VILNIUS UNIVERSITY
FACULTY OF MATHEMATICS AND INFORMATICS
SOFTWARE ENGINEERING STUDY PROGRAM

Ensuring quality of crystallographic data with the help of interactive and automatic validation tools

Kristalografijos duomenų kokybės užtikrinimas naudojant interaktyvias ir automatines validacijos priemones

Master's thesis

Done by:	Monika Kaltenytė	(signature)
Thesis Supervisor:	prof. dr. Saulius Gražulis	(signature)
Reviewer:	prof. dr. Linas Laibinis	(signature)

Vilnius – 2023

CONTENTS

1. Acknowledgements

I would like to express my sincere gratitude and appreciation to all those who have contributed to the completion of this master's thesis. I am extremely grateful to my supervisor Saulius Gražulis, for sharing his knowledge and guiding me into the crystallography world with his valuable and insightful suggestions throughout the entire process of this thesis. I've learned a lot and was amazed by how much support I've gotten during these two years. The encouragement and belief in my abilities have been of significant importance to my academic and personal growth and the talks we've had broadened my perspective on a variety of topics.

I would like to express my deepest appreciation to my family, friends, and colleagues for their support, encouragement, and love throughout this journey even when I questioned my own sanity. Their belief in my abilities and their constant motivation was instrumental in overcoming challenges and maintaining my focus. I am forever grateful for their understanding and for not letting me give up when things got hard. To my fellow students, for commiserating over shared struggles, providing comic relief during the most stressful times, and reminding me that it's okay to take breaks.

I am deeply grateful to everyone who has contributed to this master's thesis, directly or indirectly. I've learned a lot during these two years and realized how lucky I am to be surrounded by so many amazing professionals. I really hope that with some more work, my system will be able to make a difference in the crystallography world.

2. Master thesis

2.1. Motivation and innovation

No matter the science field, gathering data and information is a very important step in the research process. Having data, and most importantly correct data, helps scientists and researchers to take the right decisions and allows them to support and prove hypotheses. As science, in general, is an extremely wide subject, this thesis will mainly put focus on crystallography. Crystallography is a science that explores the structure of crystalline solids and provides an understanding of the internal structure of materials. Its achievements opened doors in many areas of science, including, but not limited to biology, drug design, chemistry, archeology, geology, and physics. As crystallography is progressively evolving, causing more interest and attracting more funds, the number of crystal structures provided by crystallographers is increasing. As of today, Crystallography Open Database (COD) contains more than 400 000 entries placed in the public domain by the contributors [Cry21] [GCD+09], while Protein Data Bank also exceeds 100 000 deposited structures [WMD+08]. Even though the quality of structures increased over the years, some crystal structures that are published are incorrect [Har96], consequently, thorough check and analysis is a critical step to ensure the integrity of the data before the structures are associated with journal publications and published in the database. The majority of structural databases and crystallographic journals use strict grammar files called Crystallographic Information Files (CIF) for storing and reporting data on specific crystal structures. These files are established by the International Union of Crystallography (IUCr) and work as a standard for the transmission of crystallographic data [Bro96].

Crystallography validation aims to confirm that the data is reliable and the structure determination process is accurate, in order to make sure that the data is suitable for its intended purpose. Validation can be described as value comparison against a set of test criteria and takes into account various parameters like whether files have all expected data and whether all related parameters are consistent. Data validation is an essential step to catch defects and mistakes in the early stages and ensure that data is usable, accurate, and clean. One of the validation ways is formal verification which essentially is a process of proving the correctness of data by using mathematical methods. Only well-validated data should be used and stored in science. If data is not validated correctly it could result in various inaccurate outcomes and potentially huge issues such as bad quality data, wrong results of experiments and research, etc. In crystallography validated data would mean that correct atom types are assigned, the structure syntax is valid, the crystal structure has correct symmetry (this is one of the most common errors in crystal structures [Har96]), and has correct unit cells, etc. Sometimes even structures that do not exist at all are found using formal validation tools. The formal check of CIF files consists of three levels: syntax validation, validation according to dictionaries, and check according to additional subject area criteria [VMG21].

Crystallographers use the COD CIF parser tool to examine and detect the position and origin of syntactic deficiencies as well as automatically fix the most common errors such as adding missing data headers for files or quotes where necessary [MVB+16]. Syntactic and semantic error detection is then left to higher-level tools. There are different bindings of a parser available in Perl, C, Java,

and Python programming languages. Another tool for running a check is checkCIF [Spe03], a website that reports on the integrity and consistency of crystal structure determinations reported in CIF and returns responses on syntax, geometry and cell details, structure factors, etc.

Dictionary validation is done by using CIF dictionaries and dedicated programs, such as cif_validate [VMG21]. These dictionaries provide a formal categorization of crystallographic ideas and terms designed in a machine-readable manner to cover the requirements of processed data and derived structural results and can check whether the data names used in the CIF were defined in the dictionary and whether the data values lay within the prescribed ranges [Bro96]. cif_validate tool is not executed in COD at the moment, but since it also provides messages that could be easily parsed and used in boundaries calculations for the system, might be included in future versions. This tool will be further investigated later in the thesis.

However, automatic validation for these structures is usually not enough because it is not always possible to determine whether the structure is correct only by automated tools as they show that the data is well-formed but there can be subject-specific details that need to be reviewed by a person. For this reason, in crystallography and any other science fields, scientists' research has to be reviewed before it can be published. This process is known as peer review. It is standard practice by most credible scientific journals and acts as another filter to ensure that only high-quality information is published and a scientific manuscript is experimentally and factually correct. The peer review process to publish crystallography research in IUCr is carried out by independent experts who assess each manuscript. It is a process that evaluates the validity, quality, and originality of publications before they are published. Its ultimate goal is to protect scientific integrity by weeding out publications that are either invalid or of bad quality. [KSA14].

At first, the article is sent to the main editors, who decide whether it should be rejected or passed for peer review. For example, in IUCr journals suitable articles are assigned to a co-editor who is responsible for the whole review process and for rejecting or accepting the article. Co-editor also assigns (most often) two referees that evaluate the work and allow the co-editor to make a judgment. Usually, authors do not know the identities of the referees, while the referees know the author, but there are attempts to anonymize the author as well to protect it from discrimination and lessen the risk of conscious or unconscious bias. Based on the referee's reports, the author may have to revise his work, or the co-editor can reject the work if revisions are major or accept the article if it doesn't need further revision.

Not only human-readable text but also data has to be checked for possible errors by a reviewer to ensure state-of-the-art crystallographic results as only having automated checks is not enough to ensure that data is of the best quality and correctness. However as there's an exponential increase in the number of determined structural data entries, it is nearly impossible to review all structures [GCD+09]. There have been cases when falsified and fabricated research pages were published and multiple incorrect structures were added to widely used crystallography databases. It takes a massive amount of time and effort to investigate fraudulent data and the largest ever fraud in protein crystallography took even nine years and had important implications for discovering drugs against the dengue virus and for understanding the human immune system [Cha03]. Such investigations

result in the retraction of these structures and whole articles which is expensive not only in terms of money but also because it may impede further advances in science research and innovations and needs a lot of human resources [CZL+11].

To prevent erroneous data to be published it is important to include the crystallography community in the crystal structures review process so that when crystallographers have doubts about a particular structure they would be able to discuss it and accept the structure to be published to the database if it's correct or leave comments on the issues otherwise. Such an approach is not new and is already widely used in the programming community. There is a number of systems available like Github, Gitlab, or Bitbucket where programmers can do code reviews and point out logical problems, uncovered edge cases, or other issues. All of these systems have similar guidelines: firstly, a developer (a contributor) creates a pull request – an event where that person asks the maintainer of the specific repository to review code they want to add to a project. The default approach is to select someone who is an expert in this field and depending on the agreement, there might be several reviewers needed to confirm that all changes for high-impact risks to quality, performance, reliability, security, and maintainability were analyzed until the pull request can be merged. If not, additional functionality and fixes have to be added and reviewers and contributors discuss the work until it is considered done and approved by the reviewers [KBG+15]. This practice is used both in companies and open-source projects where anyone who has expertise in programming can suggest ideas or participate in development and improvement. The main goal of code reviews is to improve the quality, the performance of the code, prevent errors that users face when using the systems, and confirm that the system is reliable and will be available as needed as well as keep the history of each feature so that it would be possible to see what decisions and why were made [MKA+14].

The assumption is that the approach of peer reviews which is used in programming could be successfully applied to crystal structures reviews. However, current implementations of systems like Github, Gitlab, or Bitbucket are not immediately usable with COD infrastructure, because it is needed to combine both formal checks and crystallographers' conclusions and first of all the right balance between the level of human control and automation must be established. Automatic validation is not sufficient for crystal structures as having only formal validation means having a decision-making system that solely depends on automatic solutions. Since final decisions are complex and should be taken by humans before publishing data to the database, an expert must be involved in the decision-making process [TFA88]. However, to do that, opinions and evaluations of scientists need to be converted to numeric value and for that, specific constraints, for example, experience, number of published structures or articles, reviewed number of structures, or others, must be taken into account. The validity and correctness of the reviewer's decision could be checked by setting different weights for the pieces of information (examining results with and without experts' opinions) and comparing the results with those already known. The system's validity could be checked by taking known incorrect results and comparing them against the created system. Having both automatic checks and the numeric value of crystallographers' decisions, a restraint scale could be introduced to allow control of workload and the need for human attention with automatic validations. As a person's time is an extremely valuable resource, this scale would allow delegating scientists' at-

tention to the most critical structures and could be decreased and increased as needed. A balance between automation and human control is described as levels of autonomy [JMR+17] that should also be defined. After that, it is needed to review whether it is possible to use existing open source solutions to integrate them in the review process or implement a new system and merge automatic validation and decisions of scientists as well as integrate created system into COD infrastructure.

To summarise, some systems only use automated verification and do not have the means to include insights of the experts or stand-alone tools like checkCIF that use internal expert work [GCD+09], but do not allow the input of reviewers from the scientific communities like e.g. CCDC [KWT72]. There is a need to have a system that would open to combining both formal checks, including new parameters or methods, and experts' opinions into a single workflow, and be able to read multiple indicators and signals from different sources, present them to a decision maker or, if a person does not make a decision, to an automatic decision maker. As of today, such a combined system is not implemented anywhere and is the subject of this research.

2.2. Hypothesis

To prevent erroneous data it is important to include the crystallography community in the crystal structures review process so that when crystallographers have doubts about a particular structure they would be able to discuss it and accept the structure to be published to the database if it's correct or leave comments on the issues otherwise. Integrating formal validation with experts' opinions would improve the quality of crystal structures overall.

2.3. Goals

1. Design an information workflow that combines inputs from humans (e.g. experts and reviewers) and machines (automated validation systems, AI systems), which could facilitate the decision-making process about publishable (crystallographic) data and possibly decision-making when human decisions can not be taken timely. Assess the efficiency of this workflow.

2. Enable crystallographers and scientists to view and efficiently validate crystal structures submitted to the database. Design and implement a system that allows reviewers to make correct decisions on the trustworthiness of crystallographic data. A successful system should demonstrate that the data peer review process is capable of catching some errors that are not detected automatically, at least on test examples.

2.4. Tasks

1. Examine quality standards for crystal structures and what is done to validate and ensure the reliability and quality of data that is uploaded to the Crystallography Open Database.

2. Review what parameters, methods, and tools are currently used to evaluate and validate crystal structures.

3. Review formal check levels: syntax validation, validation according to dictionaries, and check according to additional subject area criteria.

4. Detect the most common quality issues in crystal structures, their nature, and how they could be prevented.

5. Examine scientific publication and data review methods and how human expert input can be combined with the automated assessment to ensure high data quality.

6. Analyse how crystallography experts' opinions could be converted to probabilities so they could be included in check levels.

7. Analyse decision-making and decision support systems and levels of automation to find the balance between the automatic validation and experts' input processes and the optimal way to combine them.

8. Using analysis results from previous steps, propose and implement an interactive web system that could be used for discussions and peer reviews about crystal structures and integrate it with the COD structure.

2.5. Expected results

The expected result of the thesis work is the establishment of crystal data quality review methods. In addition to that, an interactive web system for reviewing crystal structures will be implemented based on our theoretical assessment. The system should be integrated with the COD testing server. Eventually, the system could be used by scientists and publishers to ensure scientific data quality.

3. Literature review

3.1. Review of Crystallography and databases

Crystallography is a science that studies crystalline structures and their properties. Its knowledge can be applied to various fields like mineralogy, chemistry, physics, mathematics, biology, medicine, drug design, and genomics, so the progress of these types of science highly depends on available knowledge of structures, their interactions, and functions. Because of crystallography's wide applicability, crystal data has been archived and kept for many decades in different forms like journal articles, collected structure reports, diffraction patterns, and later on in electronic databases. Progress in science over the years, lead to an increase in crystal structures as well, which created concern within the crystallographic community about how to store data, so that it would be reusable and easily looked up. The growth and availability of electronic devices allowed scientists to start managing the data automatically, especially when the first crystallographic database, the CSD (Cambridge Structural Database), was founded by the Cambridge Crystallographic Data Centre (CCDC) in 1965. As of today, there are a number of crystallographic databases for different types of crystalline structures as the use of database information is convenient to use due to its integrated search capabilities, structured view, and constant growth of databases and data management [BGH+17]. As the CSD is a closed–source database, it can't be used for this thesis for legal reasons. However, there's an open-source alternative called Crystallography Open Database (COD) which is an open–source alternative that allows scientists and researchers (including us) to use crystallographic data and have a common place for storing such data.

3.2. Crystallography Open Database

An open crystallographic database of small molecules was chosen as the source of crystallographic information for the study described in this work. The Crystallography Open Database (COD) is an open source project whose main goal is to collect all available structural data (regardless of whether it's inorganic, metal–organic, or organic) in one database, store that data in a structured way and provide search capabilities or ability to download. The project started in 2003 with the primary goal to create single storage that would hold all information regarding molecules and crystal structures. With the hope to promote crystallography worldwide and enable collaboration between scientists to share their knowledge an 'open-access' methodology was chosen. The main source of the Crystallography Open Database is peer-reviewed scientific journals like the International Union of Crystallography (IUCr), the American Chemical Society (ACS), and the Royal Society of Chemistry (RSC). Structure data from these and other scientific journals are regularly uploaded to the COD. The database also has an automated data upload interface, allowing researchers around the world to contribute to the completion of this structured information resource. As of today, COD contains more than 400 000 entries placed in the public domain by the contributors and this number is likely to grow with further advance of science [Cry21] [GCD+09].

3.3. Contents

The Crystallography Open Database has two main parts: first is an SQL database and second is a collection of structured data files from various scientific journals or donated from various crystallographic laboratories. The first part is related to data writing into the database, the second is for reading data. [GCD+09]. The COD database has a data retrieval service that allows you to find CIF files in a COD dataset according to the id, name, journal, year, etc. It is also possible to browse the data by the journal of publications such as Acta Crystallographica Section E, Inorganic Chemistry, Organometallics, etc, or by the year of publication. The database is also provided for open read access using standard SQL database access tools like MySQL protocol, where SQL query language can be used. Connection to the database with a read-only user would look like this:

```
mysql -u cod_reader -h www.crystallography.net
```

In general, structural data can be accessed using a web interface via http://www.crystallography.net/cod/search.html. Structures in the database are provided as entries that are generated from CIF files containing a bibliography and parameters that describe the size and contents of a unit cell, the diffraction experiment, and the quality of the data (e.g. R factor). Even though a huge number of checks and modifications are performed automatically, automated systems are unable to detect or correct all problems. The COD allows for a combination of automated checks and manual repairs when a scientist is required. All data inside COD was recently validated against the IUCr core CIF dictionary (standardized set of definitions and data items that describe crystallographic data) and confirmed with the COD deposition tools and results such as errors, warnings or other comments were also stored in the database [GDM+11].

3.4. The Cambridge Structural Database

The Cambridge Structural Database was founded in 1965 as a principal repository for crystal structures of organic and metal-organic compounds and is maintained by the Cambridge Crystallographic Data Centre (CCDC). It started as a computer file that contained bibliographic information and numerical data extracted from various scientific journals and over time grew into a structured database with interactive software applications to check and generate crystal structures. As COD, most of the crystal structures are provided using .cif extension files to store the information in the database [AJS+04]. (CSD). The Cambridge Structural Database includes tools for structure processing and interactive visualization. CSD-Xpedite is one of them, and it allows you to handle structure deposition more efficiently. When a structure is deposited with the CCDC, CSD-Xpedite automatically examines and validates the submitted files, leveraging the CSD's knowledge. It helps the CCDC to more efficiently manage the throughput of these depositions by storing all data (including Crystallographic Information Files (CIFs), structure factors, and supporting documentation) and correspondence in one system. CSD-Xpedite also offers an experienced editorial team to confirm scientific information before inclusion in the CSD, as well as expediting the entry of novel structures into the database. CSD-Xpedite together with a range of other tools supported by the

CSD community with regular updates and a variety of tools for CIF files, e.g. enCIFer, checkCIF can be used to verify the correctness of the structures.

3.5. ChEMBL

ChEMBL is a manually curated, open-access bioactivity database that deposits data on structures that contain drug-like properties and bioactive compounds. Information on these structures is crucial in research of diseases, new drugs discovery, and various medical experiments, however, published data usually comes in unstructured formats, for example, images that are not searchable or structures with a variety of synonyms but no unique identifiers which could be used as a reference later. As COD, it is also open source, licensed under public copyright Creative Commons (CC) license that allows the distribution of copyrighted work. Similarly, information that is uploaded to this database is also regularly extracted from peer-reviewed articles and journals like the Journal of Medicinal Chemistry, Bioorganic Medicinal Chemistry Letters, and Journal of Natural Products. After extracting data, detailed testing of the compounds is executed: structures are checked for the unusual valence of atoms, incorrect structures for common drugs, etc. The results are normalized according to the set of rules and configurations and then it can be uploaded to the database. ChEMBL database has a user-friendly interface which is accessed via https://www.ebi.ac.uk/chembldb from which data can be programmatically retrieved in XML and JSON formats. However, as mentioned above, ChEMBL is operated manually using peer-reviewed articles, which means it lacks optimized automated processes for larger-scale data deposition [GBB+11] [MGB+18].

3.6. Protein data bank

The Protein Data Bank (PDB) was founded in 1971 and from that time grew from 7 structures to over 100 000 deposited structures today [WMD+08]. It is an open-access database that holds information about the 3D structures of large biological molecules, including proteins and nucleic acids, and is one of the most important and heavily used biological data assets that give scientists and researchers the ability to search across the archive and analyze the data. Protein data bank uses .pdb or .mmCIF extension format files which are plain text (ASCII) files and are used as a standard for files that contain atomic coordinates. They contain lines of information, where each line is called a record. A record holds information for atom coordination, and location, and defines bonds and connections. The structure information and components within a file are checked for errors using a specific dictionary called Chemical Component Dictionary and validated using community-accepted standards. Different from ChEMBL, the PDB uses automated validation tools to ensure the quality of the deposited data that are developed by the PDB organization. One of the tools is PROCHECK which is available for free and is used to check the quality of protein structure. It generates several output files in the default directory which have the same name as the original PDB file, but with different extensions [JDA+11]. There's also a PDB validation system available at https://validate-rcsb-1.wwpdb.org. To validate the files they must meet requirements, for example,

all the compound's PDB coordinate files must have a HETATM at the start of the atom coordinate line, which looks like this:

```
HETATM 1574  C1  GOL A 104      -1.268 -20.812  15.070  0.75 21.78           C
```

The PDB archive is curated by the Worldwide Protein Data Bank organization and together with a variety of specialists from wwPDB Partners from around the world collect, annotate, and validate structures that are added to PDB and provide many resources free of charge for scientists, developers, educators, and the general public, but the tools are not open for communities to develop [LO03] [ZDG+16].

3.7. Crystallographic information file

Crystals consist of atoms and their positions and motions can be described using several hundred parameters that are rich in numerical information. To be able to use these parameters for calculations of e.g., bonding geometry or providing results of atoms for display it is advantageous to keep the data in electronically readable form. As scientific investigation results together with listings of atomic coordinates and parameters are often printed in scientific journals it was needed to have a file structure that would be widely accepted and understood by the scientists. The first machine-readable crystallographic data format for structures was founded by Protein Data Bank in 1976 and called PDB format [Ber07]. This human-readable format consists of fixed-width lines. Identifiers that are up to six characters long allowed the data to be quickly and easily searched. PDB was later replaced by PDBx/mmCIF due to an increase in protein size. As crystallography and technologies, in general, are changing and evolving fast, it was necessary that the file structure would be flexible as well as would be able to grow and be open to extensions. For this purpose, the acronym of CIF, crystallographic information file, was embraced by the International Union of Crystallography in 1990 as a file structure for storing and distributing crystallographic information. Standard data items of the CIF file are publicly available and defined under the authority of an IUCr Committee. In 1999 another data format called Chemical Markup Language (CML) was introduced. The file is based on XML schema. It contains many chemical definitions formed from XML tags and definitions which allows the usage of XML-oriented tools [PKT+12]. In this work, the focus will be on CIF files. Each CIF in the COD has a unique identifier assigned which consists of seven digits from the range [1000000; 9999999]. This identifier is used to get the structure from the database. An example could be used a one-dimensional aluminophosphate which can be accessed using: https://www.crystallography.net/cif/1/00/00/1000000.cif, where the last digits 1000000 are the numbers of unique identifiers. One of the main benefits of CIF files is that they allow for the standardization and organization of crystallographic data. The data is stored in a consistent format, making it easier to compare and analyze different crystal structures. Additionally, CIF files include extensive metadata, allowing for the documentation of the experimental conditions and processing history of the data. Overall, CIF files are a valuable tool in the field of crystallography, allowing for the efficient storage, sharing, and analysis of crystallographic data. [BBB+16].

3.8. Syntax of CIF

For the CIF to be compatible with crystallographic software, it is important that the file would follow strict syntax rules. As a base for crystallographic information files, Self-Defining Text Archive and Retrieval file (STAR) structure is used. This format was designed as a general and universal technique for archiving and exchanging electronic data and was soon embraced by molecular-structure sciences, including crystallography. The main structural unit of a CIF file is a block of data, but according to the format specification, there can also be several blocks of data. The data block consists of a header and a main body describing the data: bibliographic information, crystal lattice parameters, material properties, symmetry operators, atomic coordinates, and so on. None of the elements listed in the body are necessary, so the data block may consist of a header alone. A CIF consists of case-insensitive identifiers called data names which usually start with the underline character "_" and data values that consist of characters describing particular information. It can be a numeric value, phrase or word, or simply a letter. An example of a data name item followed by its value:

```
_journal_year 1965
_chemical_formula_sum 'Cl Na'
_symmetry_cell_setting cubic
_cell_length_a        5.62
```

In case there is a need to describe the sequence of data loops can be used. Loops start with the data name "loop_". Then the list of names is provided and numerical values are assigned to each data name sequentially. The data can be written inline or formatted to supply better readability. The loop is considered terminated when another data name is provided. Example of the formatted loop:

```
loop_
_atom_site_label
_atom_site_fract_x
_atom_site_fract_y
_atom_site_fract_z
A1 0.1 0.2  0.3
A2  0.3 0.4  0.5
A3  0.6 0.7  0.8
_other_data_name   # end of the loop
```

CIF, as an interchange format, has several distinguishing characteristics:

1. CIF uses mathematical notions that are independent of implementation details like numerical equation sorting methods.

2. Because it has formal and compositional semantics, it may be used to define and prove property-preserving model transformations.

3. It supports large-scale system modeling by defining and instantiating parameterized processes (reuse, hierarchy) [Agu13].

3.9. Validation

Crystallography automation validation methods refer to the procedures used to ensure that automated systems for analyzing and interpreting crystallographic data are accurate and reliable. These methods can involve comparing the results of automated analysis to those obtained manually, using standardized test samples to evaluate the performance of the system, and implementing quality control measures to detect and correct any errors or deviations from expected results. Crystallography automation has become increasingly prevalent in recent years, as it allows for faster and more efficient data processing in a variety of fields, including pharmaceuticals, materials science, and structural biology. Simply collecting the data is viewed as insufficient for scientists and researchers that put the data into the database as only thoroughly checked and validated information can be later used for analysis and research. For this reason, there are several levels of data validity checks. The syntactic accuracy is covered in the first level. The first level deals with syntactic precision, meaning that data files that do not follow the defined syntax are very likely to be rejected or misinterpreted by the processing program. The second level is concerned with semantic validity, or the adherence to a set of formal criteria by each data field and its links to other data fields. During the third stage of the investigation, more difficult tests are conducted, often based on heuristics relevant to the research topic. The tests usually require the expertise and skills of scientists and may involve complex analyses or recreation of experiments [VMG21]. Validation programs are frequently employed in COD data management tasks as standalone solutions as well as components of larger automated systems. Any changes to the file contents are followed by a set of automated quality tests from the moment a CIF file is given for deposition. The test sets vary based on the situation, but they always contain a syntactic analysis. When CIF files are being uploaded via the COD website, a certain level of semantic correctness and conformance to domain-specific criteria must be ensured. The pipeline includes tools for diagnosing and even resolving common problems. There are different severity levels of issues that were found during the validation process:

1. NOTE – can be ignored if they contain information concerning dubious data values or automatically made data adjustments. An example of a note could be missing the name or surname of the author. This doesn't mean that data cannot be deposited, but rather that it is needed to check whether the name was written correctly as the structure itself did not report any serious problems.

2. WARNING – indicates a more serious problem that needs to be investigated and fixed or justified. In general, this means that work could be continued, but most probably the result will not be satisfying. An example of a warning could be, getting a warning that there's no bibliography. It doesn't mean that work can't be continued, however, there's no way to check that information provided is correct which can result in issues later on.

3. ERROR – indicates serious problems that must be remedied before the deposition may continue [VMG21]. This means that work cannot be continued and such data cannot be used and deposited anywhere.

Overall, the validation of crystallography automation systems is a critical step in ensuring the accuracy and reliability of the data produced. After that, another validation procedure should be carried out by reviewers during the peer-review process. Automated validation decreases time which is needed to ensure data quality and shows great potential for reusability and interoperability meaning that they can be further improved and incorporated into data validation pipelines to improve the process even more [VMG21].

3.9.1. Syntactic analysis

To guarantee that CIF files that are stored in the COD are syntactically accurate, syntactic analysis is used. During this phase syntax errors, missing values, and unrecognized symbols are checked. Automated tools can be used for that, one of them is an error parser called COD::CIF::Parser has been demonstrated to be one of the quickest and most comprehensive in the field as it supports both CIF1.1 and CIF2.0 formats [VMG21]. Passing syntactic analysis means that scientists can proceed with further work with the structures.

3.9.2. Validation against CIF dictionaries

While correct grammar is extremely important, before CIF can be used, it is needed to have a set of names and rules to cover the requirements of raw and processed data. For this reason, CIF dictionaries are used. They provide a formal description of crystallographic terms in a machine-readable format that facilitates validation and structuring of data and is being maintained by COMCIFS (The Committee for the Maintenance of the CIF Standard). As new definitions can be added as needed, it gives the CIF structure flexibility and space to grow with the evolving science. The CIF dictionaries are similar to any other dictionary designed to be read by people. It consists of crystallographic terms followed by definitions:

```
Name:
  '_atom_site_[]'
Definition:
   Data items in the ATOM_SITE category record details about
   the atom sites in a crystal structure, such as the positional
   coordinates, atomic displacement parameters, and magnetic moments
   and directions.
```

For some, examples are also provided (see: https://www.iucr.org/__data/iucr/cifdic_html/1/cif_core.dic/Catom_site.html). Data items are sorted alphabetically inside each category, and categories are explained in alphabetical order. CIF dictionaries keep the same syntax structure as CIFs which enables the dictionary to read and validate CIF files and explains how to write programs that could parse CIFs. Different dictionaries such as Core dictionary, restraints dictionary, powder, symmetry dictionaries, etc contain different sets of data names that are needed for crystal structures and all of them can be found at https://www.iucr.org/resources/cif/dictionaries [Bro96]. It can detect that the values are not of the

type that they were declared, are not in a specific value range, have deprecated data items that should be replaced by other data items, etc. Dictionaries are used for formal semantic validation and play a crucial role in making sure that data is correct.

3.9.3. CIF applications and tools

There are many tools available for the visualization and validation of CIFs that help to make certain that files are format and syntax compliant and can be deposited to databases and used in journals.

1. EnCIFer – is a powerful computer program written in C++ programming language, which enables users to validate syntax integrity, and view or create CIFs. The program can load different CIF dictionaries for data validation and then locate and report violations of the syntax based on them. Each type of error is highlighted in different, configurable colors and is extremely helpful when locating fields missing semicolons. For example, the data-block header usually uses bold red color, while bold blue is used for data names in CIF dictionaries. EnCIFer also checks if all required data items are present and provides results that are divided into errors, warnings, and remarks. Besides delivering extensive syntax validation based on dictionaries, encipher is also helpful when creating CIFs. It has Crystal Data Wizard which allows it to enter crystallographic and chemical information of structures like chemical name, source of the chemical compound, different types of formulas, crystal habit and color, etc, as well as diffraction information. Even though enCIFer provides many features, it only checks the syntax integrity and cannot be used as a standalone tool for confirming crystal structure credibility [AJS+04].

2. COD::CIF::Parser – is a CIF parser, designed to detect and fix the most common syntax errors that are found in CIF files. Different implementations in Perl, C, and Python environments of the parser are available. This tool can find and report position and types of syntactic issues such as missing quotes, missing data block headers, duplicated data names, etc, which are the most common and obvious syntactic errors. Parser takes CIF and returns a structure split into key-value pairs that represent a single CIF data block as shown below.

```
[
    "name" => "example",
    "values" => {
        "_journal_year" => ["1998"],
    },
    "loops" => [
        #   0:
        [
            #   0:
            "_space_group_symop_id",
        ],
```

```
    ],
    "tags"=> [
        #   0:
        "_cell_measurement_temperature",
    ],
]
```

Error recognition is implemented using various CIF grammar rules that detect erroneous structures such as duplicate data names and items, unrecognized symbols, values before the first data block, and missing single or double closing quotes. Corrections of each or all errors can be enabled or disabled by using different types of treatments such as (but not limited to):

(a) `fix_missing_closing_single_quote`
– insert missing quote.

(b) `fix_duplicate_tags_with_same_values`
– remove duplicate values.

(c) `fix_all`
– correct all found issues.

With the ability to detect and fix the most common errors, the parser [MVB+16]

3. checkCIF – is a tool that examines the consistency and accuracy of CIF-format crystal structure determinations, and is available via https://checkcif.iucr.org and sponsored by IUCr or can be installed locally. CheckCIF takes a CIF file as an input and can provide reports in HTML, PDF, and PDF email formats and can provide different types of validation: full validation of CIF and structure, validation of CIF and structure factors together with IUCr publication validation, or only CIF validation, that checks CIF syntax and construction but doesn't take into account structure factors. The program can detect missing symmetry, completeness of the file, internal consistency, and expected values. Furthermore, it allows to select of different levels of alerts that should be visible in the report. These alerts indicate the seriousness of the problem detected in the files and mostly come in one-line, short messages together with an explanation. Alert does not necessarily mean that it is an error, they can also point out that some information is missing and should also not be neglected. However, it is necessary to resolve as many alerts as possible. Minor alerts are usually a result of some overseen problems or omission of details. More significant problems may need further measurements, structure refinements, or discussions. In case major problems (level A alert) remain, they need to be justified and an explanation has to be provided to be taken into consideration during the review process [Spe20].

3.9.4. checkCIF alert levels

Each alert has the format:

`test-name_ALERT_alert-type_alert-level`

(a) Level A alert – This type of alert indicates that most likely a serious problem with the data has been detected. Major alerts need to be fixed before or resolved by submitting data to the paper or the database. An example of such an error:

`PUBL008_ALERT_1_A _publ_section_title is missing. Title of paper.`

(b) Level B alert – indicates a potentially serious problem that should be considered.

(c) Level C alert – a problem that needs to be checked, e.g., the label has not been recognized as a standard identifier.

(d) Level G alert – contains general information that something looks unexpected or some data is missing.

`PUBL017_ALERT_1_G The _publ_section_references section is missing or empty.`

3.9.5. checkCIF alert types

Even though checkCIF A-level alerts can be ignored in some cases with the right justification and the absence of alerts does not mean there are no aspects of the results needing attention. The same conclusion can be applied to other discussed tools. Even though they provide extensive validation and even error correction, they do not give a final decision if the structure is correct or not and can be published or should be rejected. The final decision is left for reviewers, referees, or users of the reported results [Spe20].

3.9.6. Crystallographic data quality criteria

To provide reliable information, calculations need only be performed with high-quality crystal structures. To evaluate the quality of the identified structures, there are several numerical parameters divided into global ones, which describe the quality of the whole structure, and local ones, which describe the quality of individual parts of the structure. One of the most widely used numerical global criteria for the quality of crystallographic information is the crystallographic R factor. This is a metric for the discrepancy between observed amplitudes (F_o) and model-calculated amplitudes (F_c). The most typical indicator of effective refining is a low R_{free} value (the lower the value, the better the fit between the experimental data and the model). These calculations can help crystallographers to determine whether the crystal structure is measured well and can be further used. R factor is usually checked during the third stage of validation when the subject area is being validated by reviewing the data in more detail by the scientists alongside the check of chemical restrictions (e.g. whether the structure can have connections with other structures as defined). Since the judgment of such validity has to be made by a scientist, it makes this level the most difficult and the most important.

3.9.7. Additional semantic validation

While CIF dictionaries are used for formal semantic validation, it does not address the whole range of domain-specific restrictions that are important to ensure the correctness of data. For this reason, additional tests are performed (e.g. checks for R factors). In COD cif_cod_check program can be used to check data against validation guidelines that are provided by IUCr [VMG21]. The tool parses the CIF file and checks if the data matches the guidelines. It is a part of open source, command line scripts for manipulating CIF files and comes together with the cod-tools package.

3.10. Reviews

3.10.1. Peer reviews

When structure or publication passed all validation checks and is ready to be published or needs additional attention in general, other scientists might have to review provided information or re-do experiments to prove the validity of such data. The process of assessing the quality of the data or manuscript before it is published is called peer review. It is by far one of the most important processes of not just crystallography journals but of all of science. Based on peer review results scientific research is led, grants allocated for further experiments, or even Nobel prizes won [Smi06]. The figure below (see Figure 1) shows the peer review process.

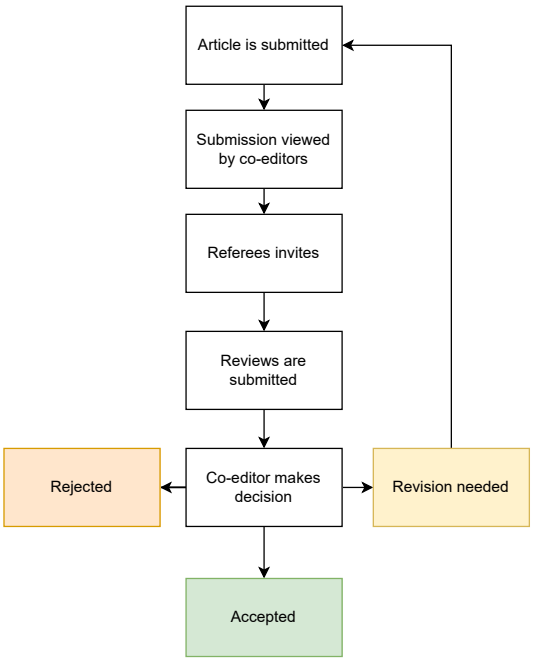

Figure 1. Peer review process diagram [IUC]

Manuscripts submitted to IUCr publications are evaluated for publication suitability through peer review. Peer review is conducted by independent specialists who evaluate each article on its own merits in order to help the editor decide whether or not to accept it for publication in an IUCr journal. In all situations, the IUCr makes every effort to guarantee that peer review is completed on time. After submitting a manuscript to an IUCr journal online, it is assigned to one of the publication's Co-editors. Authors may request a specific Co-editor at the time of submission; however,

such requests are subject to Co-editor availability, workload considerations, and manuscript subject area. Co-editors will conduct an initial evaluation of each article before requesting referees and may propose changes. Authors of publications might also be able to suggest possible referees if specific knowledge in some field is required. There are usually at least two referees and based on their comments decisions are made. If the changes are small, the updated document will normally be accepted without further examination. If the adjustments are significant, the Co-editor may request that the referees review the amended manuscript again. All accepted articles must meet the following criteria: the study must be original and new, the work must be sound and technically current, and the conclusions must be firmly based on the observations or reasoning provided. Furthermore, each work should contribute significantly to crystallography. Manuscripts are reviewed single-blind, which means that the authors are unaware of the referees' identities but are aware of the authors' names. Referee reports are anonymized to ensure that no personally identifiable information is included. In the event of a disagreement between an author and a Co-editor, the Main Editor will resolve the matter according to the author's grievance procedure. If authors are still unhappy with the judgment, they can appeal the decision to the Editor in Chief of IUCr journals for further investigation.

The problem why peer review is not enough and cannot be used as a single verification procedure is that is not completely reliable because it operates on trust that the reviewer will notice all inconsistencies. However different people may provide completely different comments even on the same publication varying from approving everything that is written to asking for raw data, further analysis, and checking all references [Smi06]. Furthermore, providing thorough, in-depth reviews is a slow and expensive process. Even though, this can be costly, the scientific community places a high priority on peer review and the publication without review would hardly be published in a journal. The peer review process can be also flawed because of the inconsistency of reviews, bias in peer review against certain sorts of authors in case reviewers and authors are not anonymized, abuse of peer reviews to get the benefit for yourself (e.g. stealing ideas and present as your own) [Smi06]. Nevertheless, even with these defects there is no apparent alternative, and scientists and editors continue to believe in peer review, it is likely to remain fundamental to research and publications as insights of scientists are extremely important in the quality of data and journals and with the help of automated tools it is possible to create more trust in the process.

3.10.2. Peer reviews in programming community

Peer reviews in programming, also known as code reviews is a manual process that involves one or several members of the team checking another member's work for mistakes that he/she made to the source code. It's a well-established and common practice in software engineering and is the key to ensuring the long-term quality of the code base and longevity of the project [KBG16]. When code review is performed correctly it can save a lot of time, and reduce the workload of quality assurance teams. Not so obvious benefit is that in the long run, it also saves money as bugs and errors can be caught during development and do not slip into production, thus do not reach the clients that use the end product which is also important for the overall product's image and customer satisfac-

tion. Furthermore, code reviews help to share knowledge and build a community of programming specialists [BCB+17] [ECN+21]. Code review is often included in the development process before the code reaches the testing environment for quality analysts or the production environment for end-user usage. There are different approaches to code reviews:

1. Email threads – one of the older approaches of code reviews that embraces email-based style. When a piece of code is ready for review, the file is emailed to the appropriate colleagues, who can evaluate it as soon as their workflow allows. While this method is more adaptable and flexible than more traditional methods, such as gathering five people in a room for a code-inspection meeting, an email thread of recommendations and various perspectives can quickly get confusing, leaving the original coder to sort it out alone [RGC+14].

2. Pair Programming – is a technique when two or more developers are put together and work on the same code side by side while checking each other's code on the go. It's a common practice to speed up the learning process of less experienced colleagues allowing them to learn from their mistakes faster. Other methods of code review, however, may provide greater objectivity because authors and even co-writers are sometimes too into their work and do not notice errors. In terms of time and staff, pair programming might also consume more resources than other methods.

3. Over-the-Shoulder – a technique similar to pair programming, however instead of writing code together, a reviewer waits for the code to be ready and then sits down to review the code while the one who developed explains how the code was written. The downside of such an approach is that it lacks tracking and documentation.

4. Tool-assisted – one of the most efficient methods to review code is to use software-based tools that solve limitations of the above-mentioned techniques as lack of documentation of defects and proposed techniques. They can also provide metrics or be used in reports for improvement processes. There are a number of tools, both paid and free that can be used to explore the code base. Usually, they have some kind of automated tests (e.g. unit tests, integration tests, or static code analysis) that run after each change that is added to the code base. Often the tests are created by developers, while for static analysis various plugins and tools can be used. These tools discover code quality defects, code specification problems, code security vulnerabilities, and invalid codes. If automated checks pass, the code can be passed to the reviewer. One of the tools that provide such a possibility is GitHub – a distributed version-control platform where people can collaborate on cue projects and share ideas [TDH14]. When the developer assumes that the code is ready, a pull request is created that has the newest changes. Code review tools are built into every pull request so team members can design review methods that increase the quality of your code while also fitting into the workflow. Developers can leave detailed comments on the syntax of the code, ask questions and provide feedback. Furthermore, all the conversation history, review changes, and all related information are saved in the history and can be examined later side by side with the original files. Similar to GitHub, other tools like GitLab and Bitbucket work. They both

also have reviews included in the development process which enables teams to collaborate and share knowledge. The process flow of these tools is shown in the image below (Figure 2).

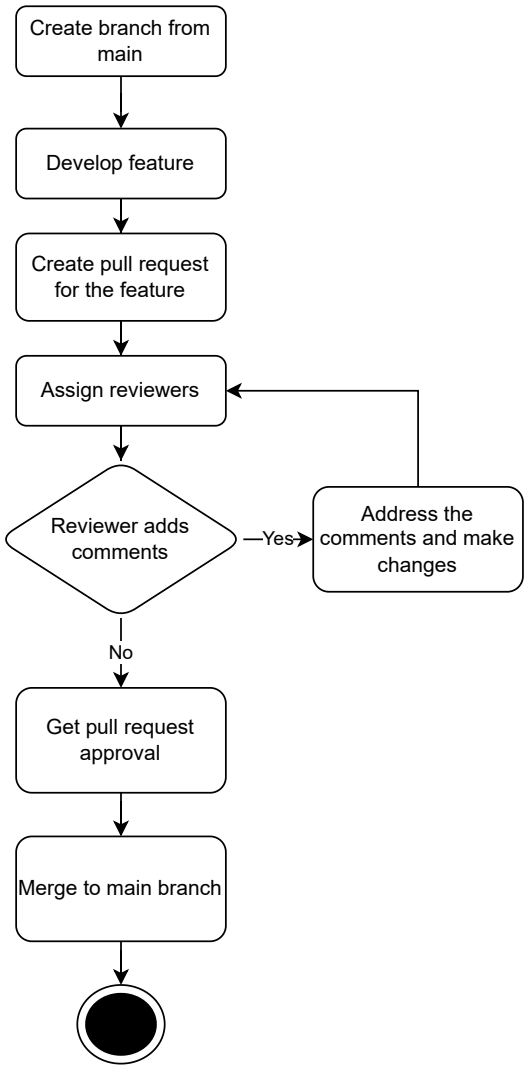

Figure 2. Code review process flow diagram

While all these tools can be used for free, they all have enterprise editions for more enhanced usage. There are other tools that are paid like e.g. SmartBear Software (https://smartbear.com) that provide thorough metrics and insights about code review in general:

1. Authors that include annotations and explanations in their reviews have considerably fewer flaws than those who do not. The cause is most likely due to authors being forced to review their own code.

2. The total time spent on the review should be less than 60 minutes and no more than 90 minutes. After 90 minutes of reviewing, the rate of defect detection drops dramatically.

3. The number of lines of code (LOC) under review should be between 200 and 400, as anything more than that can overload reviewers and drive them to stop looking for problems.

Needless to say, these metrics cannot be applied to crystallography peer reviews straightforwardly as code and publications can vary greatly in their structure and content. However, out of these metrics, some similarities can be emphasized, such as that making comments and documenting code (or in crystallography, structures, and data) makes it less prone to errors and easier to understand. Furthermore, reviewing a huge amount of code, text or data is a tiring, time-consuming process, so it should be done as a last step after all automated checks have passed. To summarise, code review, like any other review, is and should be just one part of the development process. It is, nonetheless, an essential component, since it often identifies "hidden" problems that may not present a problem now but may inhibit the product's future evolvability. There are many automated and static analysis approaches, however not everything can be spotted using them. For this reason, tool-assisted code review is the most appropriate and inclusive methodology available [BB13].

3.11. Problems encountered when managing crystallographic data

As already mention above, there are problems in crystallography that are relatively easy to check, fix and prevent like syntax errors or "honest mistakes" from authors e.g. failing to assign the correct space group to a structure. Validation mismatch errors are also quite common and can stop further work until fixed. More rare but no less serious are problems that arise from trying to publish fake data. Even though crystal data that is published is checked thoroughly by using various tools and read by reviewers, there still might be cases when errors happen or data is faked intentionally. This can cause a lot of problems as incorrect information interferes with scientific research in fields like medicine, drugs design, etc, which indirectly can correlate and affect people's well-being (e.g. pharmaceutical companies employ X-ray crystallography, an important and powerful technology in drug discovery, to identify novel treatments [ZHZ+13].) A recent study suggests, that approximately 800 fake papers might have been published between the years 2015-2022. These papers include around 1200 crystal structures in Cambridge Structural Database. Such several structures is a huge number in chemical crystallography and could mean that not only this database but possibly others as well, are filled with incorrect data. Indicators that may reveal frauds can be metal or element swapping in coordination complexes and organic compounds, duplicated figures, data sets with identical parameters, strange linguistic occurrences like using 'logistical growth phase/phage' instead of 'logarithmic growth phase', and other grammatical inconsistencies, ethics-approval anomalies, meaning that papers were not approved by committees or were approved by committees that do not exist, email anomalies like using fake emails addresses, duplicate, ineffective references, plagiarism [Bim22]. Investigations have shown that even structures that do not exist can be composed and put for deposition [Har96]. While syntax and semantics can be checked rather easily by using automatic (and manual) tools like parsers and other specialized software, they also have less effect on data usability as they result in unreadable files and incorrect supporting information and most probably would not go further until these issues are fixed or explained. However, crystal structure error detection is far more complicated as it is mostly done manually and may need to be reviewed by several people while physically and independently recreating an experiment remains to validate data. Furthermore, sometimes it is not enough to only have crystallographers reviewing the data.

Scientists from various other fields like physics, chemistry, and so on, can also make a huge impact and discover anomalies, so having and growing scientific community is an extremely important factor as well.
Frauds and retractions of publications in crystallography are not only scandalous but also need many resources in terms of time, finances, and human effort. These frauds confirm that programs and tools that validate the syntax of crystal structures are not sufficient to prevent erroneous data that is published in both scientific journals and databases and while crystallographic information is also reviewed by humans, in-depth analysis is time-consuming. Although many procedures in crystal structure research have been automated in recent years, interpreting tiny details in structural data still takes a significant amount of human talent and expertise. As a result, subjectivity is unavoidable in this process, and various people working with the same data may produce somewhat different conclusions from time to time. So new approaches to connecting these two processes are needed to improve the correctness and quality of data and minimize the issues created by frauds, which most probably cannot be eliminated, but would undoubtedly improve the whole publishing pipeline.

3.12. Solution methods

While peer reviews are one of the most important parts of ensuring the quality of the data, there is a need to have automated methods that would support manual reviews and ease the work for scientists. In this section, some of these methods that could be used will be discussed.

3.12.1. Digital signatures

Digital signatures can be used to achieve data integrity in crystallography by enabling researchers to verify the authenticity and integrity of the data. When a digital signature is applied to a file, it creates a unique code that is based on the contents of the file. If the contents of the file are changed, the digital signature will no longer be valid. This can help prevent tampering with the data and ensure that the data is accurate and has not been altered [GMR+01]. To use digital signatures in crystallography, researchers can sign their data files using a digital signature application or service. The signed data files can then be shared with others, who can use the same digital signature application or service to verify the authenticity and integrity of the data. This can provide a high level of confidence in the accuracy and reliability of the data and can help prevent fraud in crystallography.

3.12.2. Function allocation

Human roles, duties, and tasks are typically decided implicitly throughout the design phase through the selection or development of equipment and software. While this approach is rational in the sense that mechanization is usually helpful, such decisions can overlook the systematic assessment of human skills and limitations and how they affect system performance. The process of selecting how the system functions will be implemented – by humans, equipment, or both – and assigning them accordingly is known as function allocation. It strives to find a compromise between attempts

to mechanize or automate as many system operations as feasible by identifying roles and activities for humans that best utilize their strengths while avoiding human limitations [WDF00]. The term function allocation was introduced by Paul Fitts with the idea that if a man is superior to the machine or vice versa, tasks could be split and assigned accordingly [BS96]. The idea was that functions should accompany humans in tasks where it is more efficient to use systems rather than human work. Since P. Fitts's research, several new perspectives were introduced:

1. comparing human and machine performance;
2. comparing human and machine economic costs;
3. developing tasks to make use of complementary human and machine qualities;
4. grading human jobs to account for individual variances;
5. basing human duties on system functions and complementing them with machines;
6. allowing humans to alter their level of involvement in the system through flexible delegation of computer resources.

Furthermore, several characteristics can distinguish humans from machines as they have different strong sides. For example, humans are superior to machines in inductive reasoning, making judgments and decisions, keeping information for a long time and memorizing relevant portions, improvising, and using flexible approaches if needed. Meanwhile, machines are one step ahead of humans in terms of performing many complex operations simultaneously, storing information briefly and deleting it completely, quickly responding to signals and applying results precisely, data processing, and repetitive tasks. Machine work is also usually more predictable as given the same inputs, it will always give the same outputs, meanwhile, humans may disagree with one's opinions, and be mistaken in some cases, which would result in different results. Based on this knowledge, degrees of automation of decision-aid was established by A. Sheridan in 1992, where degree items can be either ANDed or ORed as provided below:

1. Offer no assistance to the system, the human makes the decision.
2. Provide the operator with a full range of options, AND
3. reduce the number of options to a small number, OR
4. suggests one of the possibilities, AND
5. execute the suggestion if the person approves, OR
6. allows the person to veto the recommendation before it is executed automatically, OR
7. informs the person after execution, OR
8. informs the person after execution if asked, OR

9. informs the person after execution if the system decides to.

10. Make decisions without consulting a person and act independently. [BS96].

With each step, the level of decision authority is increased. Since the tasks that are automated were previously done by humans, increasing the authority of machines to make decisions, decreases the need for human involvement. For example, the first step uses no automation and the human takes full control, while in the very last step, the human does nothing while the machine makes every decision. Such a concept of a level of automation can be used to solve many problems when human-system task delegation is needed. Furthermore, such flexibility between humans and automatic systems brings many advantages like greater situation awareness, more precise automated utilization, a more balanced mental workload, user acceptance, and overall performance. In the interim, the system needs less and less approval before taking action. In other words, the system is given greater amounts of autonomy in making and carrying out decisions.

3.12.3. Neural networks and deep learning

To minimize the issues that arise when creating a balanced human-machine evaluation system, the process could be defined by using the data which was already reviewed and scientists know what result should be provided, using the data as an input for machine processes. Another more enhanced way would be to use neural networks and deep learning techniques. This approach is already widely used in many scientific fields and could be successfully applied to crystallography as well. An artificial neural network has an input layer of neurons, one or two hidden layers of neurons, and a final layer of output neurons [Wan03]. The process of learning usually consists of preparing the data, performing transformations, normalizations, etc. The next stages comprise an iterative process in which a machine learning algorithm is built, trained, optimized, validated, and selected for a specific problem. The concept of an artificial neuron is based on biological neurons, where each neuron has an input that provides single output. To get the final result (output) weighted sum of all inputs is taken, including data from hidden layers which is used to allow more complex solutions. In our case, inputs could be CIF files from the COD. Based on that network could be trained to recognize correct and faulted structures and provide results to the scientists so they can decide on structure quality. Deep learning models' major fault is that they learn by observation. This means that they only know what was in the training data. If a user has a little amount of data or it comes from a single source that isn't always indicative of the larger functional area, the models won't learn in a generalizable way. From the perspective of this thesis, neural networks will not be used as before CIF files could be used to train the network, they would need to be normalized, vectorized and crystallographers are not sure what the appropriate structure should be used as many factors need to be taken into account (e.g. be independent of turning point, because if you turn the atoms in a different position you will get completely different structure). Furthermore, deep learning also demands a large amount of data and as of now, we do not have data that would be ready for that.

3.12.4. Decision support systems

Decision support systems are systems that consist of a collection of information-processing components and unify machine and human decisions to enable complex solutions that require inputs from both people and automated tools to reach the goal. It helps to manage the workload that is put on the people and allows them to perform critical tasks that cannot be performed by a machine. However, when a person is involved in the decision-making process, it is important to find the right balance between automation and human control, meaning that the process needs to be automated, but a person can make influence the decisions. One way to approach that would be to introduce procedures that define the power of decisions that are given to humans and machines. Such a method of human-machine interaction is known as Levels of automation (LOA) and was initially introduced by Sheridan and Verplank in 1978 [JMR+17]. Levels of automation framework ideas are based on traditional systems engineering framework and decision-aid degrees that were described above, however, there can be many possible interpretations of how human activity should be involved. To simplify the task Levels of Human Control Abstraction (LHCA) were introduced to provide a more human-centric instead of system-centric approach. With the increase of human-level abstraction, less operator input is needed. The framework provides five levels:

1. Direct Control – operator has full control of the system;
2. Augmented Control – operator gives inputs so that the system could make decisions;
3. Parametric Control – operator specifies the intended parameters for the system to meet, and the system uses onboard sensors and control algorithms to achieve those goals. The parameters that the operator gives are discrete and the human is also responsible for making sure that the system is acting correctly. A real-world example could be a plane with autopilot, where the pilot (human) still has to enter factors like altitude, airspeed, and other factors that are needed to ensure safety;
4. Goal-Oriented Control – human gives inputs for expected results and then the systems make all decisions so that the result would be met;
5. Mission-Capable Control – pre-launch task goals are given by the person. The person does not need to monitor the process as the system starts autonomously and independently.

[JMR+17]

As determining which LHCA level is used can be complicated at some points, the LHCA decision tree was introduced to help determine the current position. The tree provides a questionnaire to reach the answer. A visual presentation of the questionnaire is provided below (Figure 3):

LHCA framework describes human and machine interactions and helps to understand how these tasks could be assigned from one to another. While this framework was primarily created for vehicle machines [JMR+17] it could be extended to suit other domains like crystallography. Additional investigation of levels, tasks, and boundaries for the system and human interaction is

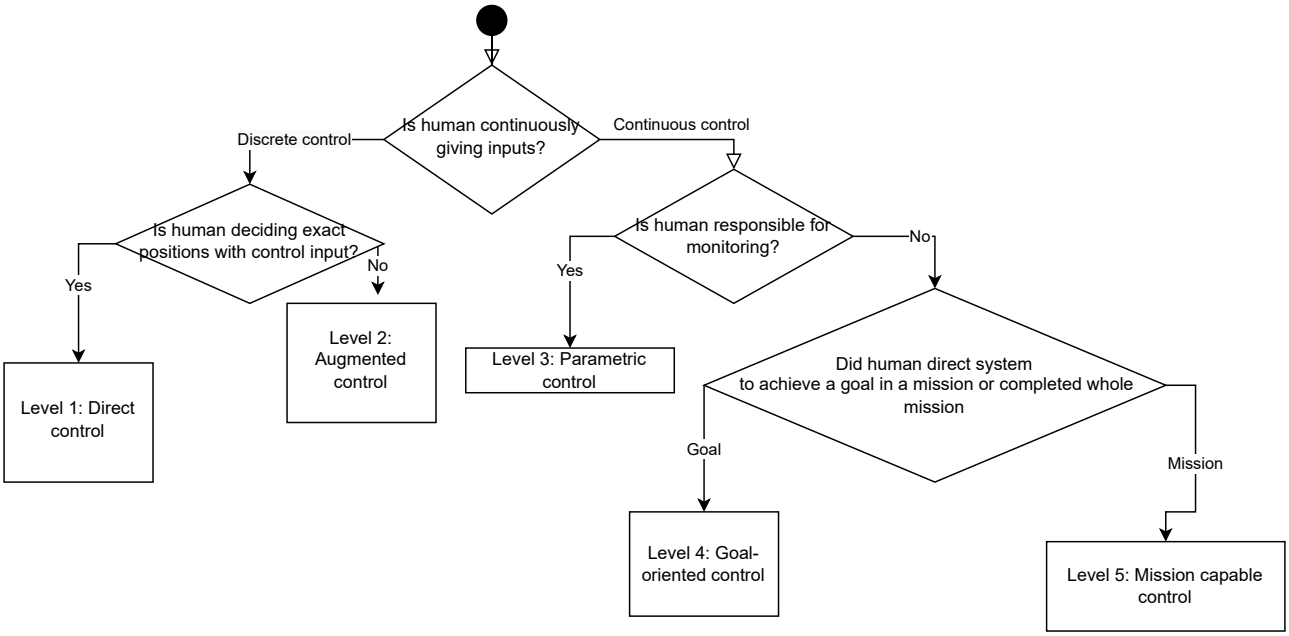

Figure 3. LHCA decision tree

needed to adapt this framework suitable for validating crystal structures and will be done in further work.

In conclusion, when it comes to integration between human input and automated tools, decision support systems emerge as an effective and promising solution. While digital signatures, neural networks, and deep learning were considered potential methods, the decision support system seems to be the best option for several reasons. First of all, decision support systems can help with managing data which makes the review process more effective. Additionally, decision support systems offer customizable and flexible approaches to structure review. Also, for different data, strictness of decision might be needed and a decision support system can be made to accommodate these variations and adapt to different scenarios. Secondly, decision support systems integrate human knowledge with automated tools. This is important because crystallography consists of complex data and usually requires specific knowledge and experience.

4. Solution

4.1. Constraints

Based on the sections above I identified these problems as the most critical:

1. Time constraints – scientists' time is an expensive and valuable resource and can be a significant challenge in the review process. When under time pressure, there may not be enough time to carefully consider all of the options and scientists may end up making rushed or hasty decisions. By integrating multiple validation tools I will point out the potential problem to make the review process runs smoothly and efficiently.

2. Software constraints – already existing infrastructure will be used, so implementation of services needs to be compatible with it and able to integrate existing tools, therefore available to include more tools in the future. A new database needs to be created and services have to be deployed on Linux servers.

3. Decision-making constraints – several factors can impact decision-making in the review process for crystallographic structures like limited information, conflicting viewpoints, and personal biases. To mitigate that problem our automated tools will gather as much information as possible to help reviewers make decisions. This will help to reduce the impact of limited information and increase the chances of making a good decision.

4. Emotional/ethical constraints – Emotional and ethical constraints can play a significant role in the review process for crystallography research. Reviewers are expected to approach their work objectively, without letting personal feelings or biases influence their evaluation. I will apply the *wisdom of crowds models* (will be explained later) to minimize the impact on the evaluations.

5. Frauds/honest mistakes – probably the hardest problem to tackle. Contrary to syntax errors or validation mismatches which are relatively easy to manage because of strict structure, fraud, and honest mistakes can be hard to find. Even though it is nearly impossible to guarantee that there will be no fraudulent structures, there are steps that can be taken to minimize the risk. Automatic tools will either skip incorrect structures or help to decide by providing checks scores and information/warning/error messages, therefore, saving time for reviewers to concentrate attention on a smaller number of structures that need investigation.

4.2. Automated tool integration

Scientists (depositors) that want to upload their structures to Crystallography Open Database can deposition interface which allows authenticated users to upload, edit, validate, and place their CIF files. The process consists of the following steps:

1. Depositor logs into the system (signs up if does not have an account yet) and chooses deposition type from prepublication data, already published data, or personal communication to COD and begins data deposition.

2. Complete bibliographic information, including author names, journal names, publication years, volumes, issues, and pages, should be included in published structures.

3. Depositor can either select CIF or ZIP files to upload to the server for the check.

4. To ensure that the provided files contain all the needed information, COD's scripts run several checks.

5. An erroneous file can be updated and fixed in the browser window and checked once again. The following step can be rerun as many times as it is needed.

6. If files pass validation they are deposited to the database and can be accessed via the web interface.

A similar approach is being used for the new system. The author will upload the file and the system will run the checks. If there will be inconsistencies, the system will print them out and stop further steps until they're eliminated.

4.3. Automated tools selection

Based on discussed tools, below I added a summarised table of some relevant existing tools and their usage:

CIF validation						
Application/Tool/ Solution	**Integrated in COD**	**Syntax validation**	**Semantic validation**	**Validation according to dictionaries**	**Can be integrated in the new system**	**Will be integrated in system**
EnCifer	No	Yes	Yes	Yes	No, EnCifer is managed by The Cambridge Crystallographic Data Centre	No
checkCIF	No	Yes	Yes	No	Currently provides different form of severity levels. Severity is defined as Level A, B, C alerts	No
cif_cod_check	Yes	Yes	Yes	No	Yes, output messages are provided in strict structure which can be parsed. Three severity levels available. Provides script options on how it should be terminated which will be useful	Yes
cif_validate	No	No	No	Yes	Yes, output messages are provided in strict structure which can be used as feedback.	Yes
Machine learning	No	Not possible to apply machine learning or neural network solutions at this moment as it is currently unknown how CIF data should be prepared so that it could act as input for training data and it is out of this thesis scope.				No

Figure 4. Tools comparison

To fit into time constraints and be able to ship the first version of the system to gain a better understanding of how users feel about the system, a decision was made to integrate tools that are already used in COD. Checks will run automatically after the author uploads the required files. All validation tools are stored on-premises on a Linux machine. Services will be implemented to call the tools from within a script to automate the validation of CIF files. After each script run, results will be aggregated and shown to the users.

4.3.1. Grammatica

Grammatica is a syntax analyzer program written in JAVA using a BNF parser generator. BNF (Backus normal form) is a metasyntax notation for context-free grammars, often used to describe the syntax of languages used in computing, such as computer programming languages, document formats, instruction sets, and communication protocols, and intended for human consumption. The Grammatica program provides error messages with different severity indicator levels:

```
BIT CONSTANT    DESCRIPTION
1   ERROR Super-severe message, meaning general fault,
          which could not be recovered.
          Must be fixed if returned by validation script.
2   WARNING Severe message means some serious problem in input data,
          but the script must not terminate if experienced with the error of
          his level. It should be taken into account.
```

```
4   NOTICE Notice might not be a real error, but in some conditions, this
           could indicate a real problem incorrectly captured.
           Could be skipped if there are a lot of warnings.
8   I_ERROR This is an information message. It is a process indicator and
           An ERROR message must follow this one.
           Process indicators are displayed during normal script execution,
           if -v (verbose)
16  I_WARNING This is an information message about an upcoming warning.

32  I_NOTICE Information notice could not be related to NOTICE, but
           in most situations, it should. Exceptions could be made if
           during the script run something s-e-e-m-s as a NOTICE, but after
           full analysis, it appears as a normal situation.
64  I_INDICATOR This is not an error message entirely. It is progress
           the indicator showed only while in DEBUG mode (or full verbose)
           and writing messages about the current script progress.
65536   ALL means that the script should output ANY actual information,
           almost every tick of progress.
```

The result of an executed check comes in a format where the first column is a command that was executed, then goes either file or status followed by an explanation message. The actual result can look like this:

```
scripts/cif_fix_values: tests/cases/cif_fix_values_116.inp data_2:
           NOTE, '_chemical_melting_point' value '251 -- 254  0C' was
           changed to '525.6(15)' -
           it was converted from degrees Celsius(C) to Kelvins(K),
           the average value was taken and precision was estimated.
```

The breakdown of the first line with explanations:

1. scripts/cif_fix_value – script that is executed
2. tests/cases/cif_fix_values_116.inp – input file to checked
3. data_2 – ID of structure
4. NOTE – severity level
5. '_chemical_melting_point' value '251 – 254 0C' was changed to '525.6(15)' – it was converted from degrees Celsius(C) to Kelvins(K), the average value was taken and precision was estimated. – explanation of parser result.

The advantage of having such standardized grammar is that it can be easily parsed and used in further aggregations. Different types of severity levels can be summed up and provided to the author

and reviewer. If there are severe exceptions, the program can be terminated until the author fixes these exceptions. Since there are different types of severity levels, they can be used to introduce boundaries that are flexible and easily adjusted, meaning that experts of the system could decide what amount of errors and warnings are critical or whether no human review is needed if no errors were returned. Out of all records that are currently in COD majority of them (495366 out of 479630 [COD]) have some kind of message, not of them are fatal, but the number is too big to look at each structure manually. What was done haphazardly, in the beginning, is, in many cases, already irretrievably lost, so being able to review and discuss structures by community upfront would reduce such numbers in the future. Based on this grammar, I implemented the CIF checks results in the parsing service, which takes script output as a parameter and parses the result to a model:

```
public static CodCheckResults ParseCifOutput(string output, string fileName)
{
    var result = new CodCheckResults
     {FileName = fileName, Checks = new List<CifCodCheck>()};
    var lines = output.Split(
     Environment.NewLine, StringSplitOptions.RemoveEmptyEntries);

    foreach (var line in lines)
    {
        var parts = line.Trim().Split(':');
        if (parts. Length < 3) break;
        if (parts[1].Trim().Split(' ').Length < 2) break;
        if (parts.Length != 3) continue;

        var levelString = parts[2].Split(',')[0];
        if (!Enum.TryParse<Level>(levelString, true, out var level)) continue;
        var message = line[(line.IndexOf(levelString, StringComparison.Ordinal)
                            + levelString.Length + 1)..].Trim();
        result.Checks.Add(new CifCodCheck {Level = level, Message = message});
    }
    return result;
}

public class CodCheckResults
{
    public string FileName { get; set; }
    public ICollection<CifCodCheck> Checks { get; set; }
}
public class CifCodCheck
{
```

```
    public string Message { get; set; }
    public Level Level { get; set; }
}
```

The code is executed every time the structure is uploaded. After the upload check execution is started and output parsed. Results are saved in the database as well as used for calculations when checking whether the structure can be reviewed and deposited to COD.

4.3.2. CIF COD check

cif_cod_check – is a tool that parses a CIF file and checks if certain data values match COD requirements and IUCr data validation criteria. The tool can be executed via terminal and has multiple option flags that are useful when deciding how the program should be executed. Below some options on how cif_cod_check can be terminated are provided:

```
OPTIONS:
  -c, --always-continue
                    Continue processing and return successful return status
                    even if errors are diagnosed.
  -c-, --always-die
                    Stop and return an error status if errors are diagnosed.
  --continue-on-errors
                    Do not terminate the script if errors are raised.
  --die-on-errors
                    Terminate script immediately if errors are raised
                    (default).
  --die-on-warnings
                    Terminate the script immediately if warnings are raised.
  --die-on-notes
                    Terminate the script immediately if notes are raised.
```

These flags allow us to easily tighten validation or vice versa and filter whether the structure should go for further checks or stay on the author's side for additional fixes and improvements.

To use cif_cod_check, the path to a CIF file needs to be provided as a command-line argument. For example:

```
cif_cod_check file.cif
```

The parsable output that looks like this is printed:

```
cif_cod_check: 200657.cif data_200657:
 NOTE, data_item '_refine_goodness_of_fit_obs'
 value '0.71' lies outside the range [0.8, 2]
```

The breakdown with explanations:

1. /usr/bin/cif_cod_check – script that is executed
2. 200657.cif – input file to be checked
3. data_200657- ID of structure
4. NOTE – severity level
5. data_item '_refine_goodness_of_fit_obs' value '0.71' lies outside the range [0.8, 2] – explanation of note.

4.3.3. cif_validate

To use cif_validate, the path to a CIF file will be provided as a command-line argument. For example:

```
cifvalidate file.cif
```

cif_validate will then parse the CIF file and check it for errors. If the check succeeds, it will print a message indicating that the CIF file is valid and exit with a zero exit code and the system will show the green light for the next steps. Here is an example of the output you might see when running cif_validate on a valid CIF file:

```
$ cif_validate: 200657.cif data_7159242: NOTE, definition
of the '_cod_data_source_block' data item
was not found in the provided dictionaries
```

The ability to parse both cif_cod_check and cif_validate results in the same format and reuse code in `C#` is a time-saving and efficient approach as well as allows to cover the automatic validation part of the system.

4.3.4. Decision making

To ease decision-making for the reviewers, I introduced different scales which will allow administrators to manipulate different settings of validation tools scripts:

1. Severity–based – based on the output which is given by automated checks reviewer three severity levels are introduced: Note, Warning, Error. Information will be aggregated and shown both to the author and reviewer. By default, none of the structures that exit with a status error will reach reviewers.
2. Selection–based – this scale is used for the cif_validate tool when performing checks against dictionaries. It will be the last automated check that is performed and will give a comprehensive summary of detected inconsistencies. This scale will be more advisory for the reviewer as not all messages are equally important. Some dictionary checks can fail depending on the dictionary version and can be ignored while some might mean that structure is invalid. It will

provide a list of messages and the reviewer will go through each one and mark it as critical or not. These scales will allow us to reduce the number of structures that need review and provide a decision aid for the reviewer.

3. General settings – these settings provide information on how uploaded structures should be manipulated in general. They consist of:

   (a) publish_ after – a flag that describes when an automatic decision should be made if neither author nor reviewer has not decided in a period. This flag consists of an enum interval (day, week, month, year) and is followed by an integer, e.g. 1 week would mean that if no decision was made after the last structure change, the automatic decision should be made and the structure will either be approved and moved to COD or rejected.

   (b) approvals_required – describes that the structure cannot be moved to COD by the author if it does not have approval from the reviewer.

   (c) number_ of_approvals – describes the number of approvals needed for the structure to be moved to COD. Takes effect when approvals_required is set to true.

   (d) automatic_decision_enabled – if set to false, the system will not deposit structures if there's approval missing.

Usage of specified general settings is implemented in `StructureUpdaterService` background service that runs continuously and updates the status of the structures in a structure review system every hour and deposits structures to COD if they can be deposited. It does this by retrieving the structures and system settings and then checking the status of the structure which should not be rejected, already in COD, or exited with error status. If the structure passes this check, then the time constraint is checked. If the elapsed time since the last update exceeds or is equal to publish_after then the structure can be moved to COD. A diagram of the background service's activity flow is shown in Figure 5.

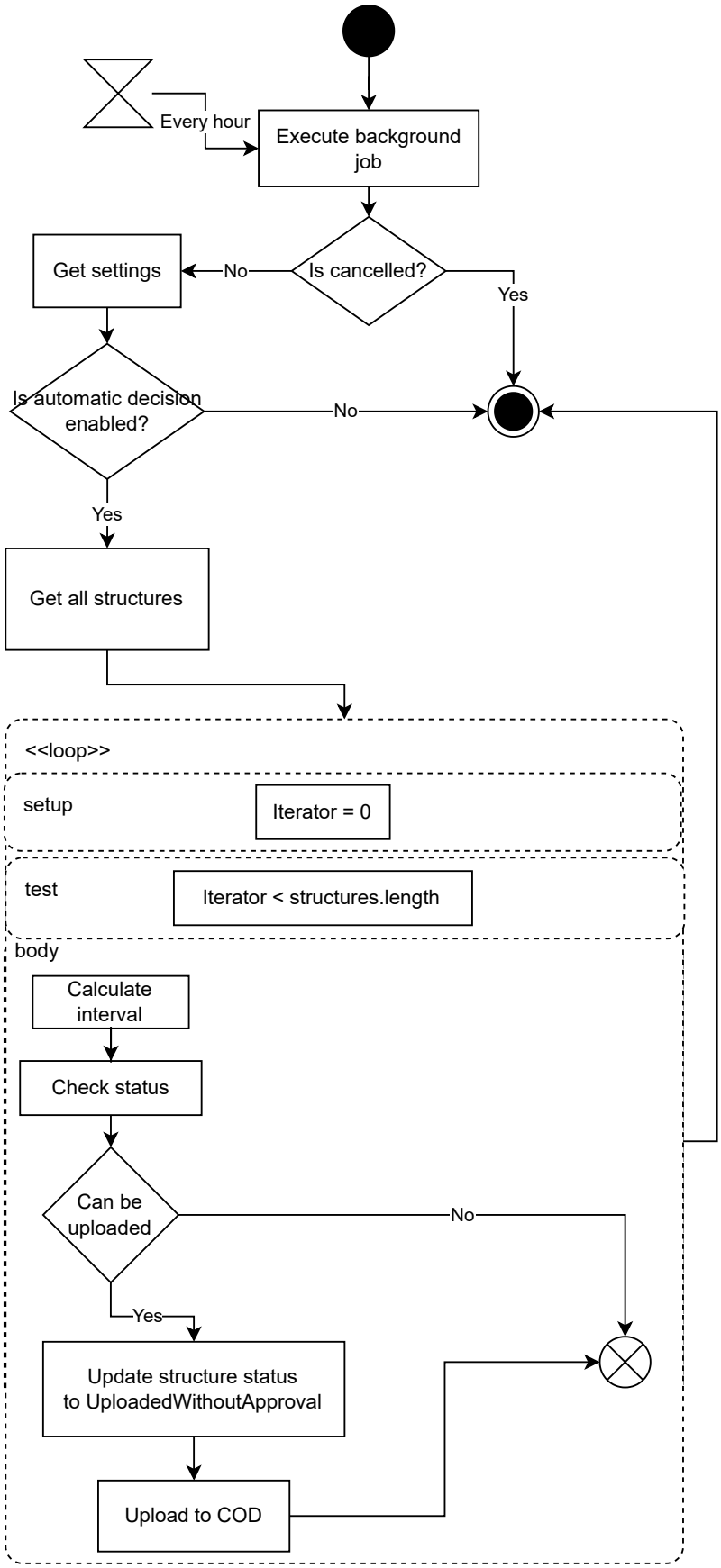

Figure 5. Background service activity diagram

4.4. Organisation of information

Collected information on structures, files, and anomalies are described in a relational database that supports SQL queries, thus facilitating the search and selection of the required information. The decision to use a relational database was made as COD already uses such type of database. A record versioning will be implemented in the database to ensure tracking, reproducibility, and changes in structure information.

4.4.1. Candidates database schema

For collected information tables 'users', 'reviews', 'structures', 'comments', 'checks', and 'statistics' are created. Each table has a unique alphanumeric identifier that makes it possible to address that entity to access it and interact as well as a recorded column to know when every change to that table was made. The 'Users' table has information about users and whether the user is a candidate, judge, or admin. The 'Reviews' table holds identifiers of the structure and the person who is making a review as well as a boolean value of whether the structure is approved. All information that the user uploads is stored in the 'structures' table. This table has columns 'content', 'status' and 'n_cod', 'version'. Each structure can have different statuses defining the current state, e.g.: status 'failed' imposes that the structure did not pass automated checks and needs to be fixed by the author, and status 'approved' means that the structure is accepted as valid and can. be moved to COD, the status 'PushedWithoutApproval' indicates that the structure passed syntax checks, however, because of some reasons (validation according to dictionaries misalignment or reviewers' comments and discussions) is not approved, despite the author choosing to put the structure in the database. Column 'in_cod' indicates whether the structure exists COD. The comments table tracks discussions and comments made on the structure. It has identifiers of structure and the commenter, the body of the comment which contains the text. Besides these columns, it also has column 'line' which indicates that the comment was left in a specific place of the structure. If 'line' is equal to NULL, that means that comment was left on the overall structure, not on a specific part of it. The 'Checks' table will contain the identifier of a structure, version, and content of error messages. Some generic tables are not associated with one another but hold important information regarding system settings. Table 'structure_display' has information that is needed to view CIF structure data in web applications in a human-readable format. Since CIF files may consist of a large number of different labels with values, system owners and scientists may want to show some properties more than others. For this reason, these sets of data names can be added to the database with provided regex combinations on how each value should be displayed. Table 'cif_cod_flags' holds settings that are needed to manipulate default script running settings, while table 'settings' contains values that overwrite defaults for general settings of the approval and review process. It contains columns for scale flags on where automated checks should fail, and the number of approvals required before the structure can go to COD (defaults to 1). An additional boolean flag 'is_automatic_decision_enabled' column exists to allow/forbid automatic decisions made by the system. The database diagram is visualized in figure 6.

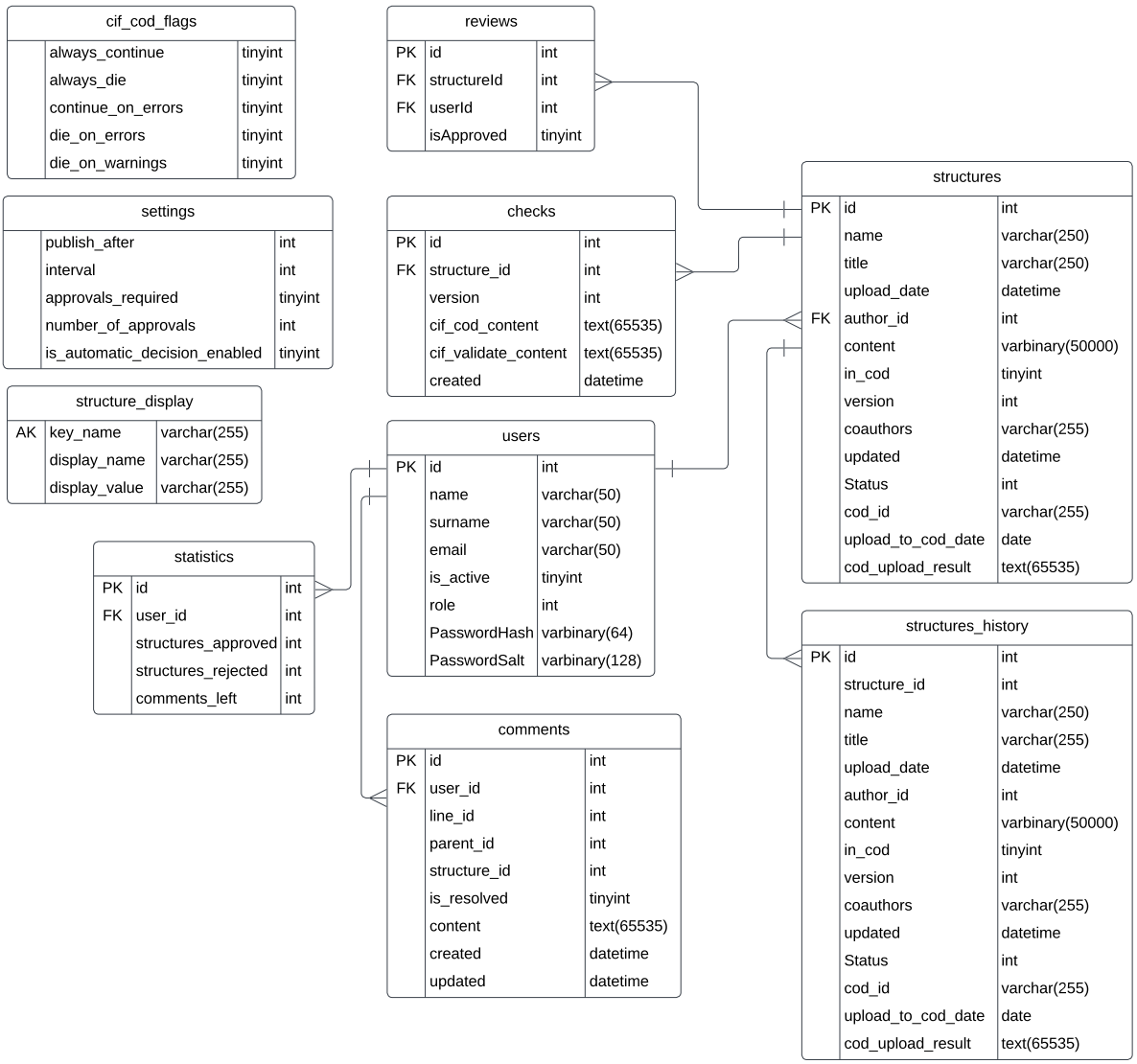

Figure 6. Database diagram

4.4.2. Versioning

To ensure reproducibility of structures and track their changes I implemented the versioning system in the database. Each structure has a column 'version'. When someone requests what changes were made to the structure, the system gets the required versions, calculates diff between these two versions, and shows it to the user. Since each structure can have many versions in order not to flood the table with tons of data, only the last versions will be stored in the 'structures' table. Older versions, as well as deleted (rejected) structures or the ones that were approved and saved in COD, will be moved to another table called 'structures_history'. This is done when a user uploads newly updated data of the structure, old data is moved to the history table and the current model's version is increased as well as the content and update time is modified.

4.4.3. Emailing

To make the process faster, the system will send users relevant information via email. The following emails will be sent:

1. Comments and discussion updates
2. Structure changes
3. Decision (approved/rejected/pushed without approval)
4. Reminders of the end date to make a decision

4.5. Requirements

1. Changes of structure changes can be viewed
2. Adjustable and flexible decision-making scale (system owners can decide when tools should continue/exit)
3. Scalable – the program can be extended with additional validation tools
4. Decision must be made either way. In the best scenario, the reviewer should make a decision, however, there might be cases when the author will still want to deposit structure even without the reviewer's approval. In that case, if there're no critical errors author can submit his structure. If a human (reviewer/author) does not make a decision, the machine (system) should make a decision based on automated check results after the time to make the decision expires.

4.6. Criteria for reviewers decisions

As already discussed in previous sections, reviews by specialists are a crucial step in ensuring data validity and quality. However as human time is a valuable and expensive resource, it is important to determine the sufficient number of opinions needed and how to guarantee that decisions and judgments are faultless and independently given. The term called the wisdom of crowds states that the accuracy of social decision-making and prediction can be increased by the collective intelligence that results from several human or computer responses to the same queries when calculating each person's level of inaccuracy and reducing the total error in the crowd agreement [WM19]. This theory is applied in a wide range of sciences starting with politics (e.g. forecasting elections [Mur17]), philosophy, statistical and sociological contexts, and medicine (e.g. predicting patient survival [WM19]). However, to benefit from crowd wisdom several conditions like opinion diversity, independence, decentralization, and aggregation must be met (Figure 7). The whole decision-making and evaluating process usually takes time as individuals need to make their own decisions first, after that crowd intelligence can be applied and decisions can be made. Over time, judgment criteria which are used to evaluate decisions can be improved by using results from aggregated answers from individuals.

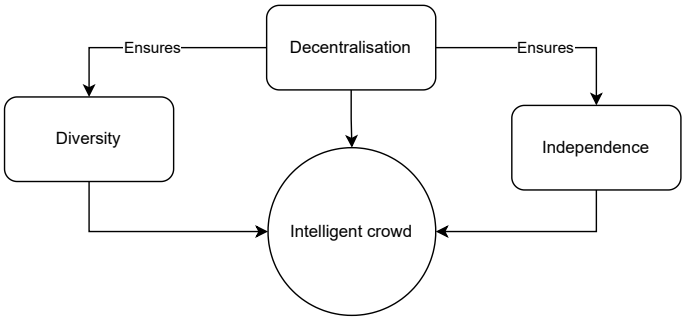

Figure 7. Characteristics of crowd intelligence

4.6.1. Diversity

To ensure the correctness of crowd judgment it is first needed to gather a group of individuals that have expertise in that specific area where decisions are needed. It goes without saying that a larger judgment pool is preferable because it increases the likelihood that a wise decision will be made. Because analysis and thought are the foundation of justice, it is essential to have experts with a range of experiences and knowledge. Individual differences guarantee a variety of information and judgment. However, it can be a problem to find ways to extend the group of people that are making these decisions in a specific domain area. With the new system, our goal is to attract as many scientists as possible which will be done by using the new system as a top layer for the already existing COD infrastructure, meaning that all authors that previously uploaded their files to the COD website will need to use the new system instead.

4.6.2. Independence

In order to ensure that an incorrect opinion of a person would not make an impact on other people's opinions it is important to guarantee that each individual makes a judgment independently and is not influenced by his peers as in the aggregating process, poor judgments may work against each other. Having to decide on his own, the individual tends to explore new information which is beneficial for extending judgment criteria, whereas if the decision is made in public it can be contemplated by the opinions of others. To fulfill this condition, a method that tracks the number of reviews out of reviews needed is implemented. This way, one reviewer will see another reviewer's comments only after they both left their first insights. A settings variable 'publish_after' is introduced to ensure that in case some reviewer does not add his/her comments, the comments of others would still appear public after some time. Several reviewers needed can also be adjusted by modifying the variable 'number_of_approvals'. At the first stage of the system, all uploaded candidate structures are shown to reviewers and each reviewer can decide what structure he/she wants to review. Later on, a different approach might be used, e.g. candidate author could assign reviewers or select additional reviewers in case the author thinks the opinions are biased, as well as reviewers could be chosen and assigned automatically. The first approach of allowing reviewers to select which structures they want to review has several benefits. From a technical perspective, it's the least effort to implement this solution which allows creation of minimal viable products faster. Furthermore, even more, significant aspects are that this approach allows us to ensure that the reviewer is inter-

ested in a particular structure and will not rush through the assessment, but give detailed and more thoughtful reviews. Additionally, this could help the review process to be more efficient because if the reviewer shows interest in the structure he/she might be more likely to prioritize that review over other work and help the review to be finished promptly. Finally, this approach should also help to assure that work responsibilities and load are fairly distributed as some structures can be more difficult or time-consuming to review and reviewers can have different expertise in different structures. Therefore, permitting reviewers to choose the structures they want to review can help ensure that the workload is distributed according to their interests, abilities, and experience.

4.6.3. Decentralisation system

The centralized system might influence people to follow the decisions of other powerful individuals, therefore it is important to ensure that individuals are keen to look for diverse judgments and have individual independence because only then final crowd's decision can be aggregated accurately. Meanwhile, a decentralized system ensures that there's no center of power that could misrepresent results [LLC+17]. This will not be covered within this master thesis work.

4.6.4. Aggregation

Aggregation is a method of combining information that comes in the form of various data into a collective output. One of the most common methods of aggregation is calculating an average. The average can be unweighted, simply taking the sum of values and dividing by the number of people.

$$UnweightedLinearAverage = \frac{1}{N}\sum_{i=1}^{N} j_i \tag{1}$$

1. N – the total number of people
2. j_i – individual estimate (judgment)
3. I – the number of each individual

Another approach is to use a weighted linear average by providing some weight next to a value. These approaches are more beneficial if there's not enough information about the individuals in the group or if some individuals in the group are more reliable, and have more knowledge and experience than others. However, the calculation of averages has disadvantages as well, as the decision can be inaccurate if judgments are not distributed around central value [LP13].

$$WeightedLinearAverage = \frac{1}{N}\sum_{i=1}^{N} w_i j_i \tag{2}$$

The main idea behind weighted average is that you should calculate weight based on past performance or whether these people are confident in their decisions. The difference from the first formula is that here we add parameter w_i which is the weight of a judgment (reviewer's response). Another more complicated method is the respected weight average. The weighted average can be

calculated with more difficulty by collecting ratings of peer esteem in addition to the evaluations. An example to illustrate that could be having a group of experts of different levels. All levels of experts are welcome to share their opinion, however, the judgment of more junior peers can sometimes be less trusted due to their lack of experience. However, if each individual has some kind of trust level (weight) it can be taken into account while making a final decision. Each individual could give votes for the person they respect/trust and aggregation of these values would then give a new judgment value for each person. For individual k, if they trust a person I to the degree w_ki the calculation looks like this:

$$RespectedWeightedAverage = \frac{1}{N}\sum_{i=1}^{N} w_{ki} j_i \tag{3}$$

After that, an average can be made from these values. However, this technique of aggregation may have a downside in that people's assessments of one another's level of skill may not track the accuracy of their judgments, meaning that one person's evaluation of another individual might not be always correct as well [WJ04]. One more difficulty with such a method is that looking from a social perspective it can be hard for people to rate each other's expertise, even more so if it is a public process [LP13]. Therefore it is important to ensure that these evaluations do not decrease experts' motivation to participate in the voting and reviewing process. After considering these approaches, Respected Weighted Average was chosen as the most suitable as it allows to have initial trust value which is important to differentiate between scientists' experience levels.

4.7. Voting systems and rankings

The idea of voting on both individuals and their thoughts is not new and is applied to multiple websites like e.g. Reddit. It is a news aggregation, discussion, and content-rating website. The website's material, which includes text entries, photographs, and videos, is contributed by registered users and is then rated by other users. Posts are arranged into user-made boards known as "communities" according to their subjects. When there are enough upvotes, posts that have received the most upvotes will eventually appear on the first page of the website. The communities on Reddit are moderated by administrators. Items are ranked based on the number of votes they receive and the post's age about other posts. A similar approach to crowd judgment and curation is used in Stack Overflow. With the help of volunteers from all across the world, it is a community-based question-answering service for developers and software engineers that compiles vast amounts of knowledge [AMR+13]. It allows users to ask and answer questions on topics in computer programming. It has a voting system that helps the website's community to identify which questions and answers are most useful and appropriate as both questions and answers can be voted up or down, therefore useful information goes to the top, while not relevant or incorrect information goes to the bottom [Ovea]. Even though the site has moderators, the community voting system is the most important mechanism for how the website runs. To make sure that the voting system is not abused, the website allows users to earn their reputation and that being the case increase their privileges on the site (right to vote, comment on posts or edit other's posts). A user's reputation is measured by reputa-

tion points and badges. You gain a reputation by persuading your peers that you are an expert in your field. Reputation is an indicator of how much the community trusts you. There are multiple badges users can receive in different categories (questions, answers, participation, moderation, etc) [Oveb]. Example of several badges:

1. Student – first question with score of 1 or more;
2. Civic duty – vote 300 or more times;
3. Critic – first vote down;
4. Supporter – first vote up;
5. Proofreader – approve or reject 100 suggested edits.

Gamification and reward systems of non-game problems are known to increase creativity, job involvement, quality of communication, and overall productivity [KV15]. It can stimulate the wish to participate in various activities and improve your knowledge to be better. This can be beneficial in our review system as well because it is important to keep participants engaged in the process and by that, they could bring value both to the community and themselves. However, it is important to keep in mind people's feelings as well because it is in human nature not to like criticism which could also lead to increased levels of conflict. Having these existing systems in mind, our goal is to gather a community of crystallography professionals from all over the world who will use the system as a primary source to upload their data. To calculate the trust level of the reviewer respected weight average was chosen as it allows to have more conditions to be included. It is calculated based on how many decisions were made (approved/rejected structure). There is a problem, that at the very first stage of the system, all reviewers will have the same trust level, regardless of their experience and knowledge. To allow differentiation 'initial_trust_level' variable will be introduced. This will enable primary owners of the system to grant higher trust levels for scientists and therefore make use of the respected weighted average to calculate the level of respect for each person providing judgments. Nonetheless, this parameter should not be manipulated and abused as it could instigate distrust in ratings.

4.8. Combining automated tools decisions with reviewers decisions

The sections above describe how automated checks and reviews are going to work. However, to ensure data quality control I combined both of the methods. Syntax checks are a focus primarily for the structure's author to make sure that the data is ready to be reviewed, is prepared according to the standards, and is of high quality. Failing to fix these errors will prevent further steps, therefore, the reviewer will not waste time running the checks manually, informing the author about needed corrections. Providing fully review-ready crystal structure information will be the responsibility of the author. Validation according to dictionaries on the other hand will be developed to be of interest both to the author and reviewer. This step runs after syntax validation is passed and provides a comprehensive explanation for the reviewer. If this validation fails, the structure needs a

more thorough review, meaning, that the whole workflow will be slightly modified (first address dictionary validation issues, resolve them or mark them irrelevant, and then proceed to scientific validation). Based on automatic tools output and scientific knowledge the reviewer will determine if the structure can be deposited. Sufficient time for reviews is needed, however, it is needed to be mindful of time constraints. This will be solved by allowing the machine to make its own decision. If the decision is not made after X time, the system will check the scale options provided in settings to determine whether the structure can be approved using background service which was described in the 4.3.4 section.

Example of possible ends for single structure review:

1. Syntax check failed – hard fail. Errors output is provided. Several errors are shown. The structure's status was updated to failed. No further steps were executed.

2. Syntax check exited with warnings/notes or passed – messages output is given. Validation according to dictionaries executed.

3. Validation according to dictionaries failed – hard fail. Errors output is provided. Several errors are shown. No further steps were executed.

4. Validation according to dictionaries passed with warnings – messages output is given. A review is needed. However, if the review is not given for X amount of time, the author can submit his/her work if in system settings it is set to allow such action. If yes, the structure will be saved to COD with all checks output.

5. Scientist's review failed – if scientist rejects structure it is not uploaded to COD if submit anyway option is not allowed. The structure's status was updated to failed.

6. Scientist's review failed – if scientist rejects structure, the author may still upload it to COD if submit anyway option is allowed. In that case, checks output and comments are uploaded together. The structure's status is updated to PushedWithoutApproval.

7. Scientist's review passed – structure uploaded to COD. The structure's status is updated to Success.

A graphical example representation of the system's decision tree:

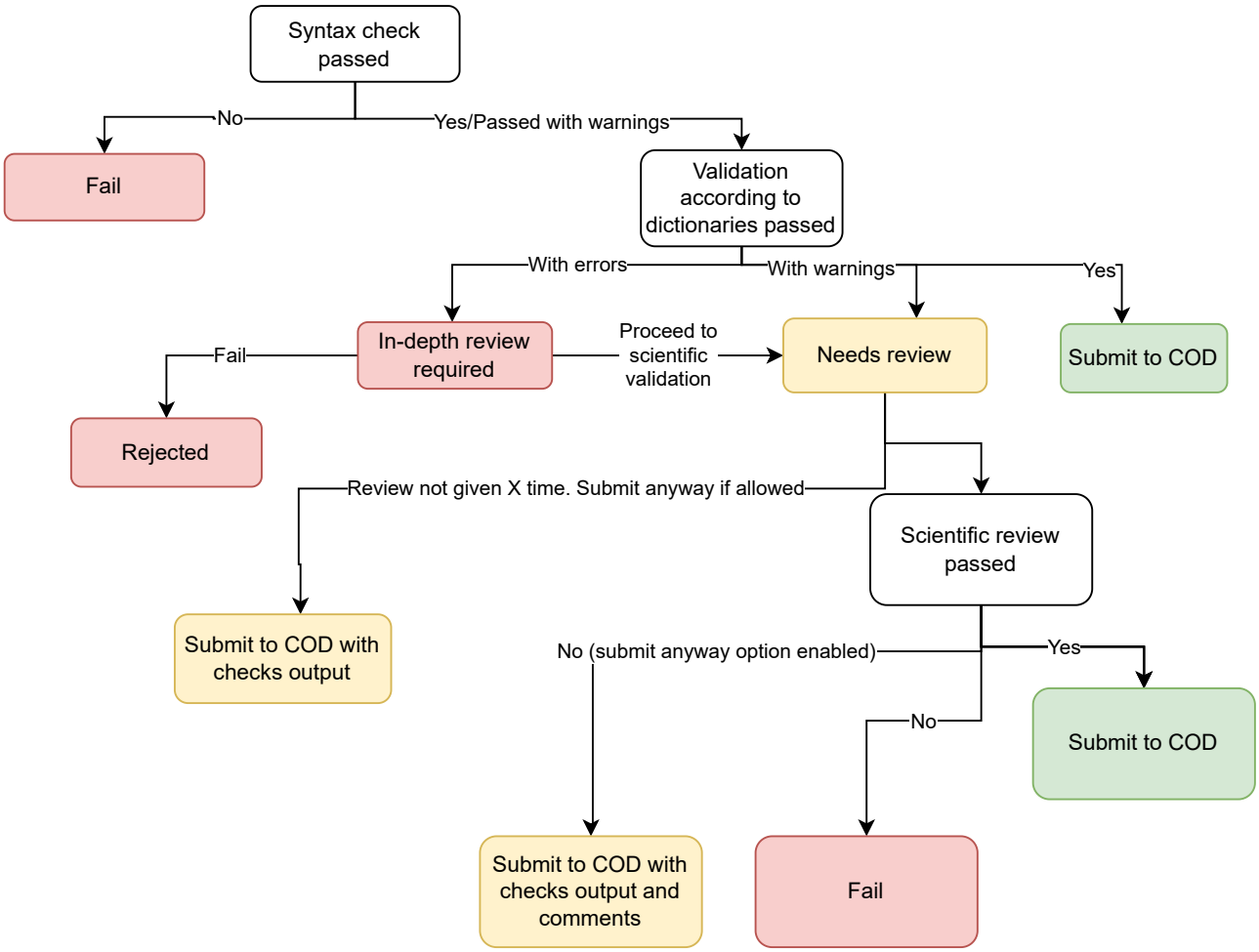

Figure 8. System decision tree

4.9. Information models

One of the problems that come with the creation of the system is that it has a lot of information:

1. General components of the system (e.g. menu, buttons, footer);
2. settings of scales;
3. author information;
4. data of structure;
5. validation tools;
6. output of validation tools;
7. comments and discussions between reviewers and author.

It's important to consider the needs of users when deciding how to display automated validation results and reviewers' comments and discussions. We want to provide enough information to be helpful, but not so much that it becomes overwhelming or confusing. To organize components for the system I used several approaches:

1. Group components by functionality – this approach organizes components based on the task they are used for. For example, there are a set of components for inputting data (uploading files, comment section), displaying data (e.g. structure information), and another set for visualizing and analyzing data (e.g. calculating and showing validation tools' outputs).

2. Organize components by role – this approach will help to differentiate between reviewers' components and the author's components.

3. Use descriptive names – helps reviewers and authors understand the purpose of components and how to use them.

4. Use descriptive colors – helps to understand the purpose of elements and can give them meaning. Several key rules will be used when choosing colors: they need to be consistent through all systems and have the same meaning (e.g. red signifies error, while green shows success cases). Colors should be complementary, meaning that they're on the opposite side of the color wheel to give a balanced look, which also helps with another rule – to focus on as many users as possible and introduce enough sufficient contrast for people with vision deficiency.

Automation validation part:

1. Clear, concise messaging – it is very important for the author to understand the results of the automated validation. Therefore, a summary of the check run will be provided stating the check exit code and several errors/warnings/notes.

2. Detailed reports – show the results of the automated validation in more depth, including detailed information on what tests were run together with specific issues that were detected.

3. Visual indicators – color coding that quickly shows the user the status of validation results. E.g. green checkmark when validation passed and a red X sign when validation failed.

This is an example view of automation validation checks result:

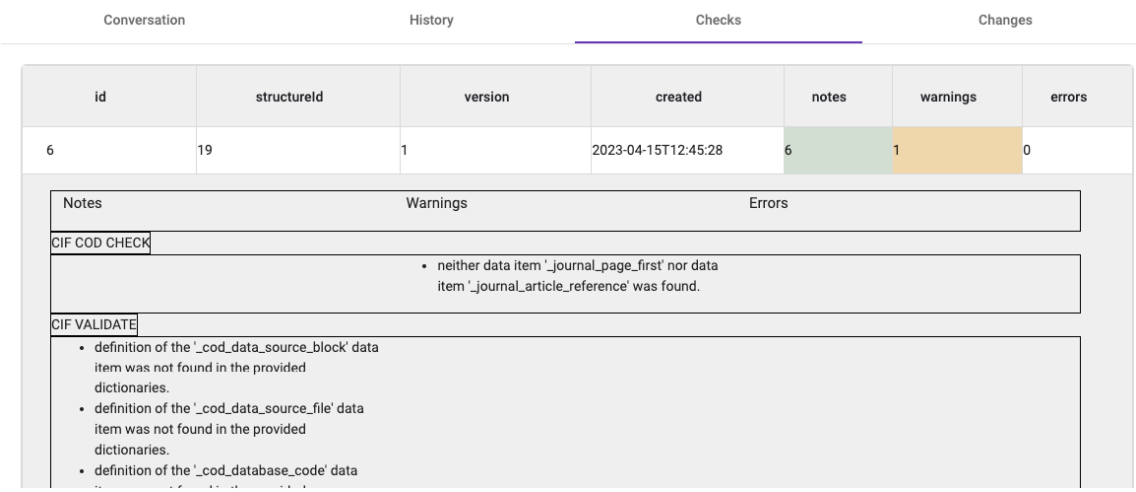

Figure 9. Automation validation view

Reviewers part:

1. Use a threaded commenting system – allows reviewers to reply to specific comments, whereas creates relevant conversation on a single topic and helps to track it.

2. Use color-coding or labels to distinguish different reviewers – helps identify people and comments made by different individuals.

3. Use email notifications – alerts reviewers when the discussion is updated and ensures that reviewers don't miss important comments.

These approaches will help to organize and display components effectively and consistently. Having automated validation results in a clear and concise style, and allowing users to drill down into the details, will help them understand the results. Using a threaded commenting system, visual indicators, color-coding or labels to distinguish different reviewers, and email notifications can help reviewers stay up-to-date.

The structure part consists of two parts:

1. Details – shows general information about the structure. Uses database table `structure_display` to determine what values should be shown. They can be dynamically added as needed to the database. Then typescript code checks which display names are present both in the table and structure content and parses according to display_value regex that is provided in the table. The code snippet below illustrates the implementation:

```
// Prepares data to be printed in a human-readable way.
// Display names can be added in the structure_display table
formatData(data: string){
  this.dataToPrint = data.split('\n').map((row) => {
    const [key, value] = row.trim().split(/\s+/);
    if (!(key in this.alias)) {
      return null;
    }
    const [displayName, format] = this.alias[key];
    const formattedValue = format.replace('%s', value);
    return { key, displayName, value: formatted value };
  }).filter((row) => row !== null);
  return this.dataToPrint.filter((row) => row.value != 'undefined');
}
```

`alias: Record<string, [string, string]>;` – defines an object that has keys of type string and values that are tuples containing two strings. It stores the data from `structure_display` table.

The result of parsed data is shown in Figure 10:

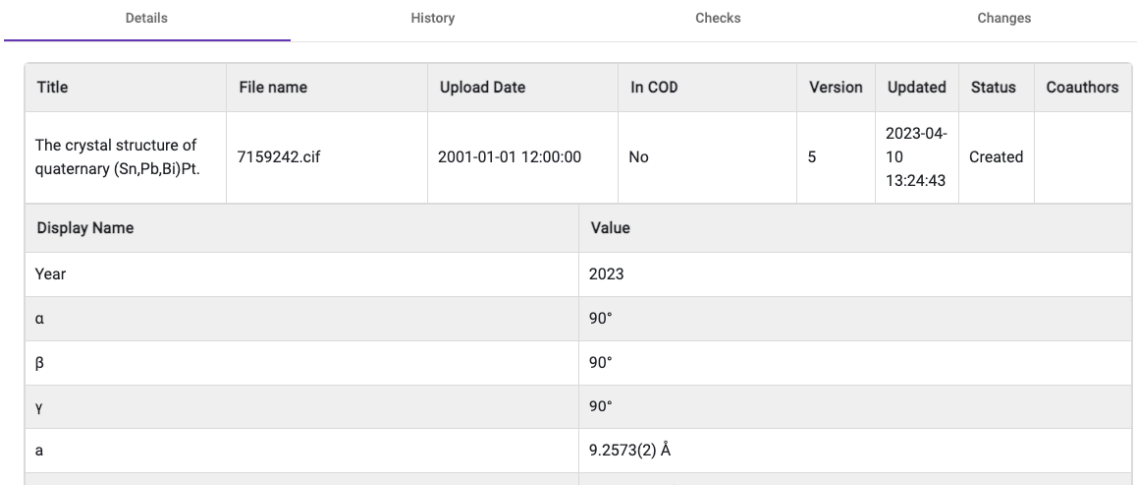

Figure 10. Structure details view

2. Changes – shows the content of the CIF file and illustrates added/removed changes compared to the last version if such exists. The example is shown in Figure 11.

Figure 11. Structure content view

4.10. Integration with COD infrastructure

As mentioned in previous sections, the new system is compatible with existing COD infrastructure as is developed to be cross-platform. Currently, cod-tools and COD servers run on Debian operating systems. The new system communicates with existing software when the structure is approved either by a reviewer or automatically or the user decides to deposit his/her structure anyway despite the absence of approvals. There are three ways CIF data can be deposited from the candidate's database to COD:

1. Using cif_cod_deposit – a plugin provided by cod-tools:

```
cif_cod_deposit 7712656.cif username=depositor password=password
```

2. Using curl or HTTP request:

```
curl -X POST http://test.crystallography.net/cod/cgi-bin/cif-deposit.pl
-H "Content-Type: multipart/form-data"
-F "cif_file=7712656.cif"
-F "username=depositor_name" -F "password=depositor_password"
-F "deposition_type=published|prepublication|personal"
-F "author_name=author_name"
-F "author_email=author_email"
```

As the new system provides Web API the option with HTTP request was chosen. Once the request is initiated, the structure validity is checked once again and if it passes all additional checks, it is uploaded to the COD database. The request returns a response in HTML or text format and is assigned to cod_upload_result. If the upload was successful the data of the candidate structure will be updated: auto-generated cod_id will be assigned, and upload_to_cod_date set to the current timestamp. The general flow is shown in Figure 12.

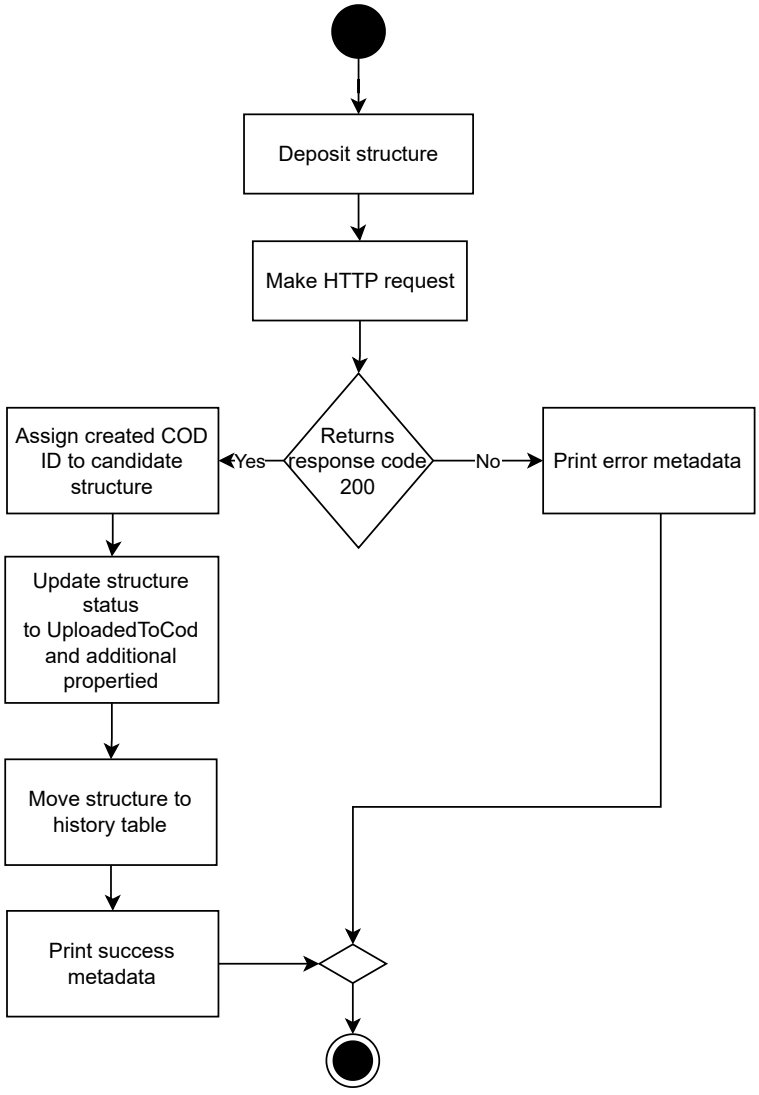

Figure 12. Deposit structure flow

4.11. Architecture

The choice of architecture plays an important part in building modern web applications. In my case, I have chosen a building application using .NET6 and Angular. While NET7 is already available, .NET is the latest version of Microsoft's flagship framework. It has a powerful set of tools and libraries for building scalable applications that can run on multiple platforms (Windows, Linux, macOS). Scalability together with performance and efficiency were some of the factors I considered when choosing technology for building Web API and background service. The ability to build scalable, rich, and interactive user interfaces (as well as experience in current technology) was the reason for choosing the Angular framework and TypeScript. One of the key benefits of using Angular, from my point of view, is being able to build complex applications with modular and clean architecture which reduces complexity and improves maintainability of the code. Furthermore, the architecture of Angular applications is relatively similar to the structure of .NET applications, and both Typescript and `C#` can be treated as object-oriented languages. For Web API, I chose the repository pattern which is a frequently used software design pattern, especially in data-driven applications. One of the most important benefits of this pattern is that encourages separation between the data access layer and the rest of the application. Having different layers also makes the system more testable as different parts can be easily mocked to simulate different scenarios.

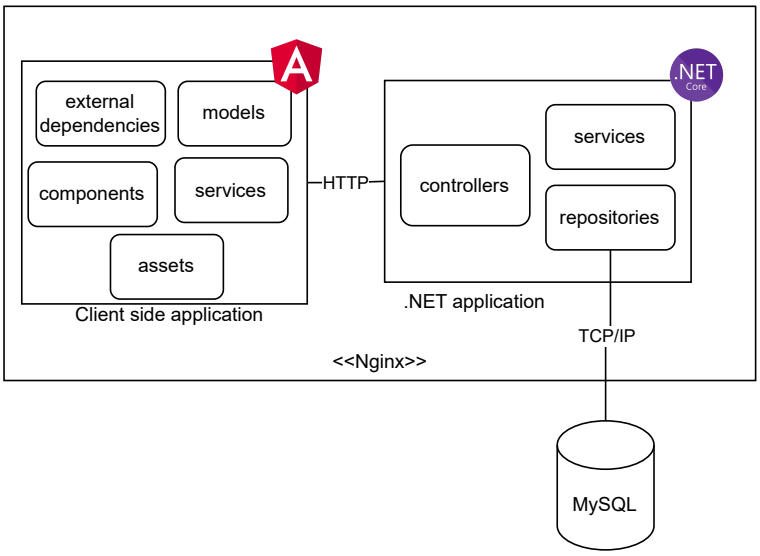

Figure 13. System architecture

The system's components are placed by their responsibility, e.g. all structure-related logic is described in `StructuresService`.

Web API's methods are documented using Swagger – an open-source tool that is used to document RESTful APIs as seen in the figure below:

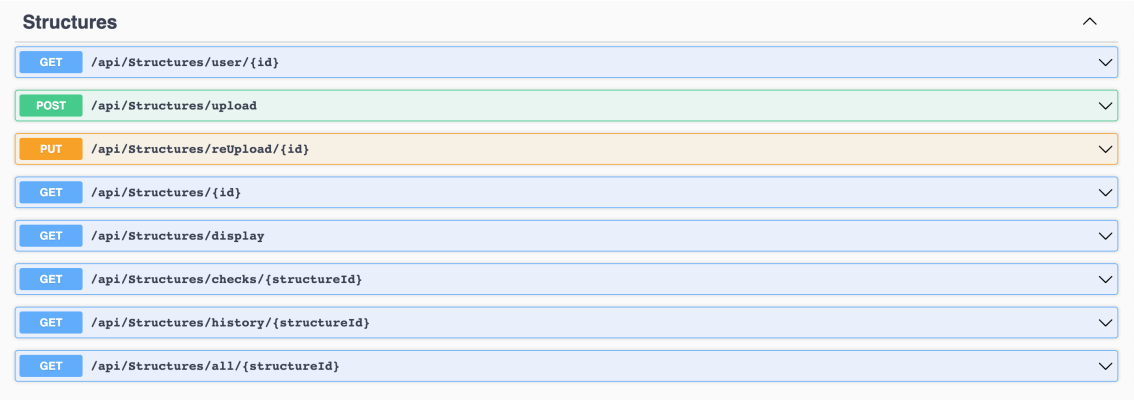

Figure 14. Example of Swagger usage

The application is deployed to the Debian server using Nginx. For the Web API part (.NET application), this is done using the dotnet publish command, which creates a self-contained deployment package that can be run on any system with the appropriate dependencies installed. For the Angular application, `ng build` command is used, which creates a bundle of static assets which are then served by a web server.

4.12. Use case example

To illustrate a real-world example a recent use case can be discussed. Recently Japanese scientists deposited a structure into the COD and encountered a problem that our system could've helped solve. Some of the files did not pass the automated checks, and the scientists reached out to the COD community via email. As an expert, they consulted with colleagues and had discussions with the Japanese team before accepting the structure. Unfortunately, ad-hoc workarounds were needed to find structure, test it and identify the problem as well as the data had to be reconstructed manually, which took three days. With our system, this could have been done in half-time or even less as the structure would have been pre-checked and all discussions would've happened in a single place.

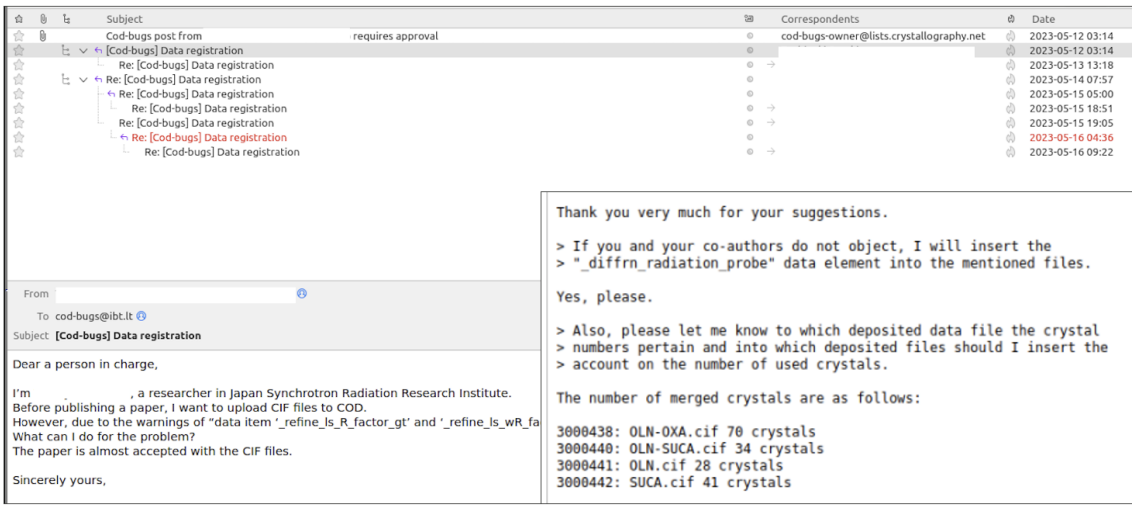

Figure 15. Use case example

4.13. Future improvements

The biggest aim is that the system will help to accelerate the review process and new structure deposition in the COD while also improving the quality of them. However, the limitation of fraud and “honest mistakes” passing through still remains. While there are two checks implemented in the system as of now, there is room for further enhancements and additional checks in the future to lessen limitations effectively. UI and feature improvements heavily rely on the feedback and experience of the users and would need to be refined together.

5. Results and conclusions

5.1. Results

1. Main problems of crystallographic data publication were identified: the most common problems are syntax and semantic issues, which can be easily fixed by automated tools in most cases. Furthermore, there's also a risk of "honest mistakes", inaccuracies that result in poor data quality. Rare, however, a more damaging problem in crystallography is frauds. They interfere with scientific inventions, are hard to detect, time-consuming and expensive. Automated tools usually cannot spot fraud and human intervention is needed.

2. Different levels of crystal structures' formal checks were reviewed. Automated checks include syntax validation, validation according to dictionaries, and checking according to additional subject area criteria. The first check should be syntax validation which makes sure that the file does not have basic errors such as misspelled keywords. The second step is to validate according to dictionaries to verify that data is consistent with known crystallographic data. Lastly comes additional subject area criteria validation checked against specific context. These checks should come in the right order to optimize the review process for the reviewers as the structures should reach scientists as late as possible so that scientists wouldn't need to check the information that could be detected by automated tools and could focus their time on making crucial decisions.

3. Tools that will be used for automatic validation were defined as well as different types of decision–making scales were introduced. The severity–based scale allows results to be manipulated on their error/warning/note statuses. Selection–based scale defines how the validation check should fail or pass and the selection–based scale provides comprehensive information on validation according to dictionaries and allows experts to mark each result as relevant or not important, thus allowing them to focus on a smaller subset of data.

4. Since there's a need for a decision to be made either way, but there can be cases when neither author nor reviewer came to an agreement or the data review was not finished, general settings were introduced to make a calculated judgment whether the structure should be moved to COD or rejected. The system uses the settings as well as output from validation tools to run a background service that checks and updates the status as needed.

5. System that combines automated tools with human reviewers' insights has been developed, deployed to the test server, and integrated with existing infrastructure. To ensure its accuracy and reliability, it was tested with unit tests and CIFs that currently exist in COD. While the system has not yet been field-tested, it has promising potential to help effectively review and ensure the quality of structures in the real world. Further refinement and testing of this system could help scientists to save a lot of time and financial resources that could be dedicated to other scientific tasks that lead to advances in various domains, including science, engineering, and business.

5.2. Conclusions

1. Universal system which would merge inputs from experts, reviewers, and machines was not found. There are several programs (EnCIFer, checkCIF, COD::CIF::Parser, etc) that can be used for automatic validation and there are separate tools for reviews.

2. Peer review is an important process in order to publish crystal data and plays a huge role in making that publication trustworthy, especially in crystallography. Even though it is a crucial aspect in science fields, there's a significant lack of peer review of pure data. This is a bothering issue for many scientists as not having thorough checks and reviews of data can result in errors and misconceptions that can have huge consequences starting from a time-consuming fraudulent data retraction process that is also a financial burden for involved parties (scientists and institutions) to waste time and resources. This can also lead to a loss of confidence in the scientific community if the data is not presented as accurate and reliable. For that reason, it is critical to emphasize the pure data peer–review process to ensure data and publications credibility so that scientists would make informed decisions on accurate data which is crucial for moving forward with scientific knowledge and solving problems.

3. Purely automated systems are not sufficient to ensure the validity of data and varying levels of human/machine participation should be investigated and applied. Having a unified system would be beneficial for the crystallographers' community as it would speed up the process of getting structures and articles published and would improve the overall quality of crystal structures data in the databases. For that decision support system approach was used as it helps to maintain logical and rational control of the human-computer process.

4. Based on the knowledge I gathered on crowd wisdom, it is evident that collective intelligence and its concepts like diversity and independence can increase the accuracy of decision-making. Scientists should not be influenced by the decisions and opinions of their peers. These ideas were taken into account while creating the system by allowing scientists to make independent decisions and aggregate results.

References

[Agu13] Nadales Agut. Syntax and semantics of the compositional interchange format for hybrid systems. *The Journal of Logic and Algebraic Programming*, 82(1):1–52, 2013-01. DOI: 10.1016/j.jlap.2012.07.001.

[AJS+04] Frank H. Allen, Owen Johnson, Gregory P. Shields, Barry R. Smith, and Matthew Towler. CIF applications. enCIFer a program for viewing, editing and visualizing CIFs. *Journal of Applied Crystallography*, 37(2):335–338, 2004-03. DOI: 10.1107/s0021889804003528.

[AMR+13] Muhammad Asaduzzaman, Ahmed Shah Mashiyat, Chanchal K. Roy, and Kevin A. Schneider. Answering questions about unanswered questions of Stack Overflow. In *2013 10th Working Conference on Mining Software Repositories (MSR)*. IEEE, 2013-05. DOI: 10.1109/msr.2013.6624015.

[BB13] Alberto Bacchelli and Christian Bird. Expectations, outcomes, and challenges of modern code review. In *Proceedings of the 2013 International Conference on Software Engineering*, ICSE '13, pp. 712–721, San Francisco, CA, USA. IEEE Press, 2013. ISBN: 9781467330763. DOI: 10.5555/2486788.2486882.

[BBB+16] Herbert J. Bernstein, John C. Bollinger, I. David Brown, Saulius Gražulis, James R. Hester, Brian McMahon, Nick Spadaccini, John D. Westbrook, and Simon P. Westrip. Specification of the Crystallographic Information File format, version 2.0. *Journal of Applied Crystallography*, 49(1):277–284, 2016-02. DOI: 10.1107/s1600576715021871.

[BCB+17] Amiangshu Bosu, Jeffrey C. Carver, Christian Bird, Jonathan Orbeck, and Christopher Chockley. Process aspects and social dynamics of contemporary code review: insights from open source development and industrial practice at microsoft. *IEEE Transactions on Software Engineering*, 43(1):56–75, 2017-01. DOI: 10.1109/tse.2016.2576451.

[Ber07] Helen M. Berman. The Protein Data Bank: a historical perspective. *Acta Crystallographica Section A Foundations of Crystallography*, 64(1):88–95, 2007-12. DOI: 10.1107/s0108767307035623.

[BGH+17] Ian Bruno, Saulius Gražulis, John R Helliwell, Soorya N Kabekkodu, Brian McMahon, and John Westbrook. Crystallography and databases. *Data Science Journal*, 16:38, 2017-08. DOI: 10.5334/dsj-2017-038.

[Bim22] David Bimler. Better Living through Coordination Chemistry: A descriptive study of a prolific papermill that combines crystallography and medicine, 2022-04. DOI: 10.21203/rs.3.rs-1537438/v1.

[Bro96] I. D. Brown. CIF (Crystallographic Information File): A Standard for Crystallographic Data Interchange. *Journal of research of the National Institute of Standards and Technology*, 101:341–346, 3, 1996. ISSN: 1044-677X. DOI: 10.6028/jres.101.035. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4894604/.ppublish.

[BS96] D. Beevis and H. Schuffel. *Improving function allocation for integrated systems design*. 1996. URL: https://apps.dtic.mil/sti/pdfs/ADA491093.pdf.

[Cha03] Geoffrey Chang. RETRACTED: Structure of MsbA from Vibrio cholera: A Multidrug Resistance ABC Transporter Homolog in a Closed Conformation. *Journal of Molecular Biology*, 330(2):419–430, 2003-07. DOI: 10.1016/s0022-2836(03)00587-4.

[COD] COD. URL: http://www.crystallography.net/cod/.

[Cry21] Crystallography Open Database. Citing & referencing: Harvard style, 2021. URL: https://www.imperial.ac.uk/media/imperial-college/administration-and-support-services/library/public/harvard.pdf.

[CZL+11] Heng-heng Cao, Hong-guo Zhang, Ding-gui Luo, and Yong-heng Chen. Notice of Retraction: Effect of COD/Sulfate Ratios on Batch Anaerobic Digestion Using Sulfate-Reduction Bacteria. In *2011 5th International Conference on Bioinformatics and Biomedical Engineering*. IEEE, 2011-05. DOI: 10.1109/icbbe.2011.5781159. URL: https://ieeexplore.ieee.org/document/5781159.

[ECN+21] Felipe Ebert, Fernando Castor, Nicole Novielli, and Alexander Serebrenik. An exploratory study on confusion in code reviews. *Empirical Software Engineering*, 26(1), 2021-01. DOI: 10.1007/s10664-020-09909-5.

[GBB+11] A. Gaulton, L. J. Bellis, A. P. Bento, J. Chambers, et al. ChEMBL: a large-scale bioactivity database for drug discovery. *Nucleic Acids Research*, 40(D1):D1100–D1107, 2011-09. DOI: 10.1093/nar/gkr777.

[GCD+09] Saulius Gražulis, Daniel Chateigner, Robert T. Downs, A. F. T. Yokochi, et al. Crystallography Open Database – an open-access collection of crystal structures. *Journal of Applied Crystallography*, 42(4):726–729, 2009-05. DOI: https://doi.org/10.1107/S0021889809016690.

[GDM+11] Saulius Gražulis, Adriana Daškevič, Andrius Merkys, Daniel Chateigner, et al. Crystallography open database (COD): an open-access collection of crystal structures and platform for world-wide collaboration. *Nucleic Acids Research*, 40(D1):D420–D427, 2011-11. DOI: 10.1093/nar/gkr900.

[GMR+01] Georgios V. Gkoutos, Peter Murray-Rust, Henry S. Rzepa, and Michael Wright. Chemical markup, xml, and the world-wide web. 3. toward a signed semantic chemical web of trust. *Journal of Chemical Information and Computer Sciences*, 41:1124–1130, 2001. DOI: 10.1021/ci000406v. eprint: http://pubs.acs.org/doi/pdf/10.1021/ci000406v. URL: http://pubs.acs.org/doi/abs/10.1021/ci000406v.

[Har96] Richard L. Harlow. Troublesome Crystal Structures: Prevention, Detection, and Resolution. *Journal of research of the National Institute of Standards and Technology*, 101:327–339, 3, 1996. ISSN: 1044-677X. DOI: 10.6028/jres.101.034. ppublish.

[IUC] IUCr. URL: https://journals.iucr.org/j/services/peerreview.html.

[JDA+11] Randy J. Read, Paul D. Adams, W. Bryan Arendall, and Axel T. Brunger. A New Generation of Crystallographic Validation Tools for the Protein Data Bank. *Structure*, 19(10):1395–1412, 2011-10. DOI: 10.1016/j.str.2011.08.006.

[JMR+17] Clifford D. Johnson, Michael E. Miller, Christina F. Rusnock, and David R. Jacques. A framework for understanding automation in terms of levels of human control abstraction. In *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, 2017-10. DOI: 10.1109/smc.2017.8122766.

[KBG+15] Oleksii Kononenko, Olga Baysal, Latifa Guerrouj, Yaxin Cao, and Michael W. Godfrey. Investigating code review quality: do people and participation matter? In *2015 IEEE International Conference on Software Maintenance and Evolution (ICSME)*. IEEE, 2015-09. DOI: 10.1109/icsm.2015.7332457.

[KBG16] Oleksii Kononenko, Olga Baysal, and Michael W. Godfrey. Code review quality. In *Proceedings of the 38th International Conference on Software Engineering*. ACM, 2016-05. DOI: 10.1145/2884781.2884840.

[KSA14] Jacalyn Kelly, Tara Sadeghieh, and Khosrow Adeli. Peer Review in Scientific Publications: Benefits, Critiques, & A Survival Guide. *EJIFCC*, 25:227–243, 3, 2014-10. ISSN: 1650-3414. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4975196/pdf/ejifcc-25-227.pdf. epublish.

[KV15] Kamasheva and Valeev. Usage of Gamification Theory for Increase Motivation of Employees. *Mediterranean Journal of Social Sciences*, 2015-02. DOI: 10.5901/mjss.2015.v6n1s3p77.

[KWT72] Olga Kennard, D. G. Watson, and W. G. Town. Cambridge Crystallographic Data Centre. I. Bibliographic File. *Journal of Chemical Documentation*, 12(1):14–19, 1972-02. DOI: 10.1021/c160044a006.

[LLC+17] Jiaqi Lu, Shijun Liu, Lizhen Cui, Li Pan, and Lei Wu. Crowd wisdom drives intelligent manufacturing. *International Journal of Crowd Science*, 1(1):39–47, 2017-03. DOI: 10.1108/ijcs-01-2017-0002.

[LO03] Xiang-Jun Lu and Wilma K. Olson. 3dna: a software package for the analysis, rebuilding and visualization of three-dimensional nucleic acid structures. *Nucleic acids research*, 31:5108–21, 2003.

[LP13] Aidan Lyon and Eric Pacuit. The Wisdom of Crowds: Methods of Human Judgement Aggregation. In *Handbook of Human Computation*, pp. 599–614. Springer New York, 2013. DOI: 10.1007/978-1-4614-8806-4_47.

[MGB+18] David Mendez, Anna Gaulton, A Patrícia Bento, Jon Chambers, et al. ChEMBL: towards direct deposition of bioassay data. *Nucleic Acids Research*, 47(D1):D930–D940, 2018-11. DOI: 10.1093/nar/gky1075.

[MKA+14] Shane McIntosh, Yasutaka Kamei, Bram Adams, and Ahmed E. Hassan. The impact of code review coverage and code review participation on software quality: a case study of the qt, VTK, and ITK projects. In *Proceedings of the 11th Working Conference on Mining Software Repositories - MSR 2014*. ACM Press, 2014. DOI: 10.1145/2597073.2597076.

[Mur17] Andreas Murr. *"wisdom of crowds"*. In 2017-04, pp. 835–860.

[MVB+16] Andrius Merkys, Antanas Vaitkus, Justas Butkus, Mykolas Okulič-Kazarinas, Visvaldas Kairys, and Saulius Gražulis. COD::CIF::parser: an error-correcting CIF parser for the Perl language. *Journal of Applied Crystallography*, 49(1):292–301, 2016-02. DOI: 10.1107/s1600576715022396.

[Ovea] Stack Overflow. URL: https://stackoverflow.com/help/privileges/vote-up.

[Oveb] Stack Overflow. URL: https://stackoverflow.com/help/badges.

[PKT+12] Weerapong Phadungsukanan, Markus Kraft, Joe Townsend, and Peter Murray-Rust. The semantics of Chemical Markup Language (CML) for computational chemistry : CompChem. *Journal of Cheminformatics*, 4(1), 2012-08. DOI: 10.1186/1758-2946-4-15.

[RGC+14] Peter C. Rigby, Daniel M. German, Laura Cowen, and Margaret-Anne Storey. Peer review on open-source software projects. *ACM Transactions on Software Engineering and Methodology*, 23(4):1–33, 2014-09. DOI: 10.1145/2594458.

[Smi06] R. Smith. Peer review: a flawed process at the heart of science and journals. *Journal of the Royal Society of Medicine*, 99(4):178–182, 2006-04. DOI: 10.1258/jrsm.99.4.178.

[Spe03] A. L. Spek. Single-crystal structure validation with the program PLATON. *Journal of Applied Crystallography*, 36(1):7–13, 2003-01. DOI: 10.1107/s0021889802022112.

[Spe20] Anthony L. Spek. Validation ALERTS: what they mean and how to respond. *Acta Crystallographica Section E Crystallographic Communications*, 76(1):1–11, 2020-01. DOI: 10.1107/s2056989019016244.

[TDH14] Jason Tsay, Laura Dabbish, and James Herbsleb. Let's talk about it: evaluating contributions through discussion in GitHub. In *Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering*. ACM, 2014-11. DOI: 10.1145/2635868.2635882.

[TFA88] Eefraim Turban, Janet Carenon FIsher, and Steve Altman. DECISION SUPPORT SYSTEMS IN ACADEMIC ADMINISTRATION. *Journal of Educational Administration*, 26(1):97–113, 1988-01. DOI: 10.1108/eb009943.

[VMG21] Antanas Vaitkus, Andrius Merkys, and Saulius Gražulis. Validation of the Crystallography Open Database using the Crystallographic Information Framework. *Journal of Applied Crystallography*, 54(2):661–672, 2021-02. DOI: 10.1107/s1600576720016532. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8056762/.

[Wan03] Sun-Chong Wang. Artificial neural network. In *Interdisciplinary Computing in Java Programming*, pp. 81–100. Springer US, 2003. DOI: 10.1007/978-1-4615-0377-4_5.

[WDF00] Peter Wright, Andy Dearden, and Bob Fields. Function allocation: a perspective from studies of work practice. *International Journal of Human-Computer Studies*, 52(2):335–355, 2000-02. DOI: 10.1006/ijhc.1999.0292.

[WJ04] Andrew Whitby and Audun Jøsang. Filtering out unfair ratings in bayesian reputation systems. *The Icfain Journal of Management Research*, 4, 2004-01.

[WM19] Lingfei Wang and Tom Michoel. Accurate wisdom of the crowd from unsupervised dimension reduction. *Royal Society Open Science*, 6(7):181806, 2019-07. DOI: 10.1098/rsos.181806.

[WMD+08] Alexander Wlodawer, Wladek Minor, Zbigniew Dauter, and Mariusz Jaskolski. Protein crystallography for non-crystallographers, or how to get the best (but not more) from published macromolecular structures. *The FEBS journal*, 275:1–21, 1, 2008-01. ISSN: 1742-464X. DOI: 10.1111/j.1742-4658.2007.06178.x. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4465431/pdf/nihms690751.pdf. ppublish.

[ZDG+16] Christine Zardecki, Shuchismita Dutta, David S. Goodsell, Maria Voigt, and Stephen K. Burley. RCSB Protein Data Bank: A Resource for Chemical, Biochemical, and Structural Explorations of Large and Small Biomolecules. *Journal of Chemical Education*, 93(3):569–575, 2016-01. DOI: 10.1021/acs.jchemed.5b00404.

[ZHZ+13] Heping Zheng, Jing Hou, Matthew Zimmerman, Alexander Wlodawer, and Wladek Minor. The future of crystallography in drug discovery. *Expert Opinion on Drug Discovery*, 9(2):125–137, 2013-12. DOI: 10.1517/17460441.2014.872623.

Terms and abbreviations

1. Crystallography Open Database (COD) – Open-access collection of crystal structures of organic, inorganic, metal-organics compounds and minerals, excluding biopolymers.

2. Crystallographic Information File (CIF) – The International Union of Crystallography (IUCr) – is an organization devoted to the international promotion and coordination of the science of crystallography. The IUCr is a member of the International Council for Science (ICSU).

3. The International Union of Crystallography (IUCr) – an organization devoted to the international promotion and coordination of the science of crystallography.

4. The Cambridge Crystallographic Data Centre (CCDC) – a non-profit organization, which primary activity is the compilation and maintenance of the Cambridge Structural Database, a database of small molecule crystal structures.

5. X-ray crystallography – is the experimental science determining the atomic and molecular structure of a crystal, in which the crystalline structure causes a beam of incident X-rays to diffract into many specific directions.