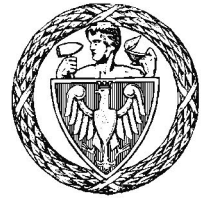# Warsaw University of Technology

FACULTY OF
ELECTRONICS AND INFORMATION TECHNOLOGY

Institute of Computer Science

Bachelor's diploma thesis

In the field of study Computer Science
and specialization Computer Systems and Networks

Structured radiological reporting system

Paweł Paczuski
student record book number 271082

thesis supervisor
Jan J. Mulawka PhD, DSc

WARSAW 2018

Summary

Design and implementation of a system that can be used by radiologists to create structured radiological reports is discussed. The system uses sets of standardized, frequently used phrases to: describe state of patient's body captured by other medical diagnostics methods, provide set of tools that minimize risk of mistake and increase productivity.
Keywords: structured reporting, radiology, health informatics, ontologies, interoperability, SNOMED CT, HL7.

Streszczenie

Rozważono projekt oraz implementację systemu wspomagającego tworzenie strukturalnych opisów radiologicznych. Oprogramowanie ma na celu zwiększenie produktywności lekarza i zmniejszenie ryzyka popełnienia błędu poprzez sugerowanie zbiorów standardowych, najczęściej występujących fraz opisujących stan pacjenta.

Słowa kluczowe: strukturalne opisy radiologiczne, radiologia, informatyka w medycynie, ontologie, interoperacyjność, SNOMED CT, HL7.

Contents

Chapter 1

Introduction

1.1   The need for medical diagnostics

Everyday millions of physicians treat injuries and illnesses of different kids. Before a doctor can plan an individual treatment for a patient, they have to diagnose which organs are in pathological states [1]. This sometimes can be achieved by simply glancing at the body, however, there are many illnesses that require specialized set of tools and methods in order to observe which parts of patient's body are in an unwanted state. Through years of research, many different techniques were established and a separate specialization emerged – radiology. Radiologists focus mainly on analyzing and interpreting diagnostic imagery and as a result of their work they create a document called radiological report which contains description of what can be observed in the image of patient body. Reports may contain description of state of particular organs, measurements (e.g. radius, volume, concentration of certain substances in the blood), comparison of medical condition of a patient observed at different times and description of overall state of the patient.

1.2   Structured Reporting

Currently, the research is mostly focused on finding new ways of diagnosing diseases by the use of more advanced equipment or brilliant algorithms that try to automate image analysis [2].

On the other hand, there exist initiatives that try to improve quality of the radiological reports themselves. There are groups consisting of both computer scientists and physicians that try to standardize reports, prepare checklists that require doctors to describe patients' state in particular order and create a set of phrases that will be understood in the same way by all physicians [3]. A lot of work has been done to provide both common medical nomenclature for medical conditions and theoretical framework to describe relations between causes and effects of patients' condition. As there are more and more methods used to diagnose, the amount of information captured increases, so the reporting methodology has to be kept up to date with the state of art. This is why a very specific field – Structured Reporting (SR) emerged. The basic idea is to provide a way to create radiological reports that conveys as much semantics as possible in a way that is easy to understand. One can find great ideas implemented in such standards as SNOMED SR [4] and also HL7 version 3 Clinical Document Architecture (HL7 V3 CDA). By using these standards, one can encode relations between organs and diseases (causality) in a very regular format. After encoding struc-

ture in the report, one can use text-processing algorithms to e.g. highlight what changed since last visit, look for diseases that were diagnosed in the specified time range etc. This is very difficult to achieve when reports are stored in plain text. Structured reporting focuses mainly on encoding meaning – the visual representation of resulting reports is a separate matter that is treated as an implementation detail [4]. In spite of the existence of these standards, it is almost impossible to find software that implements structured reporting techniques. One of the most important reasons is the fact that in order to understand benefits of SR, one has to acquire certain level of understanding of the typical work-flow of a radiologist. As there is huge demand for software that is much easier to understand (e.g. RIS and HIS software), most of the effort is made to implement simpler concepts.

1.3   Typical work-flow of a radiologist in Poland

In order to find places where optimization of productivity could be applied, one has to get to know how a radiologist works and what are activities that waste significant amounts of time.
In a medium-sized clinic, medical imagery is captured by a radiologic technologist who then uploads the data to the Picture Archiving and Communication System (PACS) and attaches identification information to the images. Next to the PACS system in most cases exists Radiology Information System (RIS) that is used by radiological staff to keep track of patients treated in the clinic. These systems are used to distribute imagery to the team of radiologists.
    Imagery can be distributed in one of the following manners:

- Particular patient is always serviced by the same radiologist.

- RIS system acts as an accumulator of service requests and radiologist decides which patient they should focus on now.

- Certain types of medical examinations are always assigned to a radiologist that is specialized in describing them.

    After receiving diagnostic imagery, a radiologist uses special software called Viewer[8] to navigate through images, make measurements by using visual tools like virtual ruler and examine what is the state of patient's body. Figure 1.1 shows an example of the Viewer software. In parallel to this, the doctor uses text editor and describes what he or she sees in the images. Some RIS systems are equipped with a simple text editors that allow doctors to write the report inside the system without using external editors.

1.4   Discussion about presented work-flow

1.4.1   The good parts

Viewer software provides expected functionality

Viewer software has a very stable position on the market and is perfectly tailored to the needs of a radiologist. It often uses advanced techniques of computer graphics to present patient's body as accurately as possible.
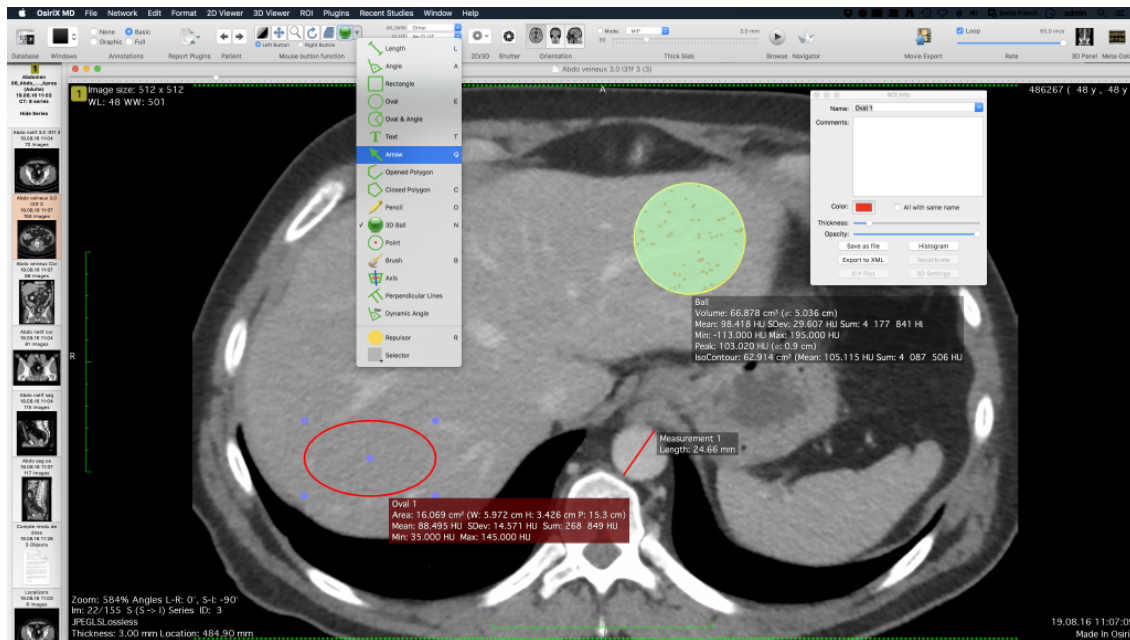
Figure 1.1: OsiriX is one of the most popular image viewers used by the radiologists

PACS systems provide centralized place for storing images

They implement functionalities that allow for archiving medical imagery. Lifetime of medical images is controlled by the government and these systems must take this into account. PACS also make it easy to distribute images not only withing local networks but also to any doctor who has the remote access granted.

RIS systems make it easy to exchange data between physicians

There is no need need for the radiologist to deliver the radiological report to the other physician personally as it is done automatically. Also, RIS systems provide good means of presenting medical records from different medical specializations allowing for interdisciplinary diagnostics.

1.4.2   The bad parts

Focus on text rather than semantics

It is expected that the radiologist will produce a consistent, ordered report by means of text editors. Some RIS systems provide basic text formatting functionality like italicization, underlining but they are limited to the textual presentation of the report. There are no dedicated tools to encode relations in this representation.

Each radiologist has their own style of writing

Usually there are no structural expectations about the resulting report. Each radiologist can have their own style of writing, order of organs observation, text formatting. This leads to the waste of time as people who read the report have to make some effort to infer the meaning from plaintext.

Selective description

It is very frequent that radiologists include in the report only things that they consider bad for the patient. This makes the report more goal-oriented but it means that it is useless to get to know the overall state of patient body.

Copy-paste

Radiologists try to solve the problem of typing on their own by creating templates that contain certain pathologies listed and what they do is execution of the commonly called 'copy-paste' method to create report content. Sometimes they do not notice parts of copied text that are different from the actual state of the body, so the reports may contain observations that are false.

1.5   Other ways to create radiological reports

In many English-speaking countries the work-flow of a radiologist differs in the way the radiological report is generated. A radiologist may record their voice while describing the imagery vocally. The recordings are then transcribed using either speech recognition algorithms or manually by technologists. Having a good skill of typing by a radiologist is not required, only knowledge to interpret imagery is used. This approach, however, has some architectural disadvantages. More personnel are needed – technologists, who transcribe the recorded voice [9]. Also, it is difficult to make changes to what has been said. In some cases, a mistake can be made while transcribing the text [10].

1.6   Problem definition

After analyzing bad parts of the presented work-flow and general situation on the market – demand for radiological services increases but the number of radiologists appears to be constant, it was decided to design and implement a system that would be used by radiologists to encode semantics of diagnostic imagery in a form that can be easily transformed to the human readable form, analyzed by algorithms.

1.7   Incentives for the thesis

A primary goal is to find a solution to the problem of not satisfactory productivity of radiologists by implementing a program used to create structured radiological reports. The system should apply sets of standardized, frequently used phrases to: describe state of patient's body captured by other medical diagnostics methods and should also provide set of tools that minimize risk of mistake and allows radiologists to create reports faster.

Chapter 2

Description of the proposed solution

2.1 General idea

2.1.1 Area of interest

The system that is proposed in this thesis focuses entirely on the part of the typical radiological work-flow in which a radiologist focuses on the textual description of what can be seen in the diagnostic imagery.

2.1.2 Contextual suggestions

Several hundreds of anonymized radiological reports have been analyzed and it was observed that a lot of time could be saved, if a radiologist would at any time select text fragments from a predefined set of possibilities that are appropriate to the current context. This approach can be similar to the way Integrated Development Environment (IDE) for statically typed languages (e.g. Visual Studio supporting C#) are suggesting what the programmer can type based on the namespace, class, scope they currently edit.

In the case of programming languages, the problem is defined in a stricter way as languages are formally defined using grammars [11]. Radiological reports, however, are written using natural language, so there always exist some exceptions. Ideally, a radiologist would always use phrases that are well-defined in the standards, but the the act of scanning long list of possibilities by a radiologist could take a lot of time making it less appealing to the practitioners.

2.2 Assumptions

After observation of actions taken by radiologists while working in environments like hospital, medium-sized clinic, independent teleradiological practice the following assumptions are suggested:

- Radiologists prefer using mouse to keyboard.

- The user interface of the program should be simple.

- Reports must be rendered in a way that allows for copying using clipboard as radiologists developed custom ways to save drafts of reports.

- The system should allow for exporting reports into formatted text and pdf.

- Text formatting should be based on semantics that is encoded by a radiologist.

- The system should favor reports that describe the whole state of body, not only organs in bad condition.

- The system should allow to automatically include in the report segments of text that are (almost) always used (sometimes due to some legal regulations).

2.3   Goals

- Minimize the time radiologist uses keyboard.

- Split report into structural parts that allow to express description of patient's state.

- Any fragment of template text should be editable.

- Maximize number of reports that can be generated by a radiologist in a unit of time.

- At any time allow to include custom phrases that are not defined in the template.

2.4   Proposed reporting ontology based on ideas from DICOM SR and HL7 CDA

In order to provide context for the radiologist at any given moment, it is proposed to split report into the following nested structures:

1. Examinations

2. Organs

3. Properties

Figure 2.1 shows how report structures are related by semantic relations.

By using this ontology, a radiologist can create a report that is similar in structure to a tree (as it is presented in figure 2.2). At any given moment, the doctor modifies the tree at a single level, which allows for suggesting what are the items that can be included. The idea was taken from statically typed languages which, thanks to their strictness, allow for coding with smaller number of mistakes at the lexical level [13]. This ontology was derived after analysis of large set of real-world radiological reports. It mirrors the way a radiologist thinks while creating a report. Because of this, it should be easier to understand the idea behind the proposed system by practitioners.
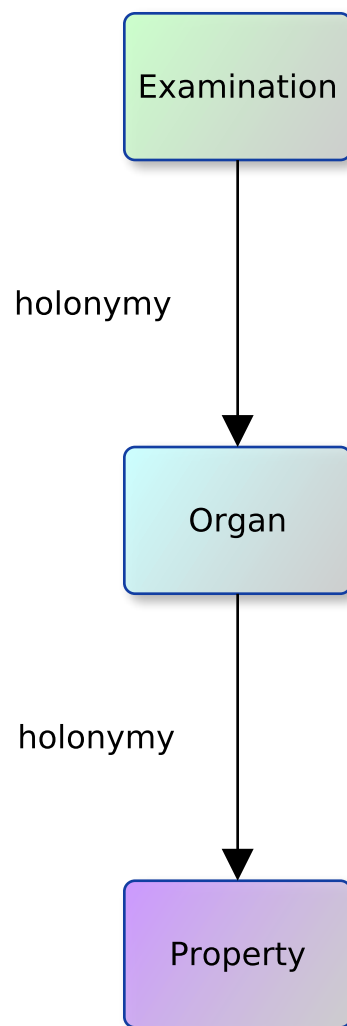
Figure 2.1: Simple ontology representing entities and relationships between them encoded in reports generated using the proposed system
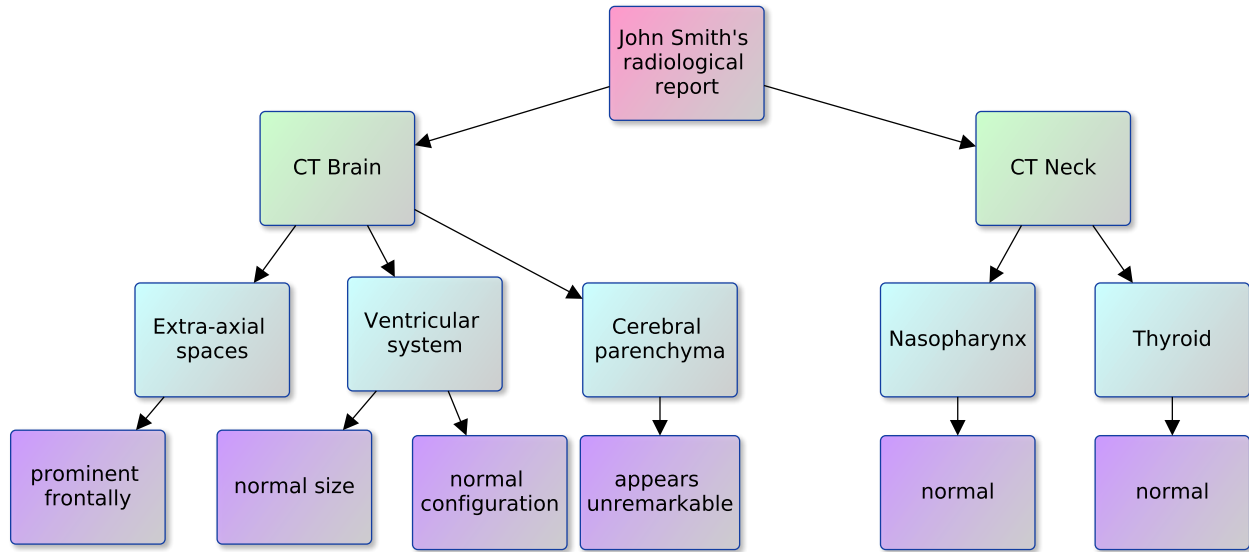
Figure 2.2: Typical structure of a radiological report created using the proposed system[1]

2.5   Work-flow of a radiologist who uses the proposed system

In figure 2.3 a typical work-flow of a radiologist is represented in the form of a flowchart.

2.5.1   Connotation

Besides having text, each property can have a semantical attribute called connotation. Connotation is used to highlight the impact of this property on the result of the organ and examination. This attribute can have one of three values: positive, negative and neutral. Figure 2.4 presents connotation selection box that is present next to the property text editor.

2.5.2   Meta-data

Report may also contain some meta-data which can be used to identify the patient whose body is described in it. Shape of these pieces of information strongly depends on the RIS system used in a clinic. Text segments present in meta-data have no special influence on the semantics of report so it has no detailed description in this work. Figure 2.5 presents user interface used to input commonly used meta-data.

Names for ontological structures shown in Figure 2.1 were chosen so it is easier for radiologists to imagine what they describe, however, they can be used not strictly (e.g. in a CT scan it is possible that a radiologist adds an organ that contains details about intracranial hemorrhage which is a kind of bleeding within the skull [12] but not an organ in its very meaning). This is one of the known weak-points of the proposed ontology.

---

[1]For simplicity, only names of nodes are presented (associated meta-data were omitted)
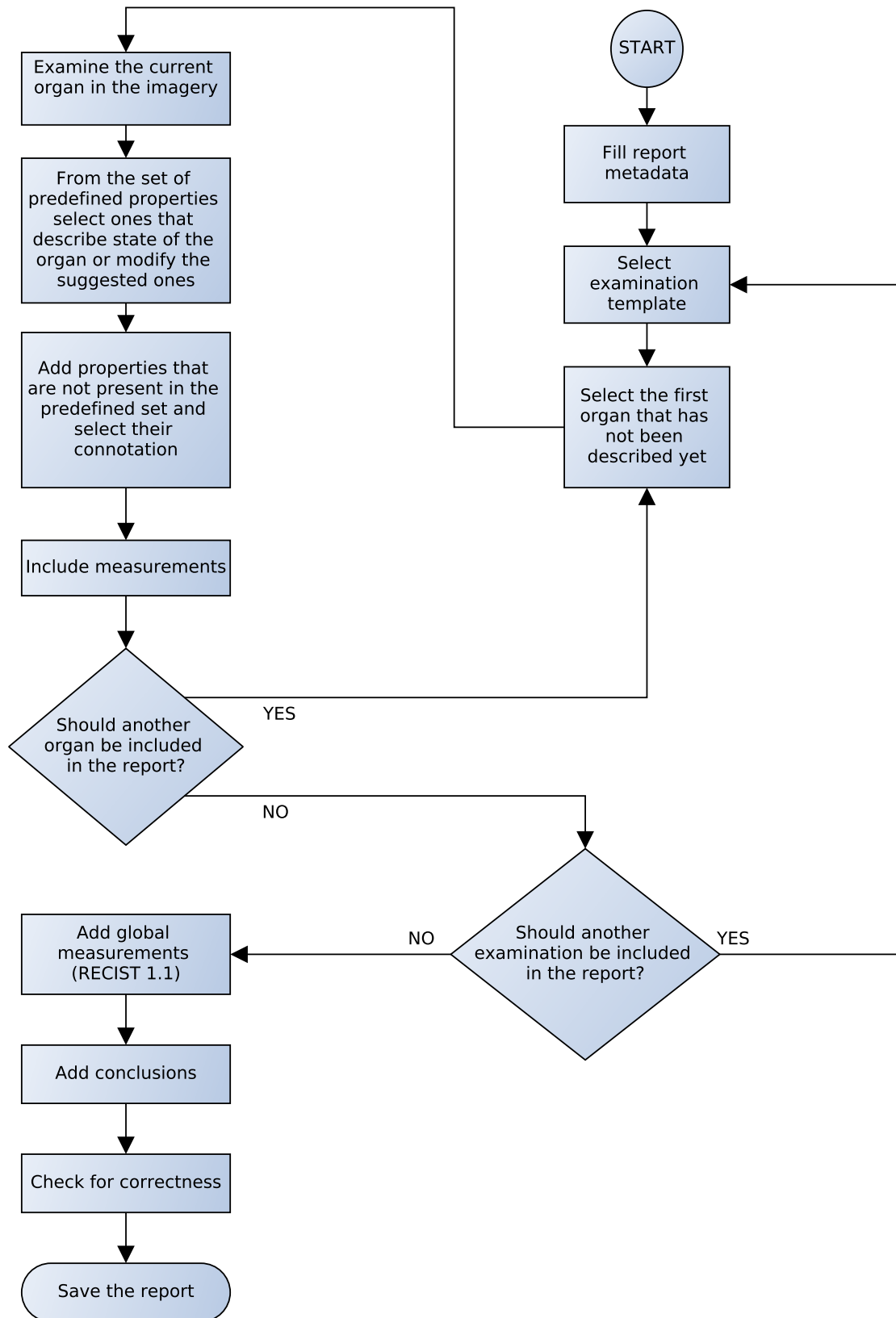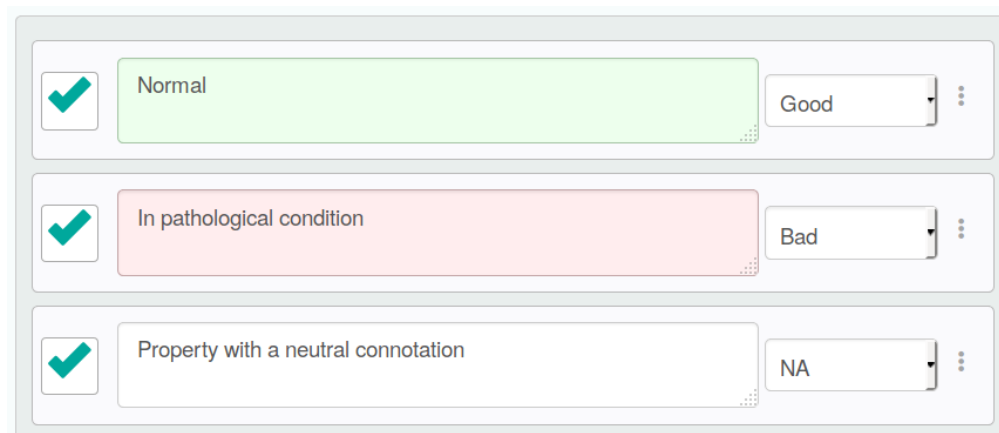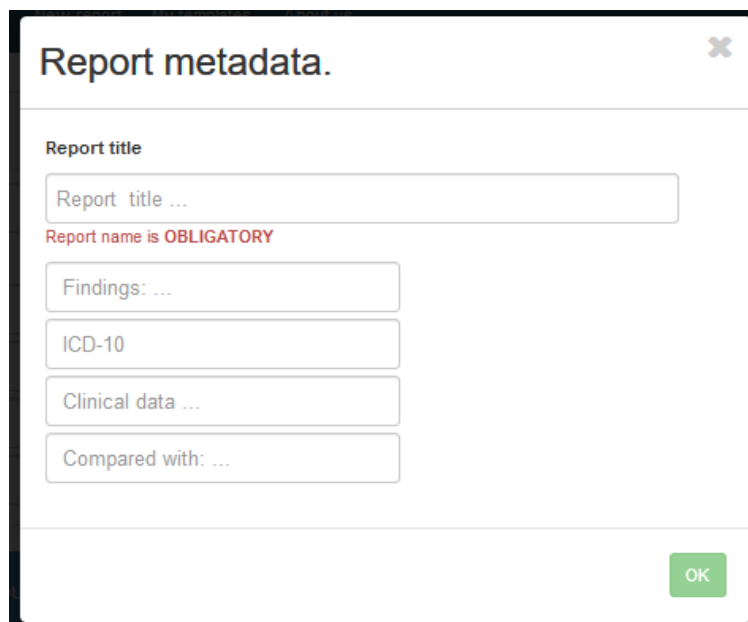
Figure 2.3: Work-flow of a radiologist who uses the proposed system to create contents of a radiological report

Figure 2.4: Connotation can have one of three values: positive, negative and neutral

Figure 2.5: Report meta-data allow user to specify general information about new report
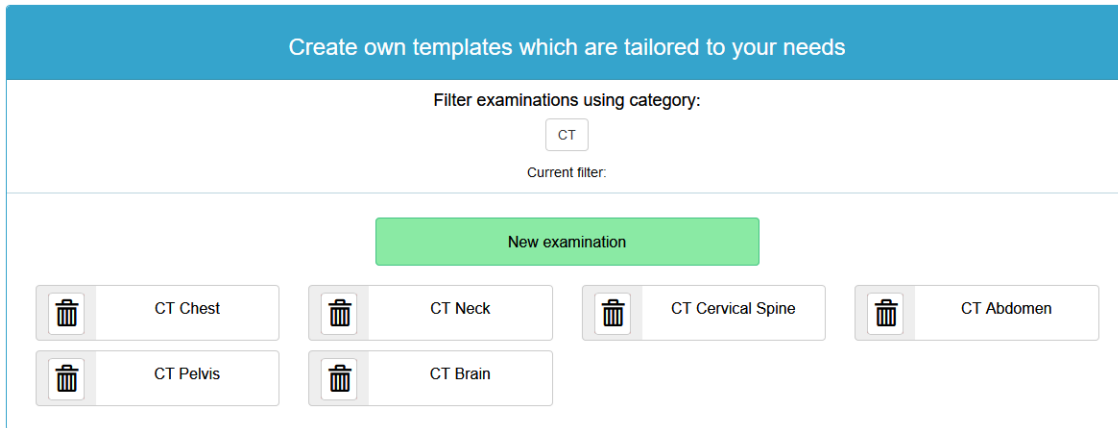
Figure 2.6: List of templates that can be edited by the user
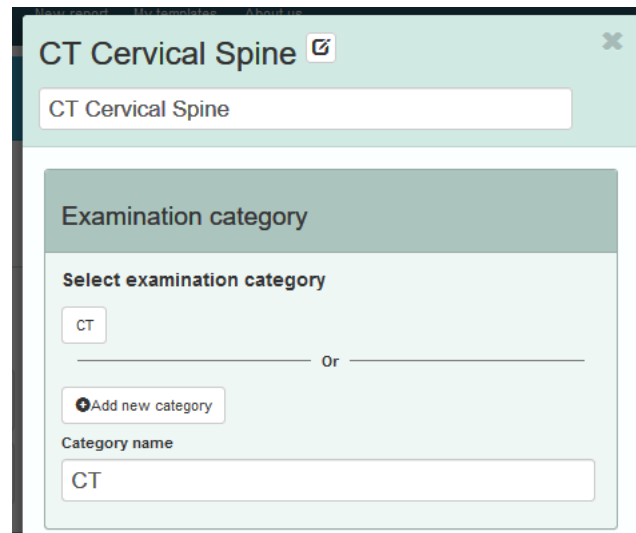
Figure 2.7: GUI used to edit template meta-data

2.6 Examination template

Examination template (or simply a 'template') consists of an ordered list of all organs that could be described when creating a report. It also has some portion of meta-data used for filtering like category, discipline. Templates use the same ontology as the reports as templates can be imagined as reports with all properties, organs included.

Examination templates can be prepared by the leader of radiologists team, so the team can use common language to describe diagnostic imagery. Templates prepared by the leader constitute the so called set of public examinations.

If a given template is not sufficient for a particular radiologist, they can create their own template from scratch or extend the existing one (an action that for people familiar with git version control system can resemble forking a repo [14]).
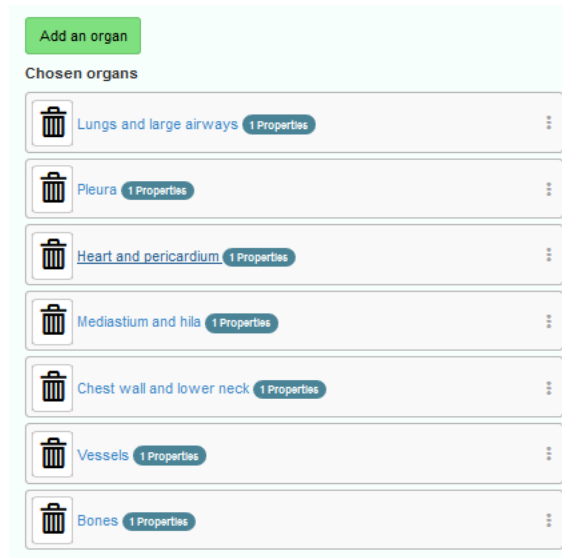
Figure 2.8: GUI used to edit organs
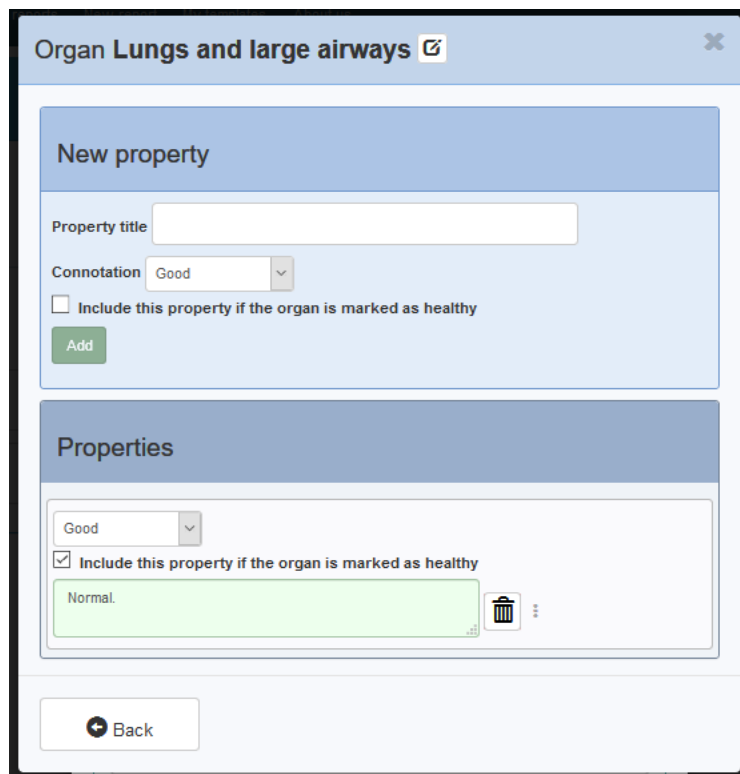
Figure 2.9: GUI used to edit properties of an organ[2]

---

[2]User can predefine connotation and tell the program that this property can be automatically included if the organ is marked as healthy

2.7 Productivity improvements

2.7.1 Using templates

The goal to make radiologists more productive is achieved mostly by decreasing the average time a doctor types on keyboard. Thanks to the usage of examination templates, the time spent to create the schema of the report is shifted from the doctor to the leader of the radiologists team. It can take a lot of time to create a template that is both universal and correct but it is a one-time-only investment. The more templates are created using a template, the more it pays off in the long run.

2.7.2 Mark organ as healthy

In the proposed system, each property has additional attribute (it can be set in template editor) named "include if organ healthy". When a radiologist creates a report, an organ can be marked as healthy, which means that all properties that have this attribute set to true are automatically included in the report. This allows to quickly add properties that are very commonly used. In order to minimize the possibility of inclusion a property that does not correspond to the actual state of patient's body, only properties with positive connotation can have this attribute set.

2.7.3 Calculators

There exist many standardized numerical methods used to asses whether values of certain parameters are in the range suggesting poor health condition, e.g. one of the most popular parameter is the body-mass-index (BMI) which is calculated using formula:

$$BMI = \frac{mass}{height^2} \tag{2.1}$$

After calculating this index, a doctor has to look through tables to check whether the value is usually attached to people with obesity or not. BMI is only one formula out of hundreds of techniques used by doctors. It is not integrated, however, into the proposed system as it was decided that it is not used frequently by end-users.

RECIST 1.1

During the design of the proposed systems several radiologists who specialize in oncological medicine reporting suggested that a method called Response Evaluation Criteria In Solid Tumors (RECIST 1.1) is of special importance to them. This method is used to assess whether tumors in cancer patients improve, stay the same, or worsen during treatment [15]. As there are hundreds of similar methods used by radiologists, it would be impossible to implement them in the proposed system all at once, so it was decided to use RECIST 1.1 as a proof of the concept of calculators that can be used in structured reports. A radiologist, while creating a report, can measure size of lesions and write down the results of measurement in the report. Proposed program should recognize that the numerical value represents a measurement (as shown in figure 2.10) and should suggest whether it should be included in the calculation of RECIST 1.1 or not (as shown in figure 2.11). Measurements are parsed with respect to a convention: a numerical value must be followed by a

Figure 2.10: Measurements data are automatically parsed when user types text into the property textbox

unit (in case of RECIST 1.1 the unit is millimeter – mm), a previous measurement is surrounded with square brackets: '[' and ']'.

2.8   Reporting quality improvements

2.8.1   Standardized nomenclature

There exist several attempts to standardize the language doctors use to describe precisely medical imagery. Templates created for the proposed structured reporting system can be created with the help of terminologists who specialize in systematized nomenclatures contained in SNOMED and LOINC to create reports. This can lead to improved reporting quality as a radiologist does not have to explore huge volumes of precise textual definitions of terms used.

2.8.2   No repetitions

Templates can be interpreted as TODO-lists containing tasks which have to be done while describing medical imagery, so a radiologist can be sure that nothing was omitted or no item appears twice in the report as a result of a mistake. Each organ that has been described is marked as complete, so the user can describe them in order.

2.8.3   Reports have ordered list of elements

Reports created using the proposed system can be represented as trees, nested lists of items, so it is easy to spot relationships between properties and organs and also differences between two reports describing one patient.

2.8.4   Highlighting the most important pieces of information

The most important pieces of information can be highlighted using proper value of connotation. When one of properties of an organ should be interpreted as a main cause of health problems it is marked as a pathology. When the report is then rendered as a document, this property can have e.g. different font color, some text-decoration element or different background color (Figure 2.12).

Figure 2.11: Just before confirming the completeness of the report the user is asked to decide which measurements should be included into RECIST 1.1 calculation

Figure 2.12: Exemplary representation of connotation in the rendered version of the report: positive connotation is represented as green background of the text, negative — red, neutral — transparent

2.8.5   Shift from difference reporting to the current state reporting

There exist practices quite popular among radiologists mostly from smaller clinics that are considered as bad among professionals. One is to include in the radiological report only properties of body which they consider as bad for patient's health. Due to the subjective nature of the report, there were cases when a radiologist skipped a finding that was important stating the patient was healthy. This led to a more difficult and expensive treatment later. There were also cases when a radiologist was unable to compare current state with descriptions of several images that were taken at different times, only with the last one. The doctor stated that there is a progress in treatment, but in comparison to the second from the last description, the health condition worsened [16]. Templates created using the proposed system can favor radiological reports that are more self-contained, it means they describe both bad and good (with respect to connotation) properties of patient's state.

2.8.6   Update of the report

Having a previous report, a radiologist can create another one that will be an update to the former. This procedure is often applied during chemotherapy to see whether tumors react to the treatment. A software functionality that allows the radiologist to introduce small changes to the report without rewriting it, would increase the quality of reporting as the resulting report would be still self-contained.

Chapter 3

Implementation

3.1　Architecture of the proposed solution

The proposed system was implemented as a web application. It is split into two parts: front-end and back-end. Diagram 3.1 presents conceptual overview of the main components of the system and how they interact.

3.2　Technological stack

3.2.1　Back-end

C#

As a main technology that is used to structurize data that is displayed to the user or received from them, ASP.NET C# framework was used. C# is an imperative, strongly-typed, object-oriented language that is used on the market to create enterprise software. There is a hight availability of good quality tooling that makes development in this language faster and less error-prone.

ASP.NET

The ASP.NET application is created using MVC pattern (Model-View-Controller) that is widely known for helping to create applications that consist of loosely coupled, pluggable components. Some interactions with user interface result in HTTP requests that are then mapped by the server to routes that are forwarded to Controllers. Controllers are used to react to user action, instantiate proper classes which implement application logic (they are part of the Model), feed them with sanitized and validated user input and invoke proper methods. When the results of execution are present, Controller finds a View that is expected to be used to present results to the user. View can be treated as a template for the data that is returned from Controller. It can be an ordinary HTML file with placeholders for certain pieces of information (it can also include formatting) or even JSON or XML documents that are only concerned with shape of the data, not the visual representation (e.g. formatting).
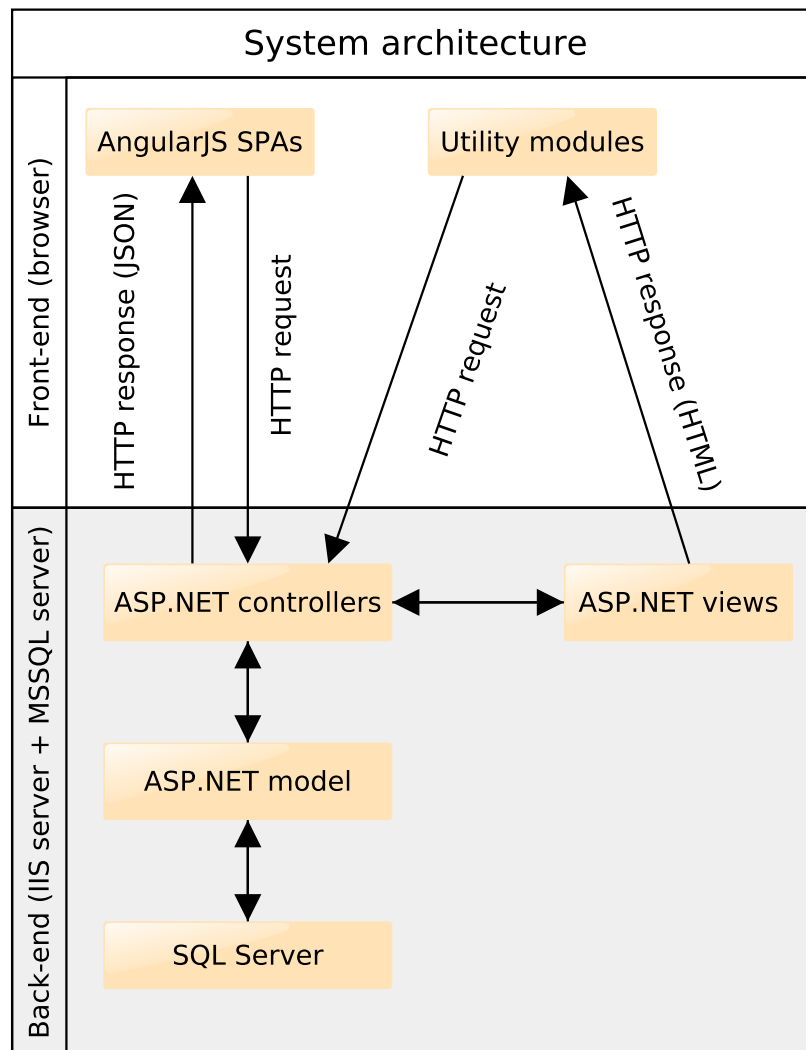
Figure 3.1: System architecture and communication paths

Entity Framework

Entity Framework is a set of libraries that is used to create high-level abstract model using C# that are then converted to sets of relations and attributes in the underlying DBMS. It can also be used to transform LINQ constructions to SQL queries taking into the account vendor-specific features of many DBMS. The approach of first creating C# classes that represent Model and then automatically converting it to the database relations and attributes (scaffolding) is called Code First Approach. It can be especially useful when the schema of database changes very often. When changes are introduced to the code representing Model, Entity Framework can be used create a migration – set of transformations for DB schema that after applying to the DB will keep code and schema synchronized.

Listing 3.1 is used to retrieve number of reports (with of creation of last 30 reports) generated in the last week for all users. What is useful in this approach that all the code of application logic can be written in a single language (in this case C#). This allows for applying tooling from this language to solve problems related to relational data management. After a more thorough analysis of the LINQ syntax, one can see elements of relational ways of thinking present in this mechanism. In the proposed system all database queries are generated using LINQ.

Listing 3.1: C# code that is converted by Entity Framework to SQL sent to SQL Server

```
var manager = new UserManager<ApplicationUser>(new UserStore<
    ApplicationUser>(db));
var prevWeek = DateTime.Now.AddDays(-7);
var usersWithAggregatedInformation = manager.Users.Select(m =>
        new
        {
                Id = m.Id,
                IsConfirmed = m.EmailConfirmed,
                Mail = m.Email,
                ReportsCount = m.Reports.Count,
                ReportsTimes = m.Reports
                        .OrderByDescending(n => n.DateCreated)
                        .Take(30)
                        .Select(n => n.DateCreated),
                WeeklyReportsCount = m.Reports.Where(n => n.DateCreated>
                    prevWeek).Count()
})
.OrderByDescending(o => o.WeeklyReportsCount)
.ToList();
```

3.2.2 Database

MSSQl server is used as a DBMS because it is a battle-tested, well-documented solution that can be easily paired with ASP.NET and Entity Framework's code generation methods are best optimized for this particular DBMS.

3.2.3   Front-end

Not all interactions with the application result in HTTP requests sent to the server, many of them
are about shaping data that has already been received from the server. In order to make use of this
fact main part of the application – tabs: New Report and Templates were developed as a Single
Page Applications (SPA) in order to minimize the time that user has to wait between actions they
take. Technologies used in these parts are JavaScript and AngularJS framework.

Razor views

The remaining front-end subpages are generated using view engine present in ASP.NET called
Razor. These views are responsible for interactions like logging in , registration (backed with
ASP.NET identity mechanism), listing reports, searching for particular report etc. They are gen-
erated on the server side and sent to the user as HTML and rendered by the browser. Listing 3.2
presents the most important part of Razor view code for logging in form.

Listing 3.2: Razor view code filled with C# embeddings that is later converted to pure HTML code
with properly translated strings

```
@model LoginViewModel
@using (Html.BeginForm("Login", "Account", new { ReturnUrl = ViewBag.
    ReturnUrl }, FormMethod.Post, new { @class = "col-lg-12 form-horizontal"
    , role = "form" }))
{    @Html.AntiForgeryToken()
        @Html.ValidationSummary(true, "", new { @class = "text-danger" })
        <div class="form-group login">
                @Html.LabelFor(m => m.Email, new { @class = "control-label"
                    , id = "label_email" })
                <div id="email_login">
                        @Html.TextBoxFor(m => m.Email, new { @class = "form
                            -control", @type = "email" })
                        <div id="danger">
                                @Html.ValidationMessageFor(m => m.Email, ""
                                    , new { @class = "text-danger" })
                        </div>
                </div>
        </div>
        <div class="form-group login">
                @Html.LabelFor(m => m.Password, new { @class = "control-
                    label", id = "label_password" })
                <div id="password_login">
                        @Html.PasswordFor(m => m.Password, new { @class = "
                            form-control" })
                        <div id="danger" class="">
                                @Html.ValidationMessageFor(m => m.Password,
                                    "", new { @class = "text-danger" })
                        </div>
                </div>
        </div>
    </div>
<input type="submit" value="@UP_Medic.Resources.Account.Login.LogMeIn"
    class="btn btn-default" id="button_login" />
}
```

JavaScript

It is a very popular scripting language that can be executed in any popular web browser. Version of the language used in this project is ECMAScript 5.1 compliant.

AngularJS

AngularJS is a JavaScript framework that helps with building MVC SPA aplications. It has several mechanisms like two-way-binding that reduce size of the code needed to implement Create, Read, Update, Delete (CRUD) functionality.

3.3 Internationalization and localization

The application was designed to be accessible by radiologists from many countries, so support for many languages is one of its core functionalities. The application automatically detects user preferred language by looking into user browser settings. Each of the most popular Internet browsers allows users to specify list of preferred languages used by web-pages. The presented solution scans this list and chooses first language it supports, if there are no languages in the list that are supported, English language is presented to the user by default.

At the back-end there are dictionaries for each supported languages that contain mapping $name \rightarrow value$. In Razor views names are used to specify where particular piece of text should be placed. During HTML generation, the application resolves user language and refers to proper resource file. Figure 3.2 presents contents of a dictionary for ReportWizard for Polish language. Adding support for another language is simply providing values for existing names in the dictionary.

For SPA parts of the program the situation is slightly different as DOM elements representing most of their content are generated at the user side with the help of AngularJS. In order to have single source of truth about translations, it was decided to serialize contents of dictionaries to JSON format, and send it to the user with page template. At the front-end the translation dictionary entries are deserialized and made available to JavaScript functions. This solution makes it faster to edit and version translations as they are kept in a single place. One of drawbacks can be the performance of this solution, but it was measured that for number of dictionary entries used in this application, the decrease in performance is negligible. The code that is responsible for serialization of resource files is presented in listing 3.3.

Name	Value
NameOfTheContrastAgent	Nazwa środka
NextStep	Dalej
Node	Węzeł
Obligatory	Obowiązkowy
OrganName	Nazwa organu
Organs	Organy
OrganSelection	Wybór organu do opisu
OrganTip	Kliknij, aby otworzyć okno wyboru organów
Preview	Podgląd opisu
PreviousReportContent	Treść wykonanego wcześniej opisu
PreviousStep	Wstecz
Properties	Właściwości organu
PropSelection	Wybór właściwości organu, które zostaną zawarte w opisie. Możliwa jest zmiana treści właściwości.
Quantity	w ilości
RecistTip	Należy wpisać wartość w nawiasie kwadratowym np. [12mm]

Figure 3.2: Dictionary for Report creator for Polish language

Listing 3.3: Method Translanslation(string lang) serializes all values in a dictionary for a given language and returns a string containing data encoded using JSON format

```csharp
public string Translation(string lang)
{
        var resourceObject = new JObject();
        var rm = new ResourceManager(typeof(Resources.ReportWizard.Index));
        var resourceSet = rm.GetResourceSet(new CultureInfo(lang), true,
            true);
        IDictionaryEnumerator enumerator = resourceSet.GetEnumerator();
        while (enumerator.MoveNext())
        {
                resourceObject.Add(enumerator.Key.ToString(), enumerator.
                    Value.ToString());
        }
        return resourceObject.ToString(Formatting.None);
}
```

3.4   Templates

Templates can be defined by the user at the examination level. Users can create, edit or remove them. In order to keep things simple and less error-prone, they are treated as immutable in the database. If user edits a template, the previous version of the template is deleted and template version with edits is inserted into database. The only thing that is not changed in this process is its primary key. This makes reasoning about database queries much easier and has satisfactory performance for templates used by most radiologists. Thanks to the use of database transactions, one can be sure that removing contents of an existing template and inserting contents of a template with modifications is performed atomically.

3.4.1 Feeding front-end with data from database

After browser finishes loading static files (scripts, fonts, etc.), an additional HTTP request is created to download report templates attached to the current account. Server handles the request by querying database, instantiating model classes that correspond to the data received from database. Finally, server serializes reports data to the JSON format and sends it to the front-end. When front-end receives JSON data, a success callback is invoked and sets the data as current view model. From now on a user can edit any existing template: add properties, organs, remove them, change connotations, reorder elements. Thanks to the two-way-binding mechanism, user interface can be directly connected with references to the fields of an object so the developer can focus on the visual representation of the editor. In order to make editor more robust, length of text contained in it is measured on each onChange event and its height is adjusted, so all its contents are visible to the user.

3.4.2 New Examination

User can create new template by clicking on New examination button. Then they are asked to input title of new examination. One of requested features was the ability to copy contents of existing examinations what is similar to forking in git. In order to achieve this, when modal window with new examination's title is opened, front-end loads from server list of user-owned examinations concatenated with list of publicly-available examinations. The latter allows users to quickly customize templates prepared by the administrator.

3.4.3 Edit examinations

On page load all templates that are owned by the user are downloaded. Any modification can be applied: renaming, reordering of elements, setting connotation, etc. When user clicks Save button, actions related to the immutability of templates described in section 3.4.1 are performed.

3.5 Report creator

3.5.1 Data loaded lazily

Report creator is the most optimized part of the application. It is the place where user spends most of their time and it has to respond to any action immediately. It is not easy to predict which examinations will be used in a particular report, so all of available examinations must be ready to be loaded. At the page load, only list of available examinations' titles (user-owned and public examinations) is loaded from the server. When the user clicks on examination's title, the rest of template's content is loaded via HTTP call and ready to be modified and included in the report. Figure 3.3 shows the initial screen of a template editor with titles of templates listed.

---

[3]Initially only a list of names of available templates is presented to the user. Contents of the template are downloaded after clicking on the pill representing a template
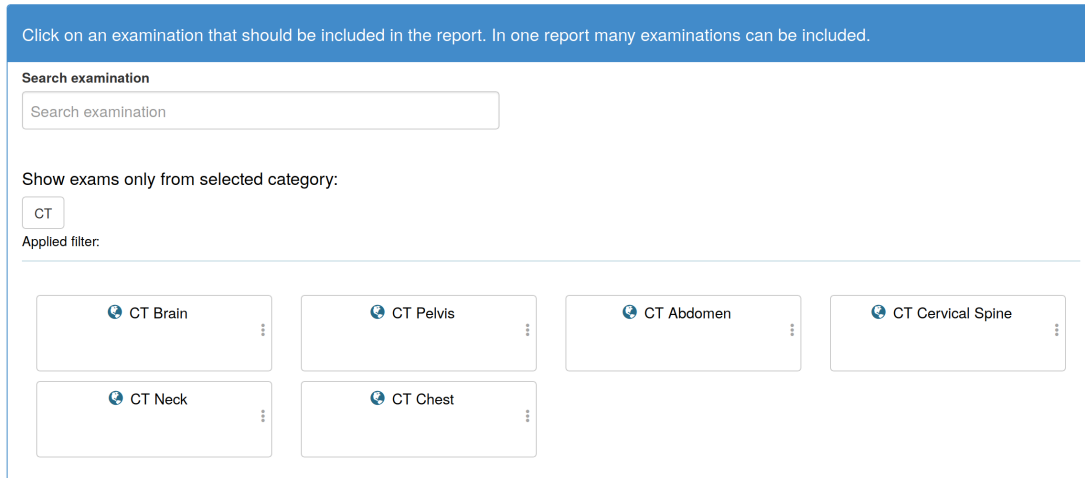
Figure 3.3: A window demonstrating list of examinations to the user[3]

3.5.2   Caching at the user side

When radiologists are specialized in a particular field, they have subset of examinations that they describe most often. Loading examination templates separately each time they create a report, would be a waste of time. In order to solve this issue, a caching mechanism at the user side was designed. After loading, JSON data describing examination elements are saved to the storage managed by the browser called localStorage. It is a dictionary-like storage that can be used from JavaScript to save string values. When user tries to open an examination, the localStorage cache is checked whether this examination is present. If the examination is in the cache, it is immediately shown to the user. Otherwise, an HTTP request to the API is issued to download data.

Cache invalidation

As templates contain no sensitive information, they can be stored in localStorage without restrictions regarding synchronization of cache invalidation with session expiry time. However, the invalidation has to happen when user updates an examination in the Templates tab. In order to keep things simple, the whole cache gets purged anytime user navigates to the Templates tab.

3.5.3   Editing functionality

As user navigates through the levels of the report structure, the program signifies this by opening modal windows one on top of another. This results in the following behavior: when user makes decisions that have the biggest impact on the shape of the report – edits report meta-data, selects examinations that will be used – the program is at the 1 st level of depth, no modals are present. When user decides to include an examination, a modal is opened hiding all information that is not necessary at the moment and the user is shown list of organs with preview of the report (As shown in figure 3.4). Work-flow is designed to favor selecting Examinations/Organs/Properties from predefined sets. It was measured that with suggested use of the templates on average 72% of report's content can be generated from template contents. As most of the content can be added to the

Figure 3.4: List of organs defined in a CT Brain examination template

report by simply clicking on a check-boxes, a lot of thought went into designing interaction model that allows radiologists to look at sequence of suggested elements and make a binary decision 'yes' or 'no' whether an element should be included in the report or not.

3.5.4   Predefined but modifiable

In some situations edits must be made to the predefined pieces of text. This was solved by putting any report text in a slightly customized text-area that is attached to AngularJS context. The text-area is aware of the length of text it contains and automatically fits its height to show the whole text to the user at any time.

Most of the elements can be dragged using mouse. This can be useful in situations, when a custom phrase is very important to the currently edited entity, as it can be positioned as first among other of its type.

3.5.5   Live preview

It is crucial for the user to always be aware of the impact of edits they make to the report contents. Live preview mechanism was designed specifically to solve this. Any significant change of context or report contents sends an event to the function that filters elements that were selected and generates report's textual representation. As this function can take a lot of time to execute, it cannot be left to AngularJS to decide when it should be called (infamous $digest cycle [17]) as it would have a severe impact on performance. AngularJS internally observes arrays of data that are attached to its scope to look for changes, so it was decided to call this function only in several situations, e.g. when report-context is switched, a property is checked etc. but not on every single text-change of the report. This allows the user to have a seemingly live preview of the resulting report while focusing on selecting predefined pieces of text.

3.5.6   Changes propagation

Connotation it attached to each property, however, it is also useful to see its effect on the organ. This way, it is easier to see which organs are in pathological condition, which examinations contain unhealthy organs. Status of an organ is calculated by accumulating connotation of all its properties: if one or more properties have negative connotations, the whole organ is marked as in unwanted condition. On the other hand, the organ is marked as healthy if all properties have neutral or positive connotation. The same holds for examinations, but states of all its organs are taken into account. Propagation of the effects of connotation is done when an organ is closed. Then the accumulated connotation is calculated at the organ and examination level.

3.5.7   Storing a report in the database

This is a very important decision from the architectural perspective. Format of the saved report can impact the performance of retrieving saved report, interoperability with other systems, etc. The format of stored report could be a topic for a very broad discussion. In order to maintain the simplicity of the whole project, it was decided to store JSON representing the report in the database. Many of the DMBS have special functionalities that are based on ideas used in NoSQL databases,

especially document databases to allow for storing JSON objects, validating and querying them as if they ware part of the relational data. This makes it easy to store data that can have numerous optional fields. Unfortunately, MSSQL Server does not support storing JSON objects natively (Microsoft calls their limited way of supporting this format as *built-in JSON support* [18]), however, it is possible to store JSON string representation in a column of type NVARCHAR and then validate it on SQL INSERT by the use of ISJSON function.

3.6   Report history

Report history is a part of the program that is used to list saved reports, search for particular ones, request pdf version of reports, sending report content via email. Users can invoke on existing reports such important actions as editing them, deleting them, comparing them. As one user may have created thousands of reports using this program, this part of the program paginates reports. There are 10 reports per page and user can increment index of the page they are currently viewing or decrement it. One useful feature is grouping reports by title. This is useful when user has the convention of naming that report's title is always name of the patient (most of the current users of the program do it in this way) as they can see and count reports for any user organized in groups.

3.7   Compare reports

This feature strongly depends on the mechanisms included in the Report creator. It copies meta-data of an existing report and creates a new report using it. Content of the old report is presented to the radiologist as a reference and it is also copied as a basis of a new report. When the doctor makes some changes to the contents of the report, only the new version is modified. This allows for fast creation of a report that has only small portion of fragments modified when compared to the previous one. Changes propagation functionality described in 3.5.6 is particularly useful when comparing two reports, as user can easily spot differences between the reports and include their observations as conclusions.

Figure 3.5 presents user interface that makes it easy for the radiologist to compare two reports.

3.8   Integration with existing systems

Many radiologists work remotely for several companies that have different RIS systems. From the practical point of view, it is very difficult to integrate the system with all of the systems as it often requires cooperation of both developer of the proposed system and the developers of the RIS system.

Current approach to integration is very robust, works for all of the systems used nowadays but is not very elegant from the point of view of software engineering. After generating the report using the proposed system, it is the radiologist's duty to manually copy the resulting report to the RIS system. It is simplified in a way that the report is automatically copied to the clipboard after clicking on its content.

On the other hand, the proposed system was designed in a way that, at least theoretically, makes it easy to create communication channels that can be used to integrate this system with

Previous report content:

John Smith

CT Brain
Extra-axial spaces: Normal .
Intracranial hemorrhage: None .
Basal cisterns: Normal .

CT Pelvis
Reproductive organs: No pelvic masses .
Ureters: Normal .
Bladder: Normal .
Bowel: Normal caliber .
Peritoneum: No ascites or free air; no fluid collection .
Vessels: Atherosclerotic changes .
Bones: Normal .

CT Chest
Lungs and large airways: 3mm[4mm] ; 1mm[3mm] .

Conclusions: Further observation recommended

Sum of previous measurements: 7
Sum of current measurements: 4
Difference (absolute): -3
Difference (relatively): -42.857%
Status: Complete response

Report content:

John Smith

CT Brain
Extra-axial spaces: Normal .
Intracranial hemorrhage: None .
Basal cisterns: Normal .

CT Pelvis
Reproductive organs: No pelvic masses .
Ureters: Normal .
Bladder: Normal .
Bowel: Normal caliber .
Peritoneum: No ascites or free air; no fluid collection .
Vessels: Atherosclerotic changes .
Bones: Normal .

CT Abdomen
Bile ducts: Normal .

CT Chest
Lungs and large airways: 3mm[4mm] ; 1mm[3mm] .

In order to copy the report, it has to be confirmed

Conclusions

RECIST 1.1

Figure 3.5: Compare reports mode presents simultaneously contents of the currently edited report and the old one

existing RIS systems. This could be achieved by using Razor templating to generate messages that would be HL7-compliant. Even though the very idea behind HL7 was to make it a universal way to exchange information between systems, there are 3 versions of the standard that are not backward compatible. Moreover, each software vendor can interpret the protocol in different way, expecting the exchanged data in unique shape. Because of this, integrations most often have to be developed separately for every pair of systems that take part in the information exchange.

3.9 Responsive design

The system can be used on any operating system as its execution environment is provided by the web browser. For well-crafted templates, majority of report contents can be generated by simply clicking check-boxes. If the interaction model is this simple, it can be convenient to generate the report on touch devices like tablets or smartphones. However, this adds the complexity to the user interface design, as human fingers are not as accurate as mouse pointer. This problem was solved by designing responsive user interface for the program with help of Bootstrap 3 library. User interface is treated as a grid that has cells. Cells have size that depends on the size of the viewport. This allows to define how the user interface changes as the size of the devices gets smaller. There are several modifications applied to the user interface when the program is executed on small and medium-sized devices:

- Check-boxes are bigger, so it is easier to tap them using finger.

- Tables have smaller number of columns in order to fit the screen.

- Navigation bar at the top of the page is replaced with a button that expands all menu items on a click.

- Modal windows are occupying the whole screen, so more content can be shown to the user.

- Lists that usually are represented as a grid, are enjambed vertically one item per line, so it is easier to tap them using finger.

- Font sizes are modified with respect to the Bootstrap's typography recommendations. [19].

The system could be used on mobile devices in situations when a radiologist is not only responsible for report but also for the technical execution of the examination. After capturing the imagery, the doctor would use tablet to generate the report, present it to the patient and send to the RIS.

3.10 Utility modules

The system was designed as a solution that could be used remotely without personal help of the creator. Because of this, several modules that would make it easier to start working with the program were developed. Modules are implemented as separate controllers and the separation is made visible to the user using distinct routes in the web address.

3.10.1   Technical support module

This part of the program is used to communicate between radiologists and people responsible for correct operation of the system. Information Technology (IT) staff can be informed about any problems that occurred while using the system. Each issue is represented as a separate topic. In a particular topic the user and IT staff can exchange messages asynchronously. Furthermore, topics are split into categories, such as: mobile devices issues, reporting problems, template problems, linguistic suggestions and also have state, e.g. new, processing, closed. State is a way to signal whether the problem was fixed or not. User that created the topic is notified via email when any change to it is made by the staff.

3.10.2   Announcements

When a new feature is implemented, active users can be notified by an announcement that is visible to them after they log in.

3.10.3   Administrator dashboard

Administrator dashboard aggregates most useful actions that can be executed in order to get to know how the system is being used. There are subpages used to check what is reporting rate of a user, how many reports have been created, what are the weekly most active users. Moreover, administrators can see all private templates. This allows to copy best parts of user-defined templates and copy them to the public template repository. This way all users can benefit from knowledge of an individual radiologist.

3.10.4   Make a public copy of a template

This functionality creates a separate instance of a template that has all data used for reporting copied. The user of newly created instance is set to $null$ as this represents a template that is public. Traditionally, one should create a copy constructor for class Examination and then recursively copy all elements of the tree by creating copy constructors for all nodes. The approach taken here is different and very simple – existing template is serialized into JSON and then deserialized back to a C# object. This way all nodes at all levels are copied and detached from their original examination instance. The deserialized object it attached to the database and saved as a publicly available template.

Listing 3.4: Code used to make a public copy of a template

```
var exam = db.Examinations.Find(id);

var serializedJson = await Task.Factory.StartNew(() => JsonConvert.
    SerializeObject(exam, Formatting.Indented,
new JsonSerializerSettings
{
ReferenceLoopHandling = ReferenceLoopHandling.Ignore
}));
```

```
var deserializedExam = (Examination)await Task.Factory.StartNew(() =>
    JsonConvert.DeserializeObject(serializedJson, typeof(Examination)));
deserializedExam.User = null;

db.Examinations.Add(deserializedExam);
db.SaveChanges();
```

3.10.5   User account

This subpage is used to change user password, change email, private data, register a new user, log in to the program.

3.10.6   Interactive guide

When the system is used for the first time on a machine, the user is asked to complete a tutorial how to use the system. The tutorial is in a form of an interactive guide that is embedded in the report creator module. It consists of steps that are describing functionality of the program and may require some interaction, e.g. clicking a button. Section of the program that is currently examined is highlighted, so the user can focus on it. Interactive guide can be closed at any time, so it does not annoy the user.

3.11   Deployment strategy

Deployment of the program is performed with the use of Microsoft Azure's App Services and is initiated after a commit is made. Firstly, the program is built, when there are no compilation errors, it is packaged and deployed to the IIS server. This allows for quick and hassle-free propagation of changes made to the code-base. An error at any stage of the pipeline is signaled either by the Visual Studio IDE, or Microsoft Azure via email. Figure 3.6 presents the sequence of actions and set of subjects that are required to deploy the application.
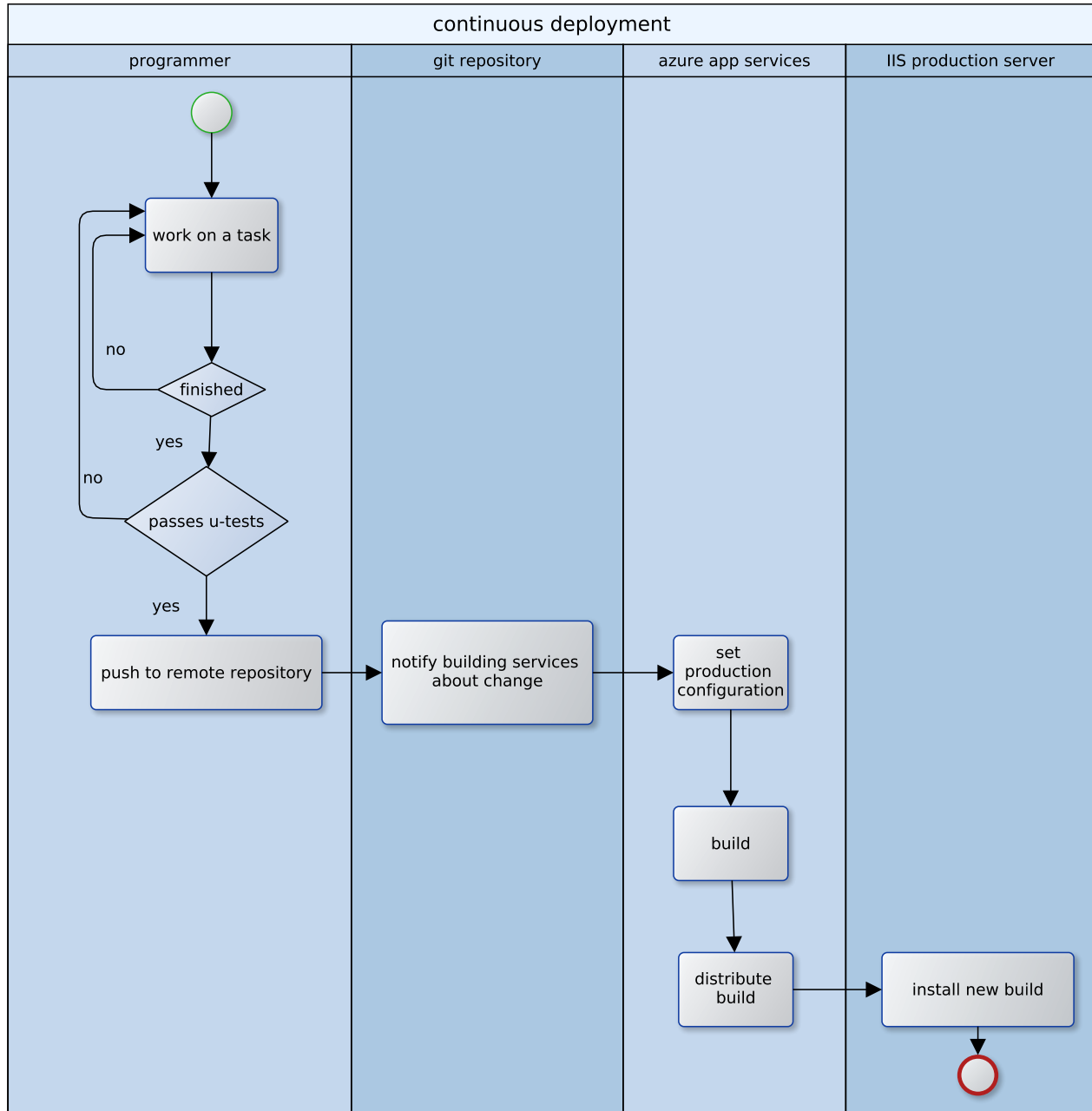
Figure 3.6: Continuous deployment setup makes it faster to distribute new builds to the production servers

Chapter 4

Testing and verification

Most of the testing and verification was performed as user-acceptance testing as the program is heavily focused on interaction with user, however, security-critical parts of the program were unit-tested. The tests were split into several classes that are described in the following sections.

4.1 Entity ownership tests

Any piece of software that is used to support health care requires special measures to be taken to create a product that ensures that patients' privacy is respected and no information is shown to a third party. Entity ownership tests are used in the system to verify mechanisms that filter data, so they see only data that is public or related to their patients.

4.1.1 User sees only their reports

When users navigate to the Report history subpage, they should only see reports created by them. Moreover, manual manipulation of web browser address should also forbid accessing reports that belong to other users – HTTP error 403 (forbidden) is thrown.

4.1.2 Users can edit only their reports

Very similar to the scenario described in 4.1.1 but relates to the report editing functionality.

4.1.3 Users can access contents of public and custom templates

When users navigate to the ReportWizard subpage, they see a list of reports that contains both public and custom templates. They are not allowed do download other user's custom template.

4.2 Database consistency tests

Database consistency tests are used to check whether all possible mechanisms provided by DBMS are applied to ensure that related data are properly treated when an entity is removed, relationships between entities are properly split into relations, etc.

4.2.1   Cascade removal

Deleting a user causes removal of all their reports, templates from the database. This way the system allows the user to be forgotten when it is no longer used. The same should hold for almost all one-to-many relations – removal of the entity that is being pointed to should result in deletion of all entities that point to is, but not the other way round!

4.2.2   Code First schema generation

Schema of the database is automatically generated using Entity Framework Code First approach, so it requires verification whether actual schema is the same as expected. Relations are created with respect to convention of naming fields of C# classes.

4.2.3   Validate JSON

In general, any user input should be treated as a potentially harmful. In case of reports, they are represented using JSON format. Any inserts of reports should check whether the data has all properties of a valid JSON format.

4.3   Report structure tests

Thanks to tests in 4.2.3 it is known, that the shape of data is a valid JSON, but it not clear whether the JSON representation has all required fields, data types, etc.

4.3.1   Validate JSON representation of the report

In this step, JSON representation is validated whether all fields have proper names, values of fields have expected data type, types of nodes are nested as presented in the proposed ontology in 2.4.

4.3.2   Report rendering

The report can be easily transformed from JSON representation to HTML that can be stored in RIS systems. However, if a JSON field has content that is valid HTMl, or even worse, JavaScript script the rendered report content could be harmful to the user because it would execute unwanted code in the browser. In order to make sure, the user is not harmed while using the system, contents of all JSON fields are HTML-escaped.

4.4   Report editor functionality

This class of tests is difficult to automate as they mostly verify user interface. All of them were performed manually as user-acceptance tests.

4.4.1   Live preview refresh rate

The reason for not refreshing report contents on each pressed key was described in 3.5.5. These tests are performed to check, if the performance of the program is satisfactory – they are performed using a report that is considered to be unrealistically large and latency is observed.

4.4.2   Tree changes propagation

Verify whether change of connotation of a property propagates into the organ and examination.

4.5   Internationalization tests

Currently the program has no users that do not speak Polish, so there was no special motivation to automate testing of the internationalization functionality. It was verified manually by user-acceptance tests using web browsers with different locale-settings.

4.5.1   Language specific templates

User that has the web browser language preference to English should see report templates in English, report items and meta-data entries should also be rendered in English. This should work analogically for any other language supported.

Chapter 5

Some examples of system output data and validation

5.1 Reports created using the proposed system

In order to asses the usefulness of the proposed system, one can take a look at the reports created using it. Quality of the resulting reports depends strongly on the level of detail encoded in templates created by the administrator or the user. Presented examples are based on real reports created the proposed system. All pieces of information that could be used to identify patients were removed or replaced with fake ones.

5.1.1 Basic report

The first example is shown in figure 5.1. It is report created in English, using template that was created using nomenclature and elements from several templates available for free at radreport.org. There is no connotation encoding, the number of properties defined for each organ is very small. Each type of entity (examination, organ, property) is presented using different formatting.

5.1.2 Standard report of a healthy patient

Another example is presented in figure 5.2. This report was created based on template in Polish language. As this thesis focuses mostly on the shape of data that is processed using the system, meaning of particular phrases does not matter. Template that was used for this report had several organs with properties having "include if organ healthy" attribute set to true. This allowed the radiologist to create the final report clicking simply on a button "mark as healthy" for each organ. This saved a lot of text editing actions and most of the time was spent on analyzing diagnostic imagery.

5.1.3 Report describing complex pathological state

The lengthiest example is presented in figure 5.3. Is is a report that describes state of an oncological patients. Two examinations were performed and described in a single document in order to keep related data close to each other. Some organs were healthy (all their properties have positive

John Smith

date: 02.04.2005

CT Brain

Extra-axial spaces: prominent frontally.
Ventricular system: normal size; normal configuration.
Cerebral parenchyma: appears unremarkable.

CT Neck

Nasopharynx: normal.
Thyroid: normal.

Figure 5.1: Report containing two examinations based on templates published at www.radreport.org

Jan Kowalski

data: 10.04.2010

USG Jamy brzusznej i przestrzeni zaotrzewnowej (JB)

```
Wątroba: normoechogeniczna, bez zmian ogniskowych, niepowiększona.
Pęcherzyk żółciowy: cienkościenny, bez złogów.
Drogi żółciowe: nieposzerzone.
Trzustka: bez zmian ogniskowych, jednorodna, normoechogeniczna.
Śledziona: bez zmian ogniskowych, niepowiększona.
Nadnercza: niepowiększone.
Obie nerki: bez kamicy, bez zmian ogniskowych, o zachowanym zróżnicowaniu
korowo-rdzeniowym, bez zastoju moczu.
Pęcherz moczowy: gładki w obrysach.
Duże naczynia jamy brzusznej: w normie.
```

Figure 5.2: Exemplary report of a healthy patient based on a template created by user working in Polish language

connotation), but there were organs that had all three types of connotation. This indicates that some parameters of those organs were in bad condition, informing about presence of pathologies of different kinds. The doctor that is about to read this report can focus their attention on properties that are in red as they are the most important to the report findings. Even if combinations of presence of different properties could be unique, most of the content of the report was generated without manually editing properties. There are, however, some properties that contain measurements data – they must have been edited manually.

5.2 Validation

5.2.1 Time savings

The easiest way to validate the proposed productivity improvements is to measure the difference in report turnaround time. Prior to using the software, a group of radiologists was observed to measure

Adam Nowak

data: 10.04.2010

```
Rozpoznanie: CML, guz nerki
```

Tomografia komputerowa klatki piersiowej

```
Technika badania: badanie przed i po podaniu środka kontrastującego.
Płuca: bez zagęszczeń; bez zmian podejrzanych o przerzuty, pojedyncza bulla
rozedmowa w S6PP 20mm.
Węzły chłonne w śródpiersiu: przytchawicze dolne 8mm.
Tchawica i oskrzela: prawidłowo drożne.
Jamy opłucnowe: bez płynu.
Serce i śródpiersie: serce powiększone w całości.
Pień i tętnice płucne: prawidłowo drożne.
Elementy miękkotkankowe ściany klatki piersiowej: guz w sutku prawym 35 x 37mm
- zmiana przemawia za złośliwością; węzły chłonne pachowe prawe kilka do 10mm.
```

Tomografia komputerowa jamy brzusznej i miednicy małej

```
Wątroba: niepowiększona; bez zmian ogniskowych; jednorodna; normodensyjna.
Pęcherzyk żółciowy: cienkościenny; bez uwapnionych złogów.
Drogi żółciowe: nieposzerzone.
Trzustka: prawidłowej wielkości; jednorodna; bez zmian ogniskowych.
Przewód trzustkowy: nieposzerzony.
Śledziona: niezmieniona.
Nadnercza: niepowiększone.
Nerka prawa: z patologicznym guzem 52 x 35mm, zlokalizowanym w części środkowej
kory , podejrzanym o zmianę rozrostową; z torbielą 12mm.
Nerka lewa: położona prawidłowo; bez zastoju moczu; bez uwapnionych złogów; z
obecnością torbieli lub zmiany rozrostowej 8mm - zbyt mała do oceny ( filtr
badania o niskiej rozdzielczości obrazu).
Moczowody: nieposzerzone.
Pęcherz moczowy: gładki w obrysach, dobrze wypełniony.
Narządy rodne: macica mięśniakowata, ze zwapnieniami.
Węzły chłonne: retrokawalny przy naczyniach nerki prawej 25 x 17x40mm.
Jama otrzewnej: bez płynu; bez patologicznych zbiorników płynowych.
Elementy kostne objęte badaniem: stan po złamaniu żebra X przykręgosłupowo;
zaburzenia struktury najpewniej o charakterze zmian naciekowych w kościach
miednicy, nie można wykluczyć drobnych zmian w trzonach - lepsza ocena w Sc;
stan po złamaniu kompresyjnym trzonu L1.

Wnioski: guz piersi prawej, guz nerki prawej z przerzutami do węzłów
zaotrzewnowych, zmiany kostne do oceny w SC.
```

Figure 5.3: Exemplary report (with two examinations and connotations marked) of an unhealthy patient based on a template created by user working in Polish language

the time it takes to create a report. The time was measured for each of the most popular examinations e.g, Abdomen Ultrasonography, Brain Computed Tomography, etc. for reports that were created using traditional methods. Then, the radiologist was allowed to use the software for several months, create their own templates. After some time for adjustments and getting accustomed to the proposed work-flow, the time to create a report was measured one again. The biggest time savings were observed for patients that were healthy or their symptoms were frequently observed. However, significant increase in productivity was also measured for reports containing unusual and complex observations. Overall, doctors who used the program with battle-tested templates and figured out what the strengths of the solution are, benefited from using the system by decreasing report turnaround time to a third of the time required to create a report without the use of the proposed system.

5.2.2 User satisfaction surveys

Over the time of first few months after the release of the software to the users, several surveys were conducted. The users were asked open-ended questions such as: "What is the most important feature that you would like to use everyday?", "What is your overall experience you have when you are interacting with the program?", with addition to scale questions like "How likely would you recommend the software to your colleague? 1 – very unlikely, 5 – very likely", "Using the software as my main text-editing tool makes me more productive. 1 – strongly disagree, 5 – strongly agree".

It could be observed that users who initially assigned good marks to user experience, were keener to suggest further improvements. Moreover, quick reactions to the suggestions, tended to result in better marks in the subsequent surveys. On the other hand, users that were not interested in using additional tools to be more productive, were very difficult to encourage to try using the software for a longer period of time. The best results were achieved when a director of radiology observed benefits of using the program and taught their team how to use the system.

5.2.3 Users from several working environments

Teleradiologists

Structured reporting system is being used by several independent teleradiologists in Poland as their main text editing tool. They receive reporting requests with images they are asked to describe. They use several RIS systems from different vendors to store reports they generate. Most of them simply copy and paste reports from the system to the RIS systems they are obliged to use. As majority of the teleradiologists who have used this software are paid on the per-report basis and the average time to create a report using this program is on average three times shorter, most of the users observed significant increases in their salary.

The system is deployed to Microsoft Azure IIS server and is globally accessible, which makes the updates easier to install and new users do not face problems with execution policies. However, it is forbidden to enter as report meta-data any piece of information that could be used to identify the user because of privacy policies in the European Union.

Hospital

Proposed system was used in hospital in Wieliszew by a group of radiologists who specialize in oncological reporting. Reports created there were lengthy, as this specialization requires very often comparisons between many points in time to asses reaction to chemotherapy. The work-flow was organized in a way that a radiologist had a group of patients they are assigned to, so it was easier to remember the treatment history of a patient. Radiologists who worked in Wieliszew provided precious feedback and many of their suggestions were included in the system. Several extensions were developed to make state comparison easier, e.g., grouping reports by title, so the radiologists can see history of a single patient easily. Report templates were shared between radiologists, so each new user could start getting to know the software without the tedious step of creating their own templates. Changes to templates were propagated to all users automatically, so the team could discuss what the best phrasing of symptoms are. A lot of effort has already been made to create standardized terminology by the nosologists who worked on LOINC and SNOMED, but unfortunately these systems have no translations for Polish language, so they were not applicable to this working environment. Instead of using existing controlled vocabularies, templates were based on reporting history of the most experienced members of the team.

Network of clinics

Structured reporting system is used by a large network of private clinics in Łódź. The system is deployed in the local network, limiting access to not authorized users and providing better security and privacy. Integration using HL7 protocol with their RIS system is being developed, so the systems will cooperate with existing systems to optimize the work-flow of radiologists. Proposed system is used as an extension to the RIS system, focusing on the contents of the report and then exporting the report in HTML format and storing it in the RIS, which holds all administrative and billing data, e.g. SSN, detailed address.

   Moreover, it is planned to use several concepts from the proposed system to develop a program that could be a more generic tool to create different types of medical documents. This would require changes not to the proposed ontology, but also to all the parts of the program that contain radiology specific terminology and meta-data.

Chapter 6

Conclusion and ideas for the future

6.1 Realization of the thesis objectives

The main objective of the thesis was to increase overall productivity of radiologists as the demand for their services seems to unsatisfiable. This objective was achieved by means of:

- Proposing a modified work-flow that maximizes time spent on using field knowledge.

- Designing and implementing a system that backs the modified work-flow and minimizes risk of making a mistake by providing predefined sets contextual suggestions to the doctor.

In order to gain satisfactory level of understanding of the problem, observations of real-world working environments were necessary. Although it is a time consuming task, it allowed for tailoring the software to the end-user needs. Moreover, it required patience from the radiologists who were cooperating, as many questions were asked and several prototypes of the solutions were presented. The final solution to the problem was not implemented on the first try – several prototypes have been presented and user feedback was collected. The evolutionary approach to software development allowed for better understanding the domain of the problem and observing how users learn to use new pieces of software.

6.1.1 Knowledge engineering vs. template approach

Existing solutions to the problem are not very well described in the literature. Productivity of radiologists is often treated as an emergent property of a system that has knowledge base as their back-end. It has been observed that it does not have to be always true. Tools for knowledge structurization can be very complex as they are not focusing on minimizing time spent to encode knowledge but on completeness of the resulting knowledge graphs. Eventually, a solution that achieves the primary goal was formed by separating the creation of knowledge base (template creation) from the usage of the knowledge (report creation). Due to limited workforce, knowledge engineering functionalities are very limited, however, it makes the software more accessible to the radiologists as there is no steep learning curve. The report creation part of the program was optimized to use the stored knowledge in a way that only the currently important pieces of information are shown to the user.

6.1.2   StructuRad – similar idea

One similar solution has been found after the design and implementation of the proposed system – StructuRad. The program used to operate in Midway Hospital. It was a .NET application and its development started in 1996 [21]. Unfortunately, only an overview presentation and two screenshots of the program can be found on the Internet, as the company was acquired by another entity. What can be inferred from the available materials are the similarities in the productivity-first approach. The radiologists used to create reports based on predefined templates by selecting check-boxes next to suitable phrases. The report was also represented as a tree that would then reorganize nodes in order to create text that is linguistically correct. American style of creating reports has some differences caused by the fact that very often reports are dictated, that allowed the software to create reports with different information granularity and not to worry about details of the language grammar.

6.1.3   Simplicity of the template approach

Ideas implemented in this structured reporting system are very simple from the theoretical point of view. In fact, template-based text generation existed for years and can be found in most of the programs developed hitherto [24]. Template approach gives users the predictability of the resulting text, what increases their confidence while using the program. However, even such simple methods, supported with well-thought user experience design can give satisfying results. This is another example of one of the principles governing our world that the complexity can emerge from very simple rules [25]. Although the software is not designed for wide spectrum of types of users, several dozens of radiologists from different working environments have used the software as their main tool to create radiological reports.

6.1.4   Balance between structure and customizability

In a perfectly structured world, any phrase used in the report could be explained using some existing ontology, what would allow for automated reasoning about the causes of a pathology. Context of a phrase would allow for additional questions that would help including more detailed information, e.g. after stating "pain in the leg", the system could ask "Which leg, left or right?". In the extreme case, one could translate all relations and entities in the ontology, allowing for automatic translation of the whole report content without any external linguistic help. However, reports are created using natural languages that have their own nuances, irregularities that make it difficult to analyze custom phrases automatically. One of the biggest challenges is to find a point on the structure–customizability axis that allows for automated information extraction and allows for level of customization that makes the users of the system satisfied. In the case of this structured reporting system, more attention was paid to the customization of report content, as radiology was a new field to the designer of the system and even the end-users were not aware of the state of the art in this field. Rich customization functionality allowed for observation of commonly occurring patterns. This knowledge could be later used to structurize the parts that are occurring frequently thus making them standardized and increasing the overall productivity of the user.

6.1.5 Achievements of this thesis

The system has been used by a number of radiologists in Poland. The opinions about the program are indicating that everyday users like the overall user-experience and they are more productive while using the software. This program was used not only by independent indivudual teleradiologists, but also by radiologists from hospital in Wieliszew and a network of clinics and Łódź. By generating more than a thousand reports using this program, radiologists and their employers saved a lot of time what allowed to treat more patients in the same period of time.

6.2 Possible further extensions

One should look at the proposed system as a battle-tested idea that can be used to create bigger system focused on optimizing work-flow of not only radiologists but also other medical specializations. There are several paths that can be taken to make the reports created using the system more specific and language-agnostic and decrease the report turnaround time. Some ideas, as well as may they sound to a Computer Scientist, would require radical changes to the way doctors work currently. Revolution in reporting is not very likely to happen as there are many existing systems on the market that are being used continuously. Radical change to the reporting philosophy would make them irrelevant as they would not be able to store complex data of different shapes like trees or graphs in comparison to the relations native to relational databases that do not make it easy to store information encoded in these structures [27].

6.2.1 Complex ontologies support

One could generalize the ontology that is backing the system or use existing, well-designed ones like SNOMED CT. This approach would allow for better interoperability with other systems as publicly available ontologies leave no space for ambiguity and provide global identifiers that can be used to infer exact meaning from text phrases. One could try to map free text to the SNOMED CT ontology and ask the radiologist for more information in case of need for additional data to get rid of ambiguities. This, however, would make the solution not applicable to Polish market as this ontology does not support Polish language.

6.2.2 Recurrent neural networks and natural language generation methods

In the future, one could think about a way to generalize the reporting ontology or implement a mechanism to create custom ontologies to provide tools that would allow to encode a hierarchy of an arbitrary depth. After obtaining a well-formed tree representing relations between causes and effects, one could experiment with applying recurrent neural networks (RNNs) to automatically generate report's text from structured data source [20]. This approach is already used at Google to generate natural language phrases from structured data, e.g. user-friendly weather forecast descriptions based on numerical data [28]. This approach is getting more and more popular as computer systems capture enormous amounts of data that must be then presented to the human user in a way that can be easily understood.

6.2.3   Linguistic engine

Another direction for improvements would be a natural language processing mechanism that would learn typical co-occurrences of report items and would ask the radiologist whether an unusual report content is caused by a mistake made while creating the report or by a truly unusual pathology. As there is no natural language agnostic solution, a separate linguistic engine would have to be used for each supported language.

Bibliography

[1] Occupational outlook handbook: Physicians and Surgeons, available at: https://www.bls.gov/ooh/healthcare/physicians-and-surgeons.htm, [08.10.2017]

[2] Recht, et al, Artificial Intelligence: Threat or boon to Radiologists?,, Journal of the American College of Radiology, Volume 14 , Issue 11.

[3] Benson, Principles of health interoperability HL7 and SNOMED, Springer-Verlag, London, 2010.

[4] Clunie, DICOM Structured Reporting, 1st Edition, 2000.

[5] HL7 Structured Documents group subpage, available at: http://www.hl7.org/Special/committees/structure/index.cfm [10.10.2017]

[6] Careers in radiologic technology, available at: https://www.asrt.org/main/careers/careers-in-radiologic-technology/who-are-radiologic-technologists [13.11.2017]

[7] Bhavinj, The Typical Radiologist Work-Day, available at: https://radiologystories.com/2013/10/31/the-typical-radiologist-work-day/ [10.10.2017]

[8] Osirix software product page, available at: http://www.osirix-viewer.com/osirix/overview/ [15.10.2017]

[9] Langer, Impact of Speech Recognition on Radiologist Productivity.

[10] Toit et al, The accuracy of radiology speech recognition. reports in a multilingual South African teaching hospital.

[11] C# language formal grammar definitions, available at: https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/language-specification/lexical-structure [02.10.2017]

[12] Liebeskind, Intracranial Hemorrhage, available at: http://emedicine.medscape.com/article/1163977-overview [03.10.2017]

[13] Hanenberg, et. al., An empirical study on the impact of static typing on software maintainability.

[14] Forking a repo using GIT version control system, available at:
     https://help.github.com/articles/fork-a-repo/ [10.10.2017]

[15] RECIST 1.1 definition, available at:
     https://en.wikipedia.org/wiki/Response_Evaluation_Criteria_in_Solid_Tumors [07.10.2017]

[16] Smith, Risk management for the radiologist, available at:
     https://www.ncbi.nlm.nih.gov/pubmed/3497558 [29.10.2017]

[17] Panda, Understanding Angular's $apply() and $digest(), available at:
     https://www.sitepoint.com/understanding-angulars-apply-digest/ [17.10.2017]

[18] Popovic, JSON support in SQL Server 2016, available at:
     https://blogs.msdn.microsoft.com/jocapc/2015/05/16/json-support-in-sql-server-2016/
     [14.10.2017]

[19] Bootstrap Framework documentation, available at: https://getbootstrap.com/docs/3.3/css/#type
     [15.01.2018]

[20] Kvasnička, How to Process Structured Data by Neural Networks?, available at:
     http://www2.fiit.stuba.sk/ kvasnicka/NeuralNetworks/6.prednaska/Kvasnicka_RNN_exten_transp.pdf
     [17.03.2018]

[21] Zucherman, StructuRad Presentation Reporting Tool Overview, available at:
     http://www.structuredreporting.com/structurad-presentation.pdf [10.04.2018]

[22] Zadrożna et al, Private sector is crucial for Polish health-care system, available [in Polish] at :
     https://www.polskieradio.pl/42/4393/Artykul/1535315,Prywatna-sluzba-zdrowia-jest-w-
     Polsce-niezbedna-A-sektor-niepubliczny-ochrony-zdrowia-ma-szanse-na-blyskawiczny-
     rozwoj [10.04.2018]

[23] Dunlevy, Inefficient public hospitals wasting $1 billion, finds Grattan Institute report,
     available at:
     http://www.news.com.au/lifestyle/health/inefficient-public-hospitals-wasting-1-billion-finds-
     grattan-institute-report/news-story/7aed1d890868572790cdc2a2a04c7868

[24] Reiter, NLG vs. Templates, available at:
     https://arxiv.org/pdf/cmp-lg/9504013.pdf [10.03.2018]

[25] Wolfram, A New Kind of Science, Wolfram Media, 2002.

[26] Ladrowski, Private health-care in Poland – can it even be profitable?, available [in Polish] at:
     https://www.klinikiwpolsce.pl/aktualnosci/prywatna_sluzba_zdrowia__czy_to_sie_oplaca/
     [03.03.2018]

[27] Celko, Joe Celko's SQL for Smarties (Third edition), Morgan Kaufmann, San Francisco 2005.

[28] NLG at Google Research, available at: https://www.youtube.com/watch?v=MNvT5JekDpg
     [video] [10.04.2018]