By default, Synthea contains demographics, zip codes, providers, names, and costs for the entire United States, post-processed from publicly available files.

You can modify these files or have Synthea use alternative files by altering the `src/main/resources/synthea.properties` file:

```
# Abridged synthea.properties file
# Default demographics is every city in the US
generate.demographics.default_file = geography/demographics.csv
generate.geography.zipcodes.default_file = geography/zipcodes.csv
generate.geography.country_code = US
generate.geography.timezones.default_file = geography/timezones.csv

# Default provider files (1 of 10 files)
generate.providers.hospitals.default_file = providers/hospitals.csv
generate.providers.primarycare.default_file = providers/primary_care_facilities.csv

# Payers
generate.payers.insurance_companies.default_file = payers/insurance_companies.csv
```

Abbildung 6: Ausschnitt aus synthea.properties, die zu sehenden Dateien müssen angepasst werden um deutsche Patienten generieren zu können (50)

Um Städte in der demographics.csv Datei hinzuzufügen habe ich dann nach einigen Statistiken gesucht. Der Aufbau der CSV Tabelle ist klar definiert, es gibt eine einzelne Reihe für jede Stadt und die Spalten beinhalten die entsprechenden Daten. Dabei benötigt man für jede Stadt: eine ID (einzigartige Identifikationsnummer), eine County Nummer, den Stadtname, den Namen des (Bundes-)Staates, die Einwohnerzahl der Stadt, den Namen des County, die Einwohnerzahl des County, Prozentsatz der männlichen und der weiblichen Bevölkerung, den jeweils prozentualen Bevölkerungsanteil der Ethnien, eine Altersverteilung der Bevölkerung in der Stadt, eine prozentuale Verteilung der Einwohner nach Einkommen und nach höchstem Bildungsabschluss. Dabei ist es wichtig die exakte Reihenfolge einzuhalten. Eine genauere Aufschlüsselung der erwarteten Spalteninhalte, sowie eine Übersicht findet man wieder auf der entsprechenden Wiki Seite des Synthea GitHub (50). Dabei war es nicht immer möglich Statistiken für jede einzelne „Rubrik" zu finden und ich habe ein paar Vereinfachungen vorgenommen. Da Synthea aber per se keine medizinische oder statistische Aussagekraft besitzt und die generierten Daten nur für Softwaretests, Demonstrationen, etc. genutzt werden, werden die gemachten Vereinfachungen den Realismus nicht schmälern (die Struktur der Daten bleibt erhalten).

So wird zum einem ein County benötigt, was es in Deutschland so nicht gibt. Am ehesten wäre dies mit einem Landkreis zu vergleichen, jedoch haben die Countys in den USA mehr Befugnisse und zum Teil eigenen (Landes-)Regierungen je nach Bundesstaat. Daher habe ich vereinfachend festgelegt das ich sowohl in die Spalte für County, als auch bei State, die Daten des jeweiligen Bundeslandes eintrage. Dass Daten dort doppelt vorkommen führt nicht zu Fehlern, sofern sie innerhalb der zulässigen Werte sind. Als Beispiel-Städte habe ich Leipzig und Dresden gewählt, es wären aber auch andere möglich gewesen. Das bedeutet, dass ich in den CSV Dateien eine Reihe pro Stadt ergänze und dabei bei County (und State) Sachsen eintragen muss. Während der Suche nach Daten fiel mir dann auf, dass es für z.B. Leipzig von offizieller Seite keine aussagekräftige Statistik über die Einkommensverteilung der Einwohner nach den von Synthea benötigten Einkommensgruppen gab. Auch für andere Großstädte habe ich wenig veröffentlichte Statistiken dazu gefunden. Als Lösung für dieses Problem nehme ich den bundesdeutschen Durchschnitt an dieser Stelle, da ich einzig vom statistischen Bundesamt eine entsprechende öffentliche Statistik zur benötigten Einkommensverteilung in Deutschland gefunden habe. Auch hierbei gehe ich davon aus, dass der „Realismus" der generierten Patienten wenig beeinflusst wird, da es zwar regionale Unterschiede beim Einkommen gibt, unser Gesundheitssystem aber solidarisch ist und alle gesetzlich Versicherten prinzipiell den gleichen Versorgungsanspruch haben. Anders als beispielsweise in den USA wo Einwohner sich privat versichern oder Behandlungskosten selbst tragen müssen. Intern berechnet Synthea daraus und aus der Bevölkerungsverteilung nach höchstem Bildungsabschluss einen sozio-ökonomischen Wert, welcher Einfluss auf das Auftreten einiger Krankheiten hat. Dabei ist die Wahrscheinlichkeit an bestimmte Krankheiten zu leiden verringert, wenn ein Patient einen höheren Wert hat (also ein höheres Einkommen und/oder eine bessere Bildung). Da das deutsche Gesundheitssystem aber auf einem Solidaritätsprinzip basiert ist es

vielleicht sogar realistischer, wenn durch die gleiche Einkommensverteilung in allen Städten der sozio-ökonomische Wert angeglichen wird.

Ähnlich ist es mit der Bevölkerungsverteilung nach Ethnien, wie sie von Synthea benötigt wird. Dort wird die prozentuale Bevölkerungsverteilung nach den Ethnien „White, Hispanic, Black, Asian, Native, Other" (50) verlangt. Solche Statistiken werden scheinbar für US-amerikanische Städte erhoben, für eine deutsche Stadt habe ich dazu aber keine Daten gefunden (höchstens zur Verteilung nach Nationalität, die aber kein Rückschluss auf die entsprechende Ethnie zulässt). Daher habe ich mich hier entschlossen den Wert für „White" auf 1.0 (100%) und alles anderen Werte (bzgl. der Ethnie) auf 0.0 zu setzen, damit alle Patienten unabhängig ihrer Ethnie generiert und simuliert werden. Die übrigen Werte für die CSV Dateien ließen sich online sammeln, aus z.B. Quartalsberichten oder statistischen Jahrbüchern die vom Amt für Statistik und Wahlen Leipzig auf deren Website veröffentlicht wurden (51), bzw. für Dresden von der äquivalenten Dresdner Stelle. So kann man mit mehr Aufwand theoretisch jede deutsche Stadt hinzufügen und am Ende Patienten aus dem gesamten Bundesgebiet generieren.

Auch für die Dateien zipcodes.csv mit den Postleitzahlen und timezones.csv für die Zeitzonen füge ich dann exemplarisch Leipzig und Dresden hinzu. Die Zeitzone zu ergänzen ist kein Problem, dafür benötigt man eine Zeile pro State (bzw. Bundesland) mit folgenden Daten: State (hier also Sachsen), Abkürzung State (SN), Zeitzone (hier Central European Time) und die Abkürzung dafür (also CET). Die Postleitzahlen (PLZ) hinzuzufügen ist etwas umständlicher, vor allem da Längen- und Breitengrade benötigt werden. Diese dienen dem System dazu ein Krankenhaus oder andere Gesundheitseinrichtungen in der Nähe der Patienten zu finden. Die zipcodes.csv Datei enthält mehrere Zeilen pro Stadt, nämlich genau eine pro PLZ. Darin ist die erste Spalte leer (und wird von Programm ignoriert), gefolgt vom State (bzw. Bundesland), der Abkürzung des Bundeslandes, der Stadt, einer PLZ der Stadt und dann dem Breiten- und zuletzt dem Längengrad. Dabei sind 34 Postleitzahlen für Leipzig und 29 für Dresden von mir eingefügt wurden. Die geografischen Koordinaten dafür habe ich dann mit Hilfe von Google Maps gesucht, indem ich mir jeweils den Bezirk jeder PLZ habe anzeigen lassen, um dann daraus einen Punkt mittig auszuwählen. Für diesen Punkt habe ich dann die Breiten- und Längengrade genommen und in das Dokument eingefügt, um Leipzig und Dresden zu komplettieren. Da nun neue Städte erstellt wurden, aus denen die Patienten kommen können, benötigt Synthea noch entsprechende Krankenhäuser und Versicherer in der Nähe damit Patienten auch versorgt werden.

4.2.3 Krankenhäuser & Versicherer

Also müssen noch die Dateien hospitals.csv, primary_care_facilities.csv und urgent_care_facilities.csv aus dem resources\providers Ordner in Synthea ersetzt oder ergänzt werden. Hierbei füge ich jeweils ein Krankenhaus pro Stadt ein (nämlich die Universitätskliniken aus Leipzig und Dresden), mehr könnten aber nach Belieben ergänzt werden. Die Daten dafür habe ich von den Webseiten der Krankenhäuser und die geographischen Koordinaten (mit Google Maps gefunden) füge ich ebenfalls im

Anschluss ein. In der hospitals.csv Datei ist dabei einen Zeile pro Klinik zu füllen. Es werden dort laut den Entwicklern von Synthea folgende allgemeine Informationen benötigt: eine ID (eindeutige Kennung), der Name des Krankenhauses, die Adresse (Straße und Hausnummer), der Name der Stadt, die Abkürzung des Bundeslandes, die PLZ, das County (hier also das Bundesland) und eine Telefonnummer. Dazu muss dann noch „type" angegeben werden, also die Art des Krankenhauses (hier als String ‚Universitätsklinik' eingetragen), sowie „ownership" also die Verwaltung (dort überall ‚Government' eingetragen da das Feld aktuell nicht genutzt wird). Außerdem hat die Datei noch je eine Spalte für „emergency" (ob Notaufnahme vorhanden ist), „quality" (von 1 bis 5; da meinerseits keine Wertung möglich ist, habe ich jeweils das Maximum 5 eingetragen) und schlussendlich wieder für die Breiten- und Längengrade. (52)

In der urgent_care_facilities.csv und primary_care_facilities.csv Datei gehe ich dann genauso vor und ergänze je eine Reihe pro Krankenhaus. Die Daten sind dort teilweise in einer anderen Reihenfolge einzugeben und in primary_care_facilities.csv gibt es eine Menge ungenutzter Spalten, die aktuell leer bleiben können. Jedoch sind die meisten Informationen identisch zu denen in der hospitals.csv Datei. Die übrigen Dateien im providers Ordner müssen nicht zwangsweise ergänzt werden, da aktuell nur die 3 genannten genutzt werden um Krankenhäuser für Patienten auszuwählen (52).

Als nächstes muss die Datei insurance_companies.csv aus Syntheas resources\payers Ordner ergänzt werden, damit die „Patienten" einen Versicherer aus Ihre Nähe finden können. Dort wird eine Zeile pro Krankenkasse verwendet und ich habe als Beispiel die AOK Plus gewählt, eine ortsansässige Krankenkasse für Sachsen. Die benötigten Informationen nehme ich dabei wieder von der Website der Krankenkasse. Wie schon bei den Kliniken wäre hier aber auch jede andere Krankenkasse in Sachsen denkbar gewesen. Da in den USA keine Versicherungspflicht besteht und quasi jede Versicherung freiwillig ist, könnte man dies bei den deutschen Versicherern evtl. unterscheiden. Dafür könnte man die letzte Spalte „ownership" verwenden und entweder ‚Government' für gesetzliche oder ‚Private' für Private Versicherungen eintragen. Jedoch wird dieser Unterschied aktuell von Synthea nicht berücksichtigt. Im Allgemeinen werden jedoch für jede Versicherung folgende Daten benötigt: ein Laufnummer (wird vom Programm ignoriert), eine ID, der Name, die Adresse, die Stadt und den Staat in dem der Hauptsitz der Versicherung ist (hier also SN für Sachsen), sowie die PLZ und eine Telefonnummer. Darüber hinaus müssen die Staaten/Bundesländer angegeben werden, in denen die Versicherung arbeitet (alternativ ein * statt einem String möglich, wenn in allen Staaten verfügbar) und die Dienstleistungen, die abgedeckt werden (auch hier ein * möglich, wenn alles angeboten). Des Weiteren gibt es noch die Spalten „deductible" (der Betrag, den Patienten vorauszahlen bevor die Versicherung erstattet), „default_coinsurance" (derzeit nicht verwendet), „default_copay" (die Selbstbeteiligung des Patienten bei jeder Behandlung) und „monthly_premium"(die monatliche Gebühr um ein Premium Modell bzw. Zusatzleistungen der Versicherung zu erhalten). Da ich keine genauen oder allgemeingültigen Daten gefunden habe, habe ich in diesen Spalten überall 0 eingetragen. Außerdem wird ein Eintrag für die oben erwähnte Spalte „ownership" benötigt. Auch diese Datei ist im Synthea Wiki dokumentiert.

4.2.4 Kosten

Ein weiteres Detail was angepasst werden könnten, um den Realismus der erzeugten Daten zu steigern, sind die veranschlagten Kosten. In der synthea.properties Datei werden die Kosten für verschiedene Behandlungen stark vereinfachend festgelegt (siehe Abbildung 7). Diese Kosten lasse ich unverändert, da ich an der Stelle keine Informationen zu deutschen Durchschnittskosten hatte. Sofern benötigt, können diese Werte jedoch einfach ersetzt werden. Eventuell kann man hier auch etwas differenzieren und unterschiedliche Kosten für verschiedene Arten von Behandlungen (oder Medikamenten) veranschlagen.

```
124
125  # Default Costs, to be used for pricing something that we don't have a specific price for
126  # -- $500 for procedures is completely invented
127  generate.costs.default_procedure_cost = 500.00
128  # -- $255 for medications - also invented
129  generate.costs.default_medication_cost = 255.00
130  # -- Encounters billed using avg prices from https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3096340/
131  # -- Adjustments for initial or subsequent hospital visit and level/complexity/time of encounter
132  # -- not included. Assume initial, low complexity encounter (Tables 4 & 6)
133  generate.costs.default_encounter_cost = 125.00
134  # -- https://www.nytimes.com/2014/07/03/health/Vaccine-Costs-Soaring-Paying-Till-It-Hurts.html
135  # -- currently all vaccines cost $136.
136  generate.costs.default_immunization_cost = 136.00
137
```

Abbildung 7: Ausschnitt aus synthea.properties, an dieser Stelle können Kosten für Behandlungen oder Medikamente festgelegt werden

4.2.5 Namen & Sprache

Nachdem nun die Patienten aus deutschen Städten kommen und in deutschen Krankenhäusern behandelt werden, müssen noch ein paar Eigenschaften der Patienten selbst angepasst werden. Wie zum Beispiel die Namen und die Sprache. Synthea berechnet die Namen der Patienten aus ihrer Sprache und diese ist wiederum abhängig von der jeweiligen ethnischen Zugehörigkeit. In der Demographics.java Datei (unter src\main\java\org\mitre\synthea\world\geographic\ zu finden) wird dies durch die Funktion *public String languageFromRaceAndEthnicity* ausgewählt. Dort ist die Ethnie einer Person abhängig von den Wahrscheinlichkeiten aus der demographics.csv Datei und die Sprache dann wieder von den Wahrscheinlichkeiten jeder Ethnie in der Java Klasse. Zum Beispiel sprechen dort US-Amerikaner mit der Ethnie „black“ zu 97% englisch, zu 2,6% spanisch und zu 0,4% französisch (Diese Wahrscheinlichkeiten sind im Source code mit einer Quelle belegt). Da für die deutschen Städte jetzt nur „white“ möglich ist, habe ich auch die obige Funktion entsprechend angepasst. So ändere ich die Wahrscheinlichkeiten der Sprachen also dahingehend, dass Personen mit „white“ jetzt zu 100% „german“ Sprechen. Also werden aktuell aus Deutschland stammende Patienten auch Deutsch als Sprache haben und demzufolge auch deutsche Namen. Die Namen werden dann aus der names.yml Datei im resources-Ordner gewählt. Diese Datei enthält jeweils eine Liste (Array) mit männlichen Vornamen, weiblichen Vornamen und mit Familiennamen. Aus diesen Listen wird dann ein zufälliger Vor- und Nachname beim Generieren von Personen gewählt. Ursprünglich waren diese Listen mit meist englischen Namen bestückt, die im Nordamerikanischen Raum üblich sind. Diese ersetze ich durch die gängigsten deutsche Vor- und Nachnamen, auf Basis

von verschiedenen online zu findenden Verzeichnissen (53), zu denen ich teilweise weitere Namen ergänze. Diese sind möglicherweise nicht repräsentativ und dass es in Deutschland nur Patienten mit deutschen Namen gibt ist nicht zu 100% realistisch, jedoch ist dies deutlich realitätstreuer als Patienten mit anglo-amerikanischen Namen. Die names.yml Datei kann jederzeit in einem Texteditor geöffnet und nach Belieben um weitere Namen ergänzt werden.

4.2.6. Sonstige Anpassungen

Als nächstes bedarf es noch ein paar kleinen Änderungen, die auffällig amerikanisch wirken. Zum Beispiel die Währung oder das Format der Adressen, Telefonnummern, Sozialversicherungsnummern et cetera. Die meisten der folgenden Anpassung sind nicht über Ressourcen möglich, sondern müssen direkt in Quellcode von Synthea gemacht werden. Wie zum Beispiel die Währung für die entstehenden Kosten. Da in den entstehenden Gesundheitsakten (EHRs) neben den Daten des Patienten alle Behandlungen, Untersuchen und Medikamente festgehalten werden, werden auch Kosten dafür veranschlagt und festgehalten. Diese sind im Original in „USD" (US-Dollar) angegeben. Das wird in den Export Klassen festgelegt, also in den Java Dateien, die die generierten Informationen in das richtige Format bringen. Diese sind unter dem Ordner synthea\src\main\java\org\mitre\synthea\export zu finden. Da die EHRs am Ende als JSON Dokumente im FHR R4 Standard erzeugt werden sollen, bearbeite ich also die Datei FhirR4.java. Darin wird an mehreren Stellen durch den Befehl *moneyResource.setCurrency("USD");* die Währung für eine Behandlung oder Ähnliches auf Dollar festgelegt. Ich habe hier alle Strings von USD auf „EUR" geändert um somit die Kosten in Euro anzeigen zu lassen. Diese Anpassung ist nur für das R4 Format gemacht wurden, also wenn EHRs beispielsweise im FHIR STU3 Standard erzeugt werden sollen müssten die entsprechenden Klassen ebenfalls noch angepasst werden.

Danach passe ich die Telefonnummern an, welche in der Klasse LifecycleModule.java erzeugt wird. Die Datei befindet sich unter folgendem Ordner: synthea\src\main\java\org\mitre\synthea\modules und erzeugt quasi den Patienten. Dort werden Personen „geboren", bekommen ihre Stammdaten zugewiesen, altern und durchlaufen ihr Leben. Die hier entstehenden Daten werden am Schluss durch eine Export Klasse in das gewünschte Format überführt und ausgegeben. Wie in den USA üblich beginnen fiktive (teils auch reale) Telefonnummern auch in Synthea mit 555. Es werden also Nummern nach dem folgendem Schema erzeugt: mit 555 beginnend, gefolgt von drei zufälligen Ziffern und nach einem Trennzeichen nochmal vier zufällige Ziffern (z.B. 555-123-4567). In Deutschland haben Telefonnummern den Präfix „0" gefolgt von drei Ziffern entsprechend des örtlichen Netzes und abschließend sieben oder acht Ziffern für den jeweiligen Anschluss (also z.B. 0123 4567890). Nach diesem Muster lasse ich dann auch die Telefonnummern in Synthea erzeugen: nach der 0 kommen drei zufällige Ziffern, gefolgt von einem Leerzeichen als Trennung und am Ende noch sieben zufällige Ziffern. Die entsprechenden Zeilen sind in Abbildung 8 zu sehen.

```
167
168     //Telefonnummer an DE angepasst, Format z.B. 0123 9876543
169     String phoneNumber = "0" + ((person.randInt(999 - 100 + 1) + 100)) + " "
170         + ((person.randInt(9999999 - 1000000 + 1) + 1000000));
171     attributes.put(Person.TELECOM, phoneNumber);
172
173    //hier Sozialversicherungsnummer an DE angepasst, sie entspricht aber nicht den
174    //Daten des Patienten (Name, Geb.Datum, etc.) sondern ist nur äußerlich ähnlich
175     String ssn = ((person.randInt(99 - 10 + 1) + 10)) + " "
176         + ((person.randInt(999999 - 100000 + 1) + 100000)) + " "
177         + randomLetter() + " " + ((person.randInt(999 - 100 + 1) + 100));
178     attributes.put(Person.IDENTIFIER_SSN, ssn);
```

Abbildung 8: Ausschnitt aus LifecycleModule.java, zu sehen ist wie die Telefonnummer und die Sozialversicherungsnummer erzeugt werden

Des Weiteren habe ich die Sozialversicherungsnummer der Patienten überarbeitet, damit diese den deutschen Nummern ähnlicher sind. Diese werden in derselben Klasse wie die Telefonnummern erzeugt. Die Sozialversicherungsnummer (SVN) besteht in den USA aus neun Ziffern und jede Person hat, wie hier auch, zeitlebens nur eine eindeutig identifizierende Nummer. Damit die durch Synthea generierte SVN keiner echten vergebenen Nummer gleicht (wie bei den Telefonnummern), beginnen hier alle mit „999“. Danach kommen zwei zufälligen Ziffern und nach einem Trennzeichen noch vier zufällige Ziffern (z.B. 999-12-3456). In Deutschland besteht die SVN (auch Rentenversicherungsnummer) jedoch aus 12 alphanumerischen Zeichen, die auch nur einmal pro Bürger vergeben werden. Dabei spielen persönliche Informationen wie das Geburtsdatum oder der erste Buchstabe des Nachnamens eine Rolle. Damit Synthea den deutschen Patienten eine realistischere SVN gibt, habe ich auch hier die entsprechende Stelle in der LifecycleModule.java Klasse angepasst (siehe Abbildung 8). Es werden also erst zwei zufällige Ziffern generiert, dann nach einem Trennzeichen nochmal 6 Ziffern, danach kommt ein zufälliger Buchstabe (den die Hilfsfunktion *randomLetter()* liefert) und zum Schluss eine zufällige dreistellige Zahl. Eine deutsche SVN könnte jetzt zum Beispiel so aussehen: „12 150585 P 016“, wobei die Daten zufällig generiert sind und nicht das wirkliche Geburtsdatum des Patienten oder den ersten Buchstaben seines Nachnamens verwenden.

In der Klasse LifecycleModule.java wird dann ebenfalls noch die Führerscheinnummer, sowie die Passnummer der Patienten berechnet. Wenn ein künstlicher Patient „altert“ werden zu bestimmten Zeitpunkten Attribute zugefügt. Es werden im Original Führerscheinnummern hinzugefügt, wenn ein Patient 16 wird. Diesen Wert habe ich auf 17 gesetzt, da man in Deutschland früher keinen Führerschein bekommen kann. Auch die Nummer selbst habe ich angepasst. In Synthea wurde diese mit dem Präfix „S999“ erzeugt und fünf zufälligen Ziffern als Suffix, da dies dem amerikanischen Schema entspricht. In Deutschland bestehen Führerscheinnummer aus elf alphanumerischen (zum Teil zufälligen) Zeichen. Um das zu simulieren lasse ich nun eine Nummer erzeugen, die mit einem Buchstaben beginnt, dann drei Ziffern, wieder zwei Buchstaben und zum Schluss nochmal fünf Ziffern hat (alles zufällig generiert). So bekommen die Patienten für den deutschen Raum realistischere Führerscheinnummern. Genauso verhält es sich mit den Passnummern, welche in derselben Datei generiert werden. Ursprünglich werden diese Nummern beginnend mit „X“ generiert, erhalten dann eine zufällige

achtstellige Zahl und zum Schluss wieder ein „X“ (z.B. X12345678X). Für die deutschen Patienten lasse ich dann eine Passnummer generieren, die nicht mit X beginnt und endet, sondern mit einem zufälligen Buchstaben und dann acht weitere Ziffern beinhaltet. Die jetzt generierten Nummern ähneln somit mehr den deutschen Pass- oder Ausweisnummern als es vorher der Fall war.

Zuletzt ändere ich noch das Adressformat ein wenig, damit die Adressen nicht so US-amerikanisch aussehen. Die Straßen-Namen werden von Familiennamen (aus der names.yml Datei) abgeleitet und die Hausnummern per Zufall erzeugt. Wobei eine Straße dann aus dem eigentlichen Namen und dem Typ (z.B. Avenue, Boulevard, Road, Street, Alley, etc.) zusammengesetzt ist. In der names.yml Datei muss dann für Deutschland den „type“ angepasst werden, dort habe ich „Straße, Weg, Allee, Gasse, Platz, Ring, Pfad“ gewählt. Also wohnen die Patienten in zufälligen Straßen wie zum Beispiel: „Fischer Straße“ oder „Wagner Platz“. Als nächstes habe ich im Quellcode die Reihenfolge angepasst wie die Adresse ausgegeben wird. Da in den USA die Hausnummer vor dem Straßennamen steht, gefolgt von einem Zusatz (Appartement, Suite, etc.) muss dies umgekehrt werden. Denn in Deutschland schreibt man erst die Straße, dann die Hausnummer und so werden die Adressen in Synthea nun auch erzeugt. Den Adresszusatz habe ich ganz entfernt, da es im deutschen eher unüblich ist, zum Beispiel die Wohnung zu beziffern und als Zusatz in die Adresse zu schreiben. Im Original werden also Adressen wie „203 East Street, Suite 1157“ generiert und nach den Anpassungen lauten diese beispielsweise „Obermann Straße 17“. Damit können jetzt summa summarum vollständig deutsche Patienten samt Ihrer Gesundheitshistorie erzeugt werden, da der medizinische Kontext keine großen Änderungen benötigt.

4.2.7 Verifizierung

Weil Synthea nach den gemachten Anpassungen deutsche Patienten generieren kann, muss noch überprüft werden ob die entstehenden Daten auch fehlerfrei erzeugt und syntaktisch korrekt verarbeitet werden können. In der Eingabekonsole habe ich dann *run_synthea Sachsen „Leipzig“* eingegeben um Patienten aus Leipzig zu erhalten, alternativ könnte die gewünschte Stadt in Anführungszeichen weggelassen werden um Patienten aus einer zufälligen Stadt in Sachsen zu erhalten. Die gemachten Änderungen führen in Synthea zu keinen Fehlern und die Ausgabe der deutsche Patienten erfolgte wie gewohnt (siehe Abb. 9), man sieht der Patient „Jeremy Schirmer“ wurde erzeugt. In der Datei selbst sind neben dem Deutschen Namen auch die Adressen und andere Stammdaten korrekt und der Patient wurde in einem deutschen Krankenhaus behandelt. Also sind die Änderungen erfolgreich gewesen und es können nun beliebig viele Patienten nach diesem Schema generiert werden. Lediglich die Struktur der FHIR Daten muss noch überprüft werden um zu testen ob die Dateien syntaktisch fehlerfrei generiert werden. Dazu teste ich die entstehenden JSON Dokumente mit Hilfe eines FHIR-Servers.

Abbildung 9: Ausgabe der Konsole (nach Änderungen)

Als Test-Server wähle ich die Open-Source Software HAPI FHIR. Diese ist eine Implementierung des HL7-FHIR-Standards in Java unter steht unter der Apache Software License 2.0 (54). Neben dem Testserver stellt die HAPI FHIR Bibliothek noch weitere Funktionen zur Verfügung. Laut den Entwicklern von HAPI FHIR gibt es unter anderem ein einfach zu nutzendes Command Line Tool, also ein fertiges Programm, das über die Eingabeaufforderung einen FHIR-Server starten kann. Darüber hinaus stehen einige integrierte Funktionen von HAPI FHIR, sowie benutzerfreundliche Befehle und Beispielressourcen zur Verfügung. Das Konsolenprogramm kann einfach heruntergeladen und installiert werden und ist danach einsatzbereit. Mit dem Befehl „hapi-fhir-cli run-server -v R4“ startet man über die Konsole den (FHIR R4) Server, der lokal auf dem jeweiligen Rechner läuft und testweise FHIR Daten über HTTP REST Anfragen speichern und verarbeiten kann. Eine genauere Anleitung findet sich auf der Dokumentationsseite von HAPI FHIR. (55)

Nachdem ich den Server also lokal gestartet habe, öffne ich die Testwebseite (siehe Abb. 10) indem ich in einem Browser auf http://localhost: 8080/ zugreife. Dort habe ich versucht einige der erzeugte FHIR Bundles auf den Server zu laden. Da die meisten Dateien im Ganzen leider zu groß waren, musste ich die einzelnen JSON Dateien teilen und mehrere Ausschnitte des FHIR Bundles einzeln hochladen. Dies ist danach recht simpel über den entsprechenden Button „Transaction“ auf der Webseite des Testservers möglich. Darüber kann ein Bundle mit mehreren Ressourcen auf dem Server gepostet und gespeichert werden in einer einzelnen Transaktion. Nach dem einfügen des JSON Codes wird dabei ein HTTP POST Befehl ausgeführt, der die eingegebenen Ressourcen auf dem Server speichert. Anschließend bekommt man gemäß dem REST Standard eine Statusmeldung ob die Transaktion erfolgreich war. Nach dem Unterteilen der erzeugten FHIR Bundles in kleinere Bundles waren die Transaktionen jedoch stets erfolgreich, der Server lieferte als Antwort „HTTP 200 OK“. Das bedeutet, dass das Bundle jeweils fehlerfrei auf den Server geladen und gespeichert werden konnte.

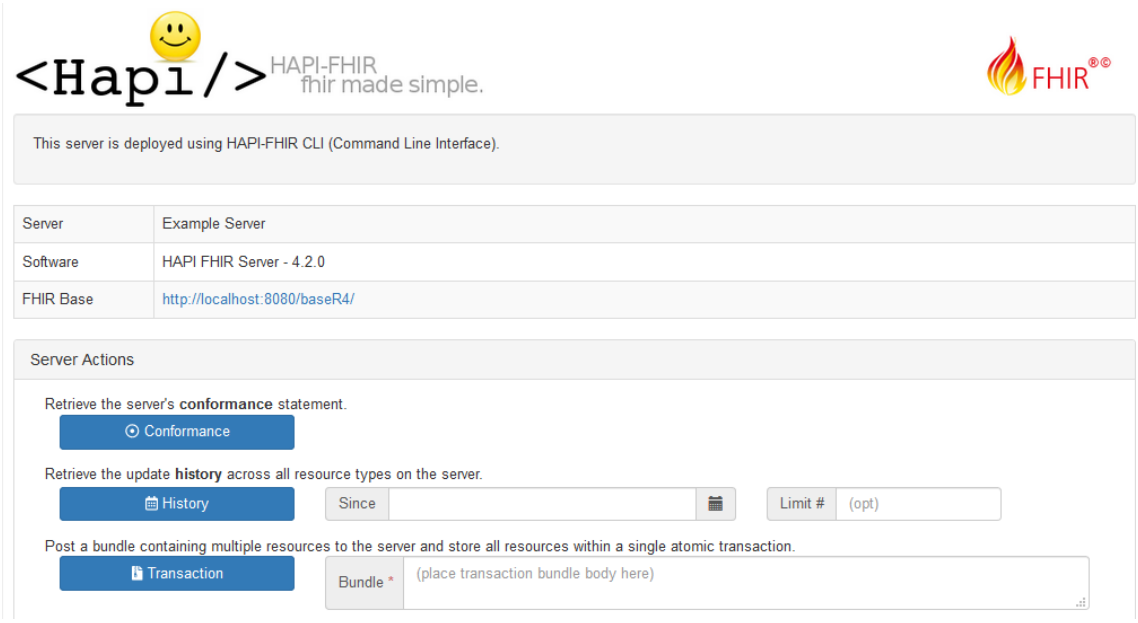


Abbildung 10: Startseite des HAPI FHIR Testservers

Danach wird schon die Anzahl aller vorhandenen FHIR Ressourcen (Patient, Observation, Practitioner, etc.) auf der Startseite des Servers angezeigt. Diese können einfach aufgerufen oder geupdatet werden. Man kann auch einzelne Ressource suchen bzw. filtern und das Abrufen der Ressourcen (wobei eine HTTP GET Anfrage gestellt wird) funktioniert ebenfalls. Hierbei bekomme ich die Antwort: „Executed request against FHIR RESTful server in 20ms“ und als Statuscode „HTTP 200 OK“. Das bedeutet, dass alle erzeugte Patienten und die dazugehörigen Ressourcen fehlerfrei dem FHIR R4 Standard entsprechen und somit syntaktisch korrekt sind. Andernfalls hätte ich an verschiedenen Stellen der Server-Abfragen Fehlermeldungen angezeigt bekommen.

4.2.8 Weitere Möglichkeiten

Abgesehen von den Änderungen, die gemacht wurden um deutsche Patienten und deren Gesundheitsakten erzeugen zu können, gibt es noch ein paar weitere mögliche Anpassungen die Synthea verbessern oder erweitern können. Diese sind gegebenenfalls für einige Anwendungszwecke interessant und könnten den Realismus weiter steigern, sind aber nicht zwingend notwendig gewesen um das Problem zu lösen und würden über die Ziele hinausgehen. So wäre es beispielsweise möglich bestimmte Behandlungen und Krankheitsverläufe zu überarbeiten falls das notwendig ist. Diese sind in den entsprechenden JSON Modulen im synthea\src\main\resources\modules Ordner vorhanden und bestimmen den Ablauf im Falle des erkranken eines Patienten. Falls für eine Krankheit die Untersuchungen oder Behandlungen in Deutschland stark von denen aus den USA abweichen, kann man das in den entsprechenden Modulen anpassen oder gar komplett neue Module erschaffen. Dafür eignet sich bestens der schon erwähnte Module Builder (12) für Synthea. Dort können auch neue Module für Krankheiten erstellt werde, die es aktuell nicht in Synthea gibt. Auch können in den Modulen die Wahrscheinlichkeiten für den Krankheitsverlauf angepasst oder Medikamente ersetzt werden, wenn in Deutschland andere Präparate gebräuchlicher sind. Da in den USA zum Teil andere Medikamente verschrieben werden oder diese im

deutschsprachigen Raum andere Namen haben, sollten vielleicht die Module separat überprüft werden, falls das für den Anwendungszweck erforderlich ist. Weil ich jedoch dahingehen keine medizinischen oder pharmakologischen Kenntnisse besitze, habe ich die Module unberührt gelassen.

Darüber hinaus wird für Krankheiten, Behandlungen, etc. die standardmäßige Kodierung in den Modulen festgeschrieben. Derzeit wird SNOMED-CT für die Krankheiten verwendet. Falls notwendig könnte das an der Stelle auch in die ICD-10 bzw. später ICD-11 überführt werden. Dafür müsste man jedes Modul einzeln anpassen und an den entsprechenden Stellen im JSON Modul den Wert bei „codes“ ändern. Man kann darunter bei dem beschreibenden Attribut "system" statt SNOMED-CT z.B. ICD-10 schreiben und müsste danach den code der Krankheit in den äquivalenten ICD Code ändern. Es wäre auch möglich das ICD Codesystem an der Stelle nur zu ergänzen, sodass immer beide Kodierungen angezeigt werden. Diese Anpassung der Codes wäre aber sehr aufwendig und ist wahrscheinlich in den meisten Fällen nicht notwendig.

Des Weiteren könnte es hilfreich sein innerhalb der erzeugten Dateien neue FHIR Ressourcen zu erstellen. Denn aktuell werden folgenden FHIR Ressourcen unterstützt: AllergyIntolerance, Bundle, CarePlan, Claim, Condition, DiagnosticReport, Encounter, ExplanationOfBenefit, Goal, ImagingStudy, Immunization, Location, MedicationRequest, Observation, Organization, Patient, Practitioner, Procedure (56). Einige weitere Ressourcen können erzeugt werden, aber nur mit FHIR R4 und US Core Implementierung (z.B. Medication oder PractitionerRole). Dabei gibt der US Core Implementation Guide eine Art Mindestanforderung an US-amerikanische Patientendaten im HL7 FHIR Standard vor (ähnlich dem Kerndatensatz der deutschen Medizininformatik Initiative für Deutschland). Für weiterführende Anwendungszwecke wäre es aber sinnvoll weitere FHIR Ressourcen generieren zu können oder einige Ressourcen wie Medication im Allgemeinen generieren zu können und nicht nur beim Einsatz des US Core Profils. Die FHIR Ressourcen werden dabei in den jeweiligen Export Klassen erzeugt, also z.B. in FhirR4.java für den FHIR R4 Standard. Dort werden Syntheas interne Zustände (57) in denen der klinische Sachverhalt abläuft, in die Ausgabe Formate umgewandelt, wie z.B. in eine FHIR konforme JSON-Datei. Das bedeutet, wenn man in einer Datei neue FHIR Ressourcen haben möchte, muss dazu eine entsprechende Funktion in der FhirR4.java Klasse geschrieben werden. Diese kann von Synthea berechnete Daten nutzen und sie in das standardisierte Format der jeweiligen FHIR Ressource schreiben, die am Ende dem FHIR Bundle der generierten Patienten hinzugefügt wird. So könnten unter Anderem Informationen von Syntheas MedicationOrder-Status abgefragt werden um FHIR Ressourcen, wie etwa MedicationStatements, zu erzeugen. Diese müssten dann noch durch die Funktion in das Bundle eingefügt werden und wären so für alle zukünftigen Patienten vorhanden. Auf äquivalente Art und Weise können auch andere neue FHIR Ressourcen ergänzt werden, falls sie benötigt werden.

5 Ergebnisse

Als Ergebnis kann ich als Erstes festhalten, was nach der Recherche bzw. dem Vergleich von Generatoren für strukturierte synthetische Daten aufgefallen ist. Es gibt insgesamt einige vielversprechende Ansätze für Generatoren von realistischen synthetischen Patientendaten, aber wenige fertige Softwarelösungen. Synthea erwies sich dabei als ausgereiftester und praktischster Generator. Die dort erzeugten „Patienten" sind vergleichsweise realistisch und können für verschiedene Anwendungszwecke verwendet werden. Des Weiteren habe ich diverse Vorteile von Synthea aufgezeigt und festgestellt, dass diese Software genau die Art von Daten generieren kann, die derzeit die Forschung in der deutschen Medizininformatik unterstützen kann. Summa summarum gibt es also kaum Werkzeuge die künstliche Patientendaten so einfach, flexibel und gleichzeitig so umfangreich generieren können wie Synthea. Da das Programm aber in den USA entwickelt wurde, sind einige Anpassungen nötig gewesen um daraus gewonnene Daten im deutschsprachigen Raum einsetzen zu können.

Ein Großteil der notwendigen Anpassungen wurden im Rahmen dieser Arbeit umgesetzt. Die Software wurde untersucht, Daten erzeugt und die zu bearbeitenden Stellen ausfindig gemacht. So ließ sich in den Originaldaten erkennen, dass viele Amerikanismen vorhanden sind, die in deutschen Patientendaten unrealistisch wirken würden. Dank Syntheas öffentlichem Quellcode und der guten online Dokumentation, sind einige Stellen direkt wie von den Entwicklern vorgeschlagen angepasst wurden. Als weiteres Ergebnis stehen auch zusätzliche Veränderungen oder Ergänzungen, die von mir durchgeführt oder empfohlen wurden, zur Verfügung.

Erste Einstellungen der Software konnten mithilfe einer Einstellungsdatei konfiguriert werden. Die Datei synthea.properties bietet eine Reihe Einstellungen, mit denen man die erzeugten Daten beeinflussen kann. So werden dort auch die CSV-Dateien mit den Statistiken zugewiesen, welche von mir später angepasst wurden sind. Um deutsche Patienten zu erzeugen, mussten deutsche Städte in der demographics.csv Datei hinzuzufügt werden. Dazu habe ich nach Statistiken recherchiert und die öffentlich verfügbaren Daten entsprechend dem Schema der CSV-Datei eingefügt. Ich habe die Datei also exemplarisch mit Daten der Städte Leipzig und Dresden ergänzt und dargestellt, wie nach dieser Vorgehensweise könnten weitere deutsche Städte hinzugefügt werden (siehe Kapitel 4.2.2). So können dann weiterführend die größten deutschen Städte hinzugefügt werden, um später Patienten aus dem gesamten Bundesgebiet generieren zu können. Ähnliche Ergänzungen wurden auch für die Dateien zipcodes.csv mit den Postleitzahlen und timezones.csv für die Zeitzonen gemacht. Darin habe ich dann ebenfalls Leipzig und Dresden hinzugefügt. Insgesamt konnten dadurch nun neue Städte erstellt werden, aus denen die synthetischen Patienten von Synthea kommen können. Dieses Vorgehen kann auch als Blaupause dienen um weitere Städte zu Ergänzen.

Darüber hinaus benötigt Synthea noch entsprechende Krankenhäuser und Versicherer in der Nähe damit Patienten auch versorgt werden und die erzeugten Dateien hin-

terher nicht fehlerhaft sind. Es wurden dafür die Dateien hospitals.csv, primary_care_facilities.csv und ur-gent_care_facilities.csv aus Synthea bearbeitet. Hierbei habe ich jeweils ein Krankenhaus pro Stadt ergänzt (nämlich die Universitätskliniken aus Leipzig und Dresden). Nach dem Vorgehen in Kapitel 4.2.3 können auch weitere Kliniken ergänzt werden. Außerdem wurde die Datei insurance_companies.csv ergänzt, damit die „Patienten" einen Versicherer haben können. Dort habe ich exemplarisch eine ortsansässige Krankenkasse für Sachsen ergänzt. Nachdem auch diese CSV-Dateien angepasst waren, lagen als Ergebnis nun alle CSV-Dateien in der Form vor, dass Patienten aus den genannten Städten erzeugt werden konnten.

Ein weiteres Ergebnis meiner Anpassungen sind die deutschen Namen, die nun für die Patienten (und für Ärzte und Straßennamen) verwendet werden. Die names.yml Datei wurde angepasst und teilweise der Code der Demographics.java Klasse verändert um die deutschen Adressen realistischer aussehen zu lassen. So werden nun alle in Deutschland generierten Patienten Deutsch als Sprache haben und auch deutsche Namen. Des Weiteren wurde durch Änderungen im Quellcode der LifecycleModule.java und FhirR4.java Klassen der Realismus der generierten Daten gesteigert. Denn jetzt haben die Patienten auch realitätsnahe deutsche Telefonnummern, Adressen, Führerscheinnummern, Sozialversicherungsnummern und Pass- bzw. Ausweisnummern. Außerdem ist die verwendete Währung nun Euro statt Dollar.

Zum Schluss wurde noch die syntaktische Konformität der generierten Dateien überprüft, mit Hilfe eines open-source FHIR Servers. Die fehlerfreien Up- und Downloads auf bzw. von dem Server bestätigen, dass die Daten dem benötigtem Standard entsprechen und korrekt verarbeitet werden können. Nachdem die syntaktische Korrektheit verifiziert ist, habe ich einen Beispieldatensatz mit circa hundert Patienten erzeugt und stelle diesen auf einem GitHub Repository, wo sich auch die veränderten Dateien finden lassen, zur Verfügung: https://github.com/MarcusRudolph/BA_Synthea_DE. Darüber hinaus habe ich eine kurze Übersicht erstellt, welche Änderungen in Synthea wo vorgenommen werden können um zum Beispiel deutsche Patienten zu generieren. Sie befindet sich in tabellarischer Form im Anhang. Diese Tabelle kann neben den Synthea Wiki Seiten als Hilfestellung für zukünftige Anpassungen genutzt werden.

Insgesamt können nach den genannten Anpassungen also beispielhaft realistische deutsche Patienten (samt ihrer Gesundheitsakten) erzeugt und in dem gewünschten Format ausgegeben werden. Entsprechende meiner Vorgehensweise können dann weitere Daten ergänzt werden, um die Variabilität in den entstehenden synthetischen Patientendaten zu erhöhen. Dadurch gibt es nun de facto eine Möglichkeit syntaktisch korrekte, synthetische Datensätze für die Nutzung im medizinischen Umfeld in Deutschland nach Bedarf zu erzeugen. Ergo kann auf diese Weise das eingangs genannte Problem der fehlenden Testdaten gelöst werden.

6 Zusammenfassung

In meiner Bachelorarbeit habe ich untersucht wie synthetische Daten in der Medizin, besonders für den Einsatz im deutschsprachigen Raum, erzeugt werden können. Ein Problem war die Lücke in der Verfügbarkeit hochwertiger Testdaten für die Medizininformatik. Es gab keine Möglichkeit syntaktisch korrekte (Test-)Datensätze für die Nutzung im medizinischen Umfeld (in Deutschland) nach Bedarf zu erzeugen und beliebig zu skalieren. Da Datenschutz besonders im medizinischen Umfeld eine große Rolle spielt, dürfen Datensätze nur anonymisiert weiterverwendet werden. Darunter „leidet" zum Teil die Softwareentwicklung, indem zusätzliche Hindernisse auftreten.

Ein Ausweg kann die Nutzung synthetischer Daten sein, die a priori keinen Datenschutzbestimmungen unterliegen. Um solche Daten zu generieren gibt es einige Ansätze, von denen viele leider nicht für z.B. die Erzeugung strukturierte Patientendaten anwendbar sind oder aber keine fertige Software liefern. Als derzeit beste Alternative hat sich dabei Synthea herausgestellt, da hier medizinische Daten (elektronische Gesundheitsakten) in den benötigten Standards, wie HL7 FHIR, erstellt werden können. Diese Software kann synthetische Patientendaten algorithmisch generieren und die Daten in aktuellen benötigten Formaten (wie beispiels-weise FHIR R4 oder CCDA) ausgeben. Synthea wurde jedoch in den USA entwickelt und ist daher auf das US-amerikanische Gesundheitswesen ausgelegt. Daher musste das Programm für den Einsatz in Deutschland angepasst werden. In dieser Arbeit habe ich daher exemplarisch gezeigt wie solche Änderungen aussehen müssen und welche Daten ergänzt werden können, damit ein späterer vollumfänglicher Umbau der Software möglich ist. Schlussendlich ist es dadurch möglich künstliche Patientendaten zu generieren die den hiesigen Anforderungen entsprechen und realistische Eigenschaften besitzen (wie deutsche Namen, Adressen, etc.). Außerdem werden auch die Formate und Kodierungen erzeugt, die die deutsche Medizininformatik-Initiative (MII) in ihrem Kerndatensatz vorschlägt.

Anschließend wurde überprüft, ob die neu generierten Dateien fehlerfrei verarbeitet werden können. Dazu habe ich mit einem FHIR Testserver verifiziert, dass die Daten syntaktisch korrekt sind und damit in den gewünschten Formaten vorliegen. Die in dieser Arbeit gemachten Anpassungen bieten daher eine Lösung für das Problem der fehlenden Testdaten, da es nun eine Möglichkeit gibt synthetische Datensätze für die Nutzung im medizinischen Umfeld in Deutschland zu erzeugen. Somit können zukünftig künstliche Patientendaten zur Unterstützung der klinischen Forschung in Deutschland beitragen.

7 Diskussion

Im Rahmen dieser Bachelorarbeit wurde also ein Generator für synthetische Daten gefunden und gezeigt wie dieser an das deutsche Gesundheitssystem angepasst wurde. Nachdem ich mich mit der Literatur beschäftigt und nach Generatoren zum Erzeugen realistischer synthetischer Patientendaten recherchiert habe, ist relativ schnell klar geworden, dass Synthea der beste Ansatz zum Lösen des Problems P1 ist. Dazu muss ich sagen, dass ich kommerzielle Software und Dienstleistungen ausgeschlossen habe. Eventuell gibt es weitere Generatoren, die solche Daten erzeugen oder Unternehmen die fertige Datensätze verkaufen, aber diese wurden hier nicht betrachtet. Abgesehen von Synthea habe kaum eine fertige Software gefunden, die synthetische Datensätze flexibel erzeugen kann. Das deckt sich auch mit der Literaturrecherche, denn es gibt einige Ansätze aber keine fertigen Programme dazu, die strukturierte Patientendaten generieren können. Obwohl der Mangel an (Test-)Daten ein offensichtliches Problem in der Forschung ist.

Nachdem ich Anforderungen festgelegt hatte, habe ich dann geschaut wie man Synthea daraufhin anpassen kann. Aufgabe war nicht diese Software in ein deutsches äquivalent zu überführen, sondern exemplarisch zu zeigen wie ein Generator angepasst werden kann. Da die Software Daten bereits in den benötigten Formaten (FHIR, SNOMED, LOINC, etc.) erzeugte, waren viele Änderungen kosmetischer Natur und seltener die Programmlogik betreffend. Weil Synthea aber gut dokumentiert ist und viele Nutzer hat, die wiederum die Software selbst weiterentwickeln, sind zukünftige Anpassungen leicht machbar. So wird Synthea weiterhin eine gute Wahl sein, wenn es um das automatische erzeugen von Testdaten geht (womöglich auch in Deutschland), auch Aufgrund der immer stärkeren Datenschutzbestimmungen. Dafür wäre wohl eine Weiterführung der hier gezeigten Veränderungen nötig, um diesen Generator vollumfänglich anzupassen. Vor allem müssten mehr Städte ergänzt werden, damit ein Datensatz nicht einseitig wirkt. Dennoch können die Ergebnisse dieser Arbeit dazu beitragen, das synthetische Daten in der deutschen Medizininformatik zukünftig stärker eingesetzt werden, weil somit dem Problem mangelnder Datensätze entgegengewirkt werden kann.

Dabei möchte ich aber auch darauf hinweisen, dass es eine relativ schlechte Verfügbarkeit von öffentlichen Daten/Statistiken gab. In den USA gibt es scheinbar eine breite Palette an statistischen Erhebungen für jede Stadt, die dann in Synthea genutzt wurden. Hierzulande waren beispielsweise bestimmte Bevölkerungsverteilungen nicht auffindbar, weil entweder nach dem Schema keine Daten erhoben werden oder diese nicht veröffentlicht wurden sind. Auch für die Bundesrepublik insgesamt gab es nur wenige (für diesen Zweck brauchbare) Statistiken und einige waren nicht aktuell. So könnten von mir genutzten statistischen Daten ungenau bzw. veraltet sein und falls es im Nachhinein neue Statistiken gibt, wäre es sinnvoll diese bei eine weiteren Nutzung zu aktualisieren.

Des Weiteren habe ich bei den Anpassungen auch ein Schwerpunkt der US-amerikanischen Pateinten außer Acht gelassen, nämlich die zahlreichen Veteranen und damit

deren Veteranenleiden. Dafür gibt es im Original Synthea eigene Krankheitsmodule, Krankenhäuser und Behandlungen. Weil Deutschland jedoch sowohl in der absoluten, als auch relativen Häufigkeit weniger Veteranen hat als die USA sind diese Module meiner Meinung nach hier nicht so relevant. Außerdem werden mögliche Veteranenkrankheiten in Deutschland auch in normalen Krankenhäusern behandelt.

7.1 Kritik

Da Synthea die derzeit beste Wahl für die Aufgabe war, möchte ich jedoch noch darauf hinweisen, dass auch diese Software einige Schwachstellen hat. So wäre es zum Beispiel wünschenswert, wenn es so etwas wie eine grafische Benutzeroberfläche oder eine Web-App geben würde. Die Benutzung ist zwar nicht sonderlich kompliziert jedoch würde eine richtige Oberfläche, bei der auch gleich Einstellungen oder Parameter übergeben werden können, die Software noch attraktiver erscheinen lassen. Dies hätte auch zu Folge, dass evtl. mehr Nutzer (auch mit weniger technischen Kenntnissen) Synthea verwenden könnten. Eine Web Anwendung, bei welcher Synthea auf einem Server laufen könnte und man nach einer Anfrage Patientendaten downloaden könnten wäre eine andere mögliche Erweiterung die diverse Vorteile mit sich bringt.

Des Weiteren ist es derzeit leider nicht möglich Patienten aus zufälligen Staaten zu generieren. Der standardmäßig eingestellte Staat ist Massachusetts, wenn kein Parameter für den Ort eingegeben wird stammen die Patienten von dort. Diesen Staat könnte man zwar im Quellcode ändern (Generator.java Klasse aus dem Unterordner synthea/engine) auf einen anderen Staat der USA oder ein deutsches Bundesland. Aber es ist derzeit nicht möglich das dieser komplett zufällig ausgewählt wird, wenn kein Parameter übergeben wird. Noch besser wäre sogar, wenn innerhalb eines Datensatzes verschiedene Herkunftsstaaten möglich wären. Das meint, dass man sich mehrere Pateienten mit einmal erzeugen lässt, von denen einige oder alle aus unterschiedlichen Staaten bzw. Bundesländern kommen. Derzeit sind alle Patienten, die in einem Programmlauf erzeugt werden immer aus demselben Staat respektive Bundesland (wenn auch aus verschiedenen Städten). Falls der Nutzer dies ändern möchte, wäre es schön eine Auswahlmöglichkeit dafür zu haben.

Ein weiterer Kritikpunkt ist, dass derzeit z.B. nicht alle FHIR-Ressourcen unterstützt werden. Diese müssten falls benötigt vom Nutzern selbst einprogrammiert werden, es wäre aber hilfreich, wenn die Entwickler selbst die erzeugbaren Ressourcen aktualisieren bzw. erweitern würden. Damit auch hier ein breiteres Spektrum an Einsatzfeldern abgedeckt werden kann.

7.2 Ausblick

Da die von mir gemachten Änderungen natürlich nicht vollständig sind, sondern nur aufzeigen sollen wie man eine Software wie Synthea umprogrammieren kann, wären einige Erweiterungen denkbar. So könnte unter Anderem wie oben erwähnt der „default-state“ Massachusettes auf Sachsen oder später ein anderes deutsches Land geändert werden, wie zum Beispiel die Hauptstadt Berlin. Zusätzlich könnte eine genauere Unterscheidung zwischen gesetzlicher und privater Krankenversicherung eingeführt werden, damit dies dem deutschen Gesundheitssystem ähnlicher ist. In Zuge dessen könnte auch die Möglichkeit abgeschafft werden, dass ein Patient keine Versicherung hat. Dies ist zwar in den USA möglich aber nicht in Deutschland, aufgrund der Versicherungspflicht.

Darüber hinaus könnte auch die von mir korrigierte Sozialversicherungsnummer verbessert werden. Diese ist zwar im korrekten Format und ähnlich den deutschen Nummern, jedoch werde dieses in der Realität aus personenbezogenen Daten zusammengesetzt. So könnte programmiert werden, dass dort das Geburtsdatum, das entsprechende Behördenkennzeichen oder der Nachnamen des jeweiligen synthetischen Patienten berücksichtigt wird. Ähnliches gilt für Namen und Sprachen, dass alle Patienten in deutschen Krankenhäusern auch Deutsch sprechen und deutsche Namen haben ist nicht komplett realistisch. Daher wäre ein weiterer Ansatzpunkt für zukünftige Arbeiten die Diversifikation von Namen und Herkunftsländern, sowie gegebenenfalls der Sprachen. Denn Patienten mit Migrationshintergrund würden den Realismus innerhalb eines großen Datensatzes sicher steigern.

Eine weiterführende größere Änderung wäre das Programmieren von Java-Tests mit deutschen Städten oder Patienten. Dann könnten aus den genutzten CSV-Dateien die amerikanischen Städte entfernt werden und Synthea wäre komplett auf Deutschland zugeschnitten. Dazu sind aber eine Reihe von Tests zu schreiben, um sicherzustellen, dass die Daten richtig verarbeitet werden und keine Fehler auftreten. Des Weiteren sind in einigen Fällen die Erweiterung um zusätzliche Kodierungen sinnvoll. So wurde zum Beispiel von der MII in ihrem Kerndatensatz vorgeschlagen, auch die ICD-10-GM und der Operationen- und Prozedurenschlüssel (OPS) als Kodierungen zu verwenden. Diese könnten im Hinblick auf zusätzliche Anwendungsmöglichkeiten ergänzt werden.

Als letztes kann ich Vorschlagen, nachdem das „Deutsche Synthea“ weiterentwickelt und um mehrere Städte erweitert wurden ist, es in das Synthea-international GitHub Repository (58) einzubinden. Dort werden internationale Abwandlungen von Synthea gesammelt und stehen allen Nutzern zur Verfügung. Außerdem könnten dort mehrere Personen gemeinsam das Programm weiterentwickeln. Eine Empfehlung für zukünftige Arbeiten ist daher, dieses Programm weiter zu verbessern, um später ein vollständig „deutsches“ Synthea zu haben, dass die aktuelle Forschung bestmöglich unterstützen kann. Vorläufig sind die geänderten Dateien in einem GitHub Repository von mir abgelegt, unter: https://github.com/MarcusRudolph/BA_Synthea_DE.

Literatur

1. Datenschutz-Grundverordnung; 2020 [Stand: 27.07.2020]. Verfügbar unter: https://de.wikipedia.org/w/index.php?oldid=194416633.

2. Bundesbeauftragter für den Datenschutz und die Informationsfreiheit. DSGVO – BDSG: Texte und Erläuterungen. Bonn; 2019.

3. HIPAA Journal. De-identification of Protected Health Information: How to Anonymize PHI; 2017 [Stand: 28.07.2020]. Verfügbar unter: https://www.hipaajournal.com/de-identification-protected-health-information/.

4. Protected health information - Wikipedia; 2020 [Stand: 28.07.2020]. Verfügbar unter: https://en.wikipedia.org/w/index.php?oldid=917859048.

5. Walonoski J, Kramer M, Nichols J, Quina A, Moesel C, Hall D et al. Synthea: An approach, method, and software mechanism for generating synthetic patients and the synthetic electronic health care record. J Am Med Inform Assoc 2017. doi: 10.1093/jamia/ocx079.

6. Chen J, Chun D, Patel M, Chiang E, James J. The validity of synthetic clinical data: a validation study of a leading synthetic data generator (Synthea) using clinical quality measures. BMC Med Inform Decis Mak 2019; 19(1):44. doi: 10.1186/s12911-019-0793-0.

7. Synthetische Daten :: ITWissen.info; 2020 [Stand: 02.08.2020]. Verfügbar unter: https://www.itwissen.info/Synthetische-Daten-synthetic-data.html.

8. Drechsler J, Jentzsch N. Synthetische Daten: Innovationspotential und gesellschaftliche Herausforderungen: Stiftung Neue Verantwortung; 2018. Verfügbar unter: https://www.stiftung-nv.de/de/publikation/synthetische-daten-innovationspotential-und-gesellschaftliche-herausforderungen.

9. Hafner H-P, Lenz R. Erzeugung synthetischer Datensätze mittels Methoden der multiplen Imputation. Berlin; 2010. (KSFE 2010). Verfügbar unter: http://sas-wiki.org/wiki/KSFE_2010#Mathematische_Statistik.

10. https://www.hl7.org/fhir/. Index - FHIR v4.0.1; 2019 [Stand: 28.07.2020]. Verfügbar unter: https://www.hl7.org/fhir/.

11. The MITRE Corporation. Synthetic Patient Generation: Realistic Health Data No Cost, No Restrictions; 2019. Verfügbar unter: https://synthetichealth.github.io/synthea/.

12. Synthea Generic Module Builder; 2020 [Stand: 22.07.2020]. Verfügbar unter: https://synthetichealth.github.io/module-builder/.

13. Über die Initiative | Medizininformatik-Initiative [Stand: 22.07.2020]. Verfügbar unter: https://www.medizininformatik-initiative.de/de/ueber-die-initiative.

14. generatedata.com; 2020 [Stand: 30.06.2020]. Verfügbar unter: http://generatedata.com/.

15. Deutsche Gesellschaft für Medizinische Informatik, Biometrie und Epidemiologie e.V. Medizinische Informatik · Aktivitäten: Deutsche Gesellschaft für Medizinische Informatik, Biometrie und Epidemiologie (GMDS) e.V.; 2020 [Stand: 30.06.2020]. Verfügbar unter: https://www.gmds.de/aktivitaeten/medizinische-informatik/.

16. MPG - nichtamtliches Inhaltsverzeichnis; 2020 [Stand: 30.06.2020]. Verfügbar unter: https://www.gesetze-im-internet.de/mpg/.

17. Verordnung (EU) 2017/745 über Medizinprodukte; 2020 [Stand: 29.07.2020]. Verfügbar unter: https://de.wikipedia.org/w/index.php?title=Verordnung_(EU)_2017/745_über_Medizinprodukte&oldid=200536669.

18. Lehmann TM, Hrsg. Handbuch der medizinischen Informatik. 2., vollst. neu bearb. Aufl. München, Wien: Hanser; 2005.

19. Prof. Dr. Christian Johner. Software als Medizinprodukt – Software as Medical Device. Johner Institut GmbH 21.10.2019 [Stand: 30.06.2020]. Verfügbar unter: https://www.johner-institut.de/blog/regulatory-affairs/software-als-medizinprodukt-definition/.

20. Reinhardt P. Software als Medizinprodukt: Definitionen und Handlungsempfehlungen; 2018 [Stand: 28.07.2020]. Verfügbar unter: https://www.devicemed.de/software-als-medizinprodukt-definitionen-und-handlungsempfehlungen-a-724307/.

21. Parrington N, Roper M. Software-Test: Ziele, Anwendungen, Methoden. Hamburg: McGraw-Hill; 1991. (McGraw-Hill software engineering).

22. Witte F. Testmanagement und Softwaretest: Theoretische Grundlagen und Praktische Umsetzung. 2nd ed. Wiesbaden: Springer Vieweg. in Springer Fachmedien Wiesbaden GmbH; 2019.

23. Albrecht-Zölch J. Testdaten und Testdatenmanagement: Vorgehen, Methoden und Praxis. Heidelberg: Dpunkt.verlag; 2018.

24. Schwartmann, Weiß. Whitepaper zur Pseudonymisierung der Fokusgruppe Datenschutz der Plattform Sicherheit, Schutz und Vertrauen für Gesellschaft und Wirtschaft im Rahmen des Digital-Gipfels 2017: Leitlinien für die rechtssichere Nutzung von Pseudonymisierungslösungen unter Berücksichtigung der Datenschutz-Grundverordnung 2017.

25. ARX Data Anonymization Tool: ToolPool Gesundheitsforschung; 2020 [Stand: 29.07.2020]. Verfügbar unter: https://www.toolpool-gesundheitsforschung.de/produkte/arx.

26. Schmitz N. De-Anonymisierung [Master Seminar]: Ruhr-Universität Bochum; 2010.

27. Narayanan A, Shmatikov V. Robust De-anonymization of Large Sparse Datasets. In: Security and Privacy, 2008. SP 2008. IEEE Symposium on. [Place of publication not identified]: [publisher not identified]; 2008. S. 111–25.

28. Dube K, Gallagher T. Approach and Method for Generating Realistic Synthetic Electronic Healthcare Records for Secondary Use. In: Gibbons J, MacCaull W, Hrsg. Foundations of Health Information Engineering and Systems: Third International Symposium, FHIES 2013, Macau, China, August 21-23, 2013. Revised Selected Papers. Berlin, Heidelberg: Imprint: Springer; 2014. S. 69–86 (Lecture Notes in Computer Science; vol. 8315).

29. Evans RS. Electronic Health Records: Then, Now, and in the Future. Yearb Med Inform 2016; Suppl 1:S48-61. doi: 10.15265/IYS-2016-s006.

30. Goncalves A, Ray P, Soper B, Stevens J, Coyle L, Sales AP. Generation and Evaluation of Synthetic Patient Data. BMC medical research methodology 2020; 20(1). Verfügbar unter: https://pubmed.ncbi.nlm.nih.gov/32381039/.

31. Klimawandel im Gesundheitswesen – Daten-Inseln werden untergehen [Stand: 28.07.2020]. Verfügbar unter: https://www.devicemed.de/klimawandel-im-gesundheitswesen-daten-inseln-werden-untergehen-a-862350/.

32. Die 10 wichtigsten Medizintechnik-Normen: Entstehung, Vorteile, Anwendung [Stand: 28.07.2020]. Verfügbar unter: https://www.devicemed.de/die-10-wichtigsten-medizintechnik-normen-entstehung-vorteile-anwendung-a-701681/.

33. HL7 – Die Familie von Kommunikationsstandards: HL7 Deutschland e.V [Stand: 28.07.2020]. Verfügbar unter: http://hl7.de/themen/hl7-die-familie-von-kommunikationsstandards/.

34. HL7. FHIR – Hl7wiki [Stand: 28.07.2020]. Verfügbar unter: http://wiki.hl7.de/index.php?title=FHIR.

35. Stefan Schulz. Wozu benötigen wir standardisierte Terminologien wie SNOMED CT?: Institut für Medizinische Informatik, Statistik und Dokumentation (Medizinische Universität Graz); 2011. Verfügbar unter: https://www.researchgate.net/publication/256437656_Wozu_benotigen_wir_standardisierte_Terminologien_wie_SNOMED_CT.

36. SNOMED CT: Häufig gestellte Fragen; 2020 [Stand: 29.07.2020]. Verfügbar unter: https://www.medizininformatik-initiative.de/de/snomed-ct-haeufig-gestellte-fragen.

37. DIMDI. LOINC und RELMA [Stand: 28.07.2020]. Verfügbar unter: https://www.dimdi.de/dynamic/de/klassifikationen/weitere-klassifikationen-und-standards/loinc-relma/.

38. DIMDI. ICD-10-GM [Stand: 28.07.2020]. Verfügbar unter: https://www.dimdi.de/dynamic/de/klassifikationen/icd/icd-10-gm/.

39. DIMDI. ICD-11 [Stand: 28.07.2020]. Verfügbar unter: https://www.dimdi.de/dynamic/de/klassifikationen/icd/icd-11/.

40. GitHub: synthetichealth/synthea; 2020 [Stand: 30.06.2020]. Verfügbar unter: https://github.com/synthetichealth/synthea.

41. synthetichealth/synthea: Getting-Started; 2020 [Stand: 27.07.2020]. Verfügbar unter: https://github.com/synthetichealth/synthea/wiki/Getting-Started.

42. synthetichealth/synthea: Module-Framework; 2020 [Stand: 27.07.2020]. Verfügbar unter: https://github.com/synthetichealth/synthea/wiki/Generic-Module-Framework.

43. Synthea - SyntheticMass; 2020 [Stand: 28.07.2020]. Verfügbar unter: https://synthea.mitre.org/.

44. Medizininformatik Initiative - ImplementationGuide - Manteldokument; 2020 [Stand: 05.07.2020]. Verfügbar unter: https://www.tmf-ev.de/MII/FHIR/Manteldokument/MII-Kerndatensatz.html.

45. Medizininformatik-Initiative, Hrsg. MI-I-Kerndatensatz. https://www.medizininformatik-initiative.de/de/kerndatensatz [Stand: 05.07.2020]. Verfügbar unter: https://www.medizininformatik-initiative.de/de/kerndatensatz.

46. PubMed; 2020 [Stand: 06.07.2020]. Verfügbar unter: https://pubmed.ncbi.nlm.nih.gov/.

47. synthetichealth/synthea: Basic-Setup-and-Running; 2020 [Stand: 10.07.2020]. Verfügbar unter: https://github.com/synthetichealth/synthea/wiki/Basic-Setup-and-Running.

48. synthetichealth/synthea: Developer-Setup; 2020 [Stand: 10.07.2020]. Verfügbar unter: https://github.com/synthetichealth/synthea/wiki/Developer-Setup-and-Running.

49. synthetichealth/synthea: Common-Configuration; 2020 [Stand: 13.07.2020]. Verfügbar unter: https://github.com/synthetichealth/synthea/wiki/Common-Configuration.

50. synthetichealth/synthea: Other-Areas; 2020 [Stand: 13.07.2020]. Verfügbar unter: https://github.com/synthetichealth/synthea/wiki/Demographics-for-Other-Areas.

51. Statistik und Zahlen; 2020 [Stand: 14.07.2020]. Verfügbar unter: https://www.leipzig.de/buergerservice-und-verwaltung/unsere-stadt/statistik-und-zahlen/.

52. synthetichealth/synthea: Provider-Data; 2020 [Stand: 16.07.2020]. Verfügbar unter: https://github.com/synthetichealth/synthea/wiki/Provider-Data.

53. Verzeichnis:Deutsch/Namen – Wiktionary; 2020 [Stand: 16.07.2020]. Verfügbar unter: https://de.wiktionary.org/wiki/Verzeichnis:Deutsch/Namen.

54. HAPI FHIR - The Open Source FHIR API for Java; 2020 [Stand: 21.07.2020]. Verfügbar unter: https://hapifhir.io/hapi-fhir/.

55. Command Line Interface (CLI) Tool - HAPI FHIR Documentation; 2020 [Stand: 21.07.2020]. Verfügbar unter: https://hapifhir.io/hapi-fhir/docs/tools/hapi_fhir_cli.html.

56. synthetichealth/synthea: HL7 FHIR; 2020 [Stand: 22.07.2020]. Verfügbar unter: https://github.com/synthetichealth/synthea/wiki/HL7-FHIR.

57. synthetichealth/synthea: Generic-Module-Framework States; 2020 [Stand: 22.07.2020]. Verfügbar unter: https://github.com/synthetichealth/synthea/wiki/Generic-Module-Framework%3A-States.

58. Synthea international [Stand: 24.07.2020]. Verfügbar unter: https://github.com/synthetichealth/synthea-international.

Anhang

Abbildungsverzeichnis

Änderungstabelle

Eine kurze Übersicht, welche Anpassungen in Synthea wo vorgenommen werden können. Ausgenommen sind hierbei Konfigurationen die einfach in der synthea.properties Datei gemacht werden. Im Allgemeinen gibt es eine gute Dokumentation der jeweiligen Dateien bzw. eine genauere Beschreibung der möglichen Änderungen unter: https://github.com/synthetichealth/synthea/wiki

Anpassung von:	***An welcher Stelle:***	***Anmerkung:***
Herkunftsort der Patienten bzw. verfügbaren Städten	demographics.csv, timezones.csv & zipcodes.csv um die gewünschte Stadt ergänzen	Wenn die Dateien komplett ausgetauscht werden, müssen Java-Tests angepasst werden; jede Stadt benötigt mind. ein Krankenhaus und ein Versicherer in der Nähe
Namen	names.yml bearbeiten	Hier werden auch Straßen-Bezeichner angegeben
Krankenhäuser	hospitals.csv, primary_care_facilities.csv & urgent_care_facilities.csv ergänzen	-
Versicherungen	insurance_companies.csv Datei um Versicherung ergänzen	-
Kosten	Dateien im resources\costs Ordner von Synthea anpassen	Auch zum Teil in Synthea.properties anpassbar
Medikamenten	Direkt in den Krankheits-Modulen anpassbar, in denen sie verschrieben werden	Module befinden sich im src\main\resources\modules Ordner von Synthea
Währung	Den entsprechenden String in der FhirR4.java Datei ändern	Entsprechend so auch in anderen Export Klassen, für andere Formate
Telefon-, Führerschein- oder Sozialversicherungsnummer	Die entsprechenden Funktionen in der LifecycleModule.java Klasse anpassen	-
Krankheiten oder Behandlungen	Direkt in den Modulen anpassbar, die jeweilige JSON Datei muss bearbeitet werden	Auch einfach über Syntheas Module Builder möglich
Neuen Ressourcen oder Formate	Müssen einprogrammiert werden durch eine entsprechende Funktion in der FhirR4.java Klasse	Alternativ so auch in anderen Export Klassen; man kann sich an bereits existierenden Funktionen orientieren

Erklärung

Ich versichere, dass ich die vorliegende Arbeit selbstständig und nur unter Verwendung der angegebenen Quellen und Hilfsmittel angefertigt habe, insbesondere sind wörtliche oder sinngemäße Zitate als solche gekennzeichnet. Mir ist bekannt, dass Zuwiderhandlung auch nachträglich zur Aberkennung des Abschlusses führen kann.
Ich versichere, dass das elektronische Exemplar mit den gedruckten Exemplaren übereinstimmt.

Ort:_____ Datum:_____ Unterschrift:_____