Automating DOI Registration with DataCite API

Giuditta Parolini, Falko Glöckler

Research Data Management Services, Museum für Naturkunde Berlin – Leibniz Institute for Evolution and Biodiversity Science

Automation is a main trend in all contexts in which data are a valuable asset. Sustained growth in the amount of available data, greater interest in extracting knowledge from data, and the significant increase in data regulations and policies are constantly shifting the balance from human-based to automated data management solutions. Research data management is also experiencing this automation trend. The tasks of research data management teams have been constantly expanding in recent years. More demanding funder policies regarding research data and growing needs on the side of researchers, who are dealing with scientific data in quantities never experienced before, are constantly increasing the workload of research data management teams. Automation becomes, therefore, a necessary choice to avoid limiting or reducing the services provided to users.

The paper will discuss an example of automation implemented to meet the significant increase in DOI (Digital Object Identifier) registration requests for data publications received at the Museum für Naturkunde Berlin (MfN). It will present the web application that the MfN research data management team has been building to automate the DOI registration process with the provider DataCite. Running in a Docker container, a Django web application calls the DataCite API and interoperates with MfN institutional databases to prompt the DOI creation and to collect the metadata that are required for DOI registration. The paper will reflect on the code development process and the design of the user interface. It will also examine the opportunities that the web application offers for the work of the MfN research data management team, such as increased processing speed for DOI requests and improved quality control in the data publication process.

1 Introduction

Automation is a main trend in all contexts in which data are a valuable asset (Hobart 2020). Sustained growth in the amount of available data, greater interest in extracting knowledge from data, and significant increase in data regulations and policies are constantly shifting the balance from human-based to automated data management solutions.

Publiziert in: Vincent Heuveline, Nina Bisheh und Philipp Kling (Hg.): E-Science-Tage 2023. Empower Your Research – Preserve Your Data. Heidelberg: heiBOOKS, 2023. DOI: https://doi.org/10.11588/heibooks.1288.c18065

Research data management is also experiencing the automation trend. Automated solutions for the creation and constant update of data management plans and rule-based management of research data pipelines are just two examples of this (Chard et al. 2017; Miksa, Oblasser, and Rauber 2021). The potential for automation is not limited to researchers' daily work. At the data publication stage, there is still room for increasing automation.

Persistent identifiers (PIDs) in the form of DOIs (Digital Object Identifiers) are now well-established tools to promote and enhance citation, sharing, and reuse of research results (texts, datasets, media, etc.; Liu 2021). Once assigned a DOI, a research output has a permanent handle and the DOI will always resolve properly. In addition, descriptive metadata are associated with the DOI and provide relevant information about the object to which the DOI has been assigned. Registration agencies, such as Crossref and DataCite, provide the technical infrastructure for DOI services and all data publishers need to collaborate with one of these agencies to successfully register DOIs for their publications.

As DOIs are both machine and human-readable, they are a relevant element to make research data FAIR (Findable, Accessible, Interoperable, Reusable). Therefore, more and more often libraries and data management services of universities and research institutions register DOIs for the organisation research outputs. In many cases the process is still carried out manually using the web interfaces made available by DOI registration agencies, but there is the necessity to rethink the balance between people, processes, and technologies to cope with increasing amounts of DOIs requests.

The paper will discuss the web application that the research data management team at the Museum für Naturkunde Berlin (MfN) has been building to automate the DOI registration process with the provider DataCite. Running in a Docker container, a Django web application calls the DataCite API and interoperates with MfN institutional databases to prompt the DOI creation and to collect the metadata that are required for DOI registration. The paper will begin with a brief description of the data publication work at MfN and the current DOI workflow, continue with a description of the software architecture and user experience of the web application developed for automating the DOI registration process, and conclude with a reflection on the role of automation in current research data management.

2 MfN data publications

At MfN the research data management team is responsible for the pipelines of the organisation's data publications. The data publication process established by the institution follows the FAIR principles (Wilkinson et al. 2016). DOIs are registered to enhance data findability, accessibility, and reuse. MfN staff who request a DOI, for instance, can make available a dataset alongside a published paper without relying on paid services offered by academic publishers. In addition, researchers do not need to concern themselves with data availability in the institutional repository, as this is managed on their behalf by the research data management team. Once registered, the DOI will permanently identify

the dataset that can be referenced with a stable URL, and cited and shared with other researchers.

MfN registers DOIs with the global non-profit organisation DataCite, one of the main registration agencies for research data.[1] MfN benefits from DataCite's services via a cooperation agreement with the public organisation Deutsche Zentralbibliothek für Medizin (ZB MED). ZB MED is an official DataCite member and represents a consortium of German institutions engaged in life sciences research. ZB MED assigns prefixes to its consortium members and MfN has been assigned the prefix 10.7479. The MfN DOI `https://doi.org/10.7479/2j13-v254` identifies, for instance, a set of audio recordings of birds singing behaviour. The dataset is part of MfN Animal Sound Archive (Tierstimmenarchiv), which is one of the oldest and most extensive collections of animal sound recordings existing worldwide. It was started at MfN in the 1950s and now includes about one hundred and twenty thousand records.[2]

Using DataCite, DOIs can be created in three possible states: draft, registered, findable. DOIs in draft state are not yet publicly available and can still be deleted even though the DOI is reserved, DOI in registered state cannot be deleted anymore, but their metadata are only available to subscribers of DataCite, findable DOIs are officially registered and their metadata are publicly available. As evident in Figure 1, the number of DOIs registered by MfN has considerably increased over the years and it has more than doubled in 2022 compared to the total registrations in the previous year. This is not really surprising given the overall trend. If DataCite assigned just over one hundred and fifty thousand DOIs in 2011, this number has grown to over four millions in 2020 and the growth trend continues uninterrupted as the most recent statistics show.[3]

DOIs popularity is definitely good news for a FAIR management of research data, but poses practical challenges to the institutions that internally manage their data publications and register DOIs for them. The manual process of registering a DOI is time-consuming. Data managers need to collect the metadata associated to the dataset from internal systems and then transfer these metadata to the external service provider to register the DOI. In addition, creating a landing page to which the DOI redirects imposes further steps. The landing page needs to display properly all the information related to the dataset and stored in the descriptive metadata associated with the DOI. At the same time, the landing page must give access to all the files (e.g., csv files, txt files, media files, etc.) associated to the dataset. This requires to connect all the information related to the DOI also within institutional databases. Therefore, the DOI workflow typically involves multiple IT systems, such as institutional databases for data and metadata and the user interface offered by the DOI provider. Data managers need to be familiar with all these IT systems and switch from one to the other to keep the metadata updated and to complete the registration process.

1 `https://datacite.org`

2 `https://www.tierstimmenarchiv.de/webinterface`

3 See DataCite 2020 Annual Report (`https://datacite.org/wp-content/uploads/2023/06/DataCite-2020-Annual-Report.pdf`) and DataCite current statistics (`https://stats.datacite.org`).

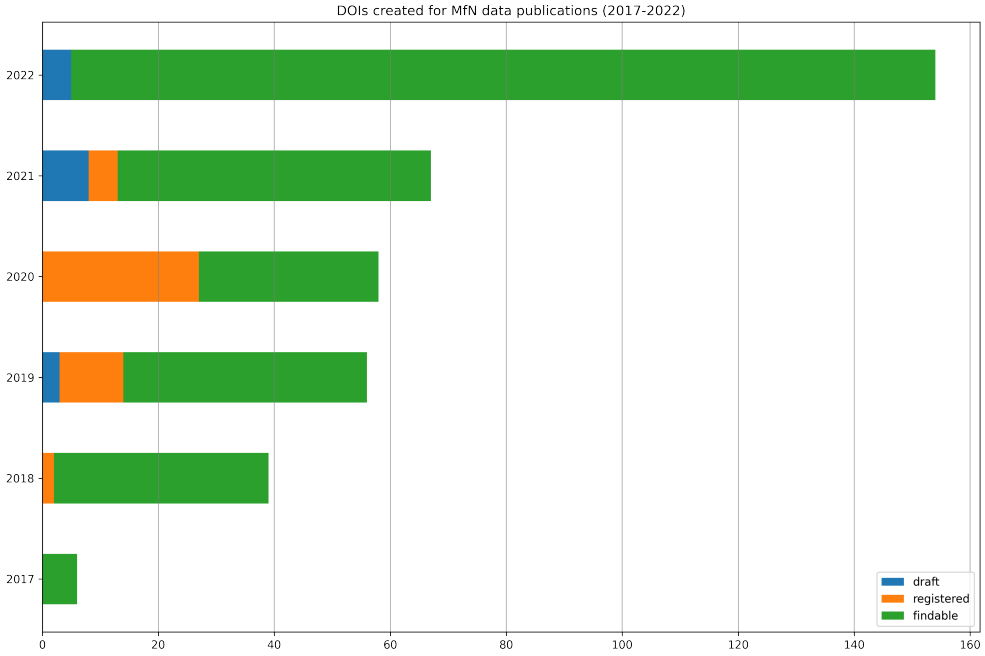

Figure 1: DOIs created for MfN data publications (2017-2022).

At MfN the DOI registration process begins with the creation of a DOI request by a member of the institution (Figure 2). The request is made by filling in a dedicated webform in the internal survey tool. The information collected includes metadata such as dataset author and abstract, license, necessity to set an embargo on the data, etc. The form is received by the data management team that checks manually the request and accepts it as it is or requires further information about the dataset for which the DOI is needed. If the request is accepted, the data manager creates a draft DOI using the web interface DataCite Fabrica.[4] The draft DOI is then added to the survey form originally submitted and the form data are transferred to MfN metadata storage, which is a MySQL database. The data transfer is also necessary to generate a landing page in the institutional repository where the dataset and metadata will be publicly available. To make the dataset available, the data files (e.g., csv files, media files, etc.) need to be uploaded to MfN digital asset management system, and each asset must be linked to the respective record of the MfN metadata database. Once this step is completed, the dataset will be properly displayed and available on the landing page created for the DOI. At this point, it will be possible to export the metadata in XML format from the landing page. These metadata will be used to add the descriptive metadata to the draft DOI record using DataCite Fabrica user interface. Once the metadata have been added, the status of the DOI can be changed to registered or findable by the data manager.

4 `https://doi.datacite.org`

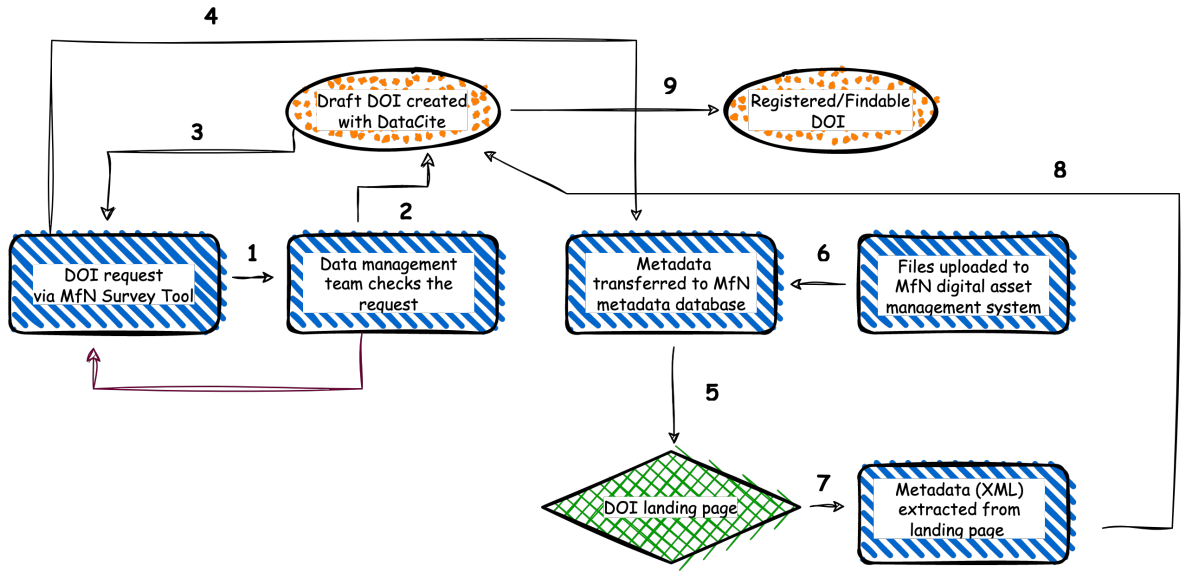

Figure 2: DOI workflow at MfN.

As evident from the description, the process of minting and registering a DOI with DataCite for MfN data publications has multiple steps and for the large part it is a human-centred process.[5] The significant increase in DOI requests registered in the past few years and the interest in eliminating possible human errors during DOI creation and metadata transfer impose to rethink the current procedure and encourage solutions that automate at least a few of the steps in the DOI creation process. The availability of a well-documented and reliable API for DataCite Fabrica suggested to start the automation process of the MfN DOI workflow from the interaction with the DataCite service. DOI creation and metadata registration and update are all tasks that can be easily carried out via the DataCite API rather than the DataCite user interface, as explained below in detail.

3 Automation opportunities offered by APIs

APIs (API is a shorthand for Application Programming Interface) are computer routines that allow communication between software components via requests and responses. Most APIs have methods that allow to perform basic operations on data. Standard operations are creating data (POST method), updating data (PUT method), retrieving data (GET method), and destroying data (DELETE method). Data are usually transferred to/from the API in JSON or XML format. APIs have a long history in computer science, but their popularity has greatly increased with the development of the World Wide Web – nowadays, most APIs are web APIs – because they allow easy interaction with remote resources using standardised protocols. Building and deploying APIs is now a key compo-

5 In the workflow described, only the metadata transfer to the metadata database (Step 4) and the creation of the landing page (Step 5) are triggered by human intervention, but otherwise automated.

nent of web development and a solution preferred by many organisations for transferring and receiving data.

DataCite API is a REST (REpresentational State Transfer) API (Masse 2011). Currently, there are two versions of the DataCite API, a Public API for non-authenticated users and a Member API for authenticated users.[6] Non-authenticated users can browse, query, and retrieve all findable DOIs and related metadata using the Public API. Authenticated users can also register DOIs, add and update DOI metadata, manage prefixes, etc. with the Member API. In short, DataCite Member API allows to automate all operations required to mint and register a DOI and to add and update the related metadata.[7] To facilitate members interested in working with the API rather than with the user interface, DataCite offers rich documentation and the opportunity to carry out development work with a test API. In the test API, all methods of the production API are available and users can experiment with all possible DOI statuses (draft, registered, findable) without interfering with the work of the production API and without cluttering the DOI register of the institution with dummy records. Both a test API and a test web interface are made available by DataCite for development purposes.[8] In addition, working with the DataCite API is facilitated by the availability of REST clients that provide all methods (e.g., POST, PUT, UPDATE, DELETE) to interact with the API. In the development work carried out at MfN, the Python API client wrapper for the DataCite API was used, as Python is the language of choice for many of the software applications currently developed by the institution.[9]

Relying on the DataCite API, at MfN it has been possible to automate all the interaction steps with the DOI registration service, starting with the creation of the DOI in draft state and proceeding with the transmission of the DOI metadata. The data manager can now request a draft DOI from within the institution's internal services and add the metadata available in the DOI request form to the draft DOI without needing to extract from the landing page the XML version of the metadata. As the data manager wanted to be able to check everything before making a DOI findable, all the DOIs minted with DataCite API are in draft status. Only when the data manager has completed the entire process of metadata transfer and file upload within MfN internal databases, and all the elements on the landing page are correctly displayed, the DOI status is manually modified in registered or findable, depending on whether there is an embargo on the dataset or not.

4 Software architecture and user experience

Two main criteria have been considered in designing the code for calling the DataCite API to mint and register DOIs automatically. The first one was the easy integration with the existing IT infrastructure at MfN and the microservice architecture that has guided its development (Bucchiarone et al. 2020). The second criterion was the necessity

6 The API url does not change. It is always `https://api.datacite.org`.
7 `https://support.datacite.org/docs/api`
8 `https://api.test.datacite.org` and `https://doi.test.datacite.org`
9 `https://datacite.readthedocs.io/en/latest/_modules/datacite/rest_client.html`

to create a user interface, as the data manager is not expected to have programming skills. Due to these requirements, it was decided to develop a Django web application running in a Docker container. The web framework Django was selected because it is Python-based, offers fast and secure development due to its several pre-coded features, and fulfils all the front-end (logging, messaging, etc.) and back-end (database connection, availability of web forms, etc.) requirements for the web application to interface with the DataCite API (Mele and Belderbos 2022).[10] In addition, it was easy to integrate with Docker, the technology of choice for running applications isolated in virtual software containers at MfN (Mouat 2015). Figure 3 is a schematic illustration of how the Django application works. The user interface allows log-in and log-out operations, access to the Django admin interface, and provides data managers with a webform to fill in to trigger the DOI registration process with DataCite. Via the front end, the user is notified of errors, warnings, and successfully completed DOI registrations. The back end of the web application manages all the interaction processes with DataCite and with the MfN databases essential for the DOI request and the functioning of the Django application.

Before triggering the DOI registration process, the back end queries the MfN Survey Database and checks that the submission number (an internal identifier for the DOI request form) is correct. If the submission number is not correct, the user is notified via the front-end interface. When a valid user request is received, it is the back end that collects the metadata for the DOI registration from the survey form with the given submission number, registers the required data (submission number, username, date and time of request creation) in the database associated to the Django admin interface, and calls the API to request the draft DOI and to put the metadata payload in JSON format. All the calls to the DataCite API are done using the available Python client mentioned above. The client greatly facilitates working with the API because it has all the required methods to transfer data to and from the DataCite service. In addition, the client enables to switch easily from the DataCite development API to the DataCite production API. This contributes to make the code ready for deployment quickly.

In the development of the Django application, a priority was to offer an effective and intuitive user interface to the MfN data manager. The application front end allows user authentication and, for the data management team, also enables access to the Django admin interface, where the submission numbers already processed can be checked. As the web application is designed only to register DOIs with DataCite, the web form that triggers the DOI registration process is immediately displayed to authenticated users. In the form, the data manager needs to provide only two pieces of information: their username and the submission number of the DOI request form in the MfN survey database (Figure 4). The submission number is the piece of information essential for the entire process. By using this number, it is possible to collect all the relevant metadata from the request form and start the process for minting the DOI with DataCite. As the submission number is the linking piece of information in the process, its existence in the MfN Survey Database is immediately checked, when the form is submitted. If the submission number

10 `https://www.djangoproject.com`

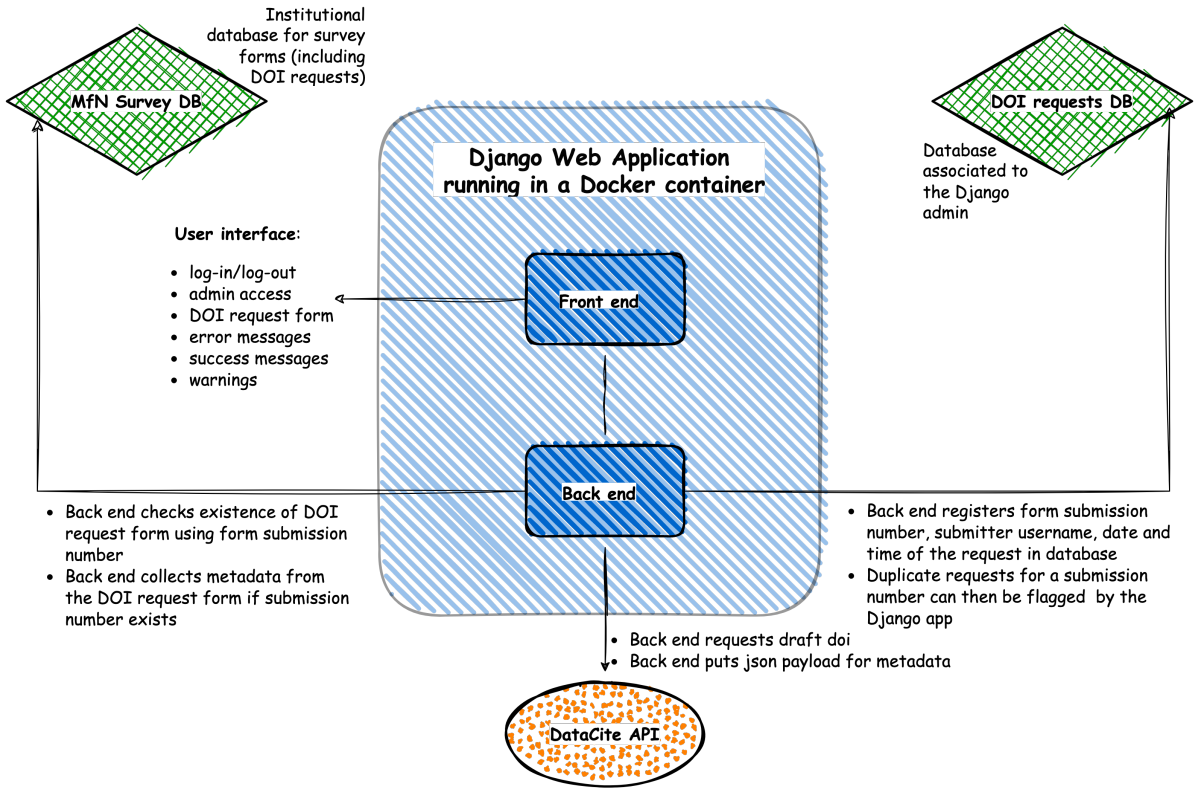

Figure 3: Code structure.

is not found among the DOI request forms in the database, an error message is generated and displayed to the user.

Once the user has typed a valid submission number, the application calls the DataCite API to generate a DOI and assign to the generated DOI the metadata extracted from the DOI request form available in the MfN Survey Database. Once the DOI has been successfully minted, a success message is displayed to the user (Figure 5 (left)). The success message contains information on the DOI created and the related submission number for which the DOI has been created. This allows the data manager full control of the process. In addition, using a button provided in the user interface, the data manager can easily copy the DOI with a click and paste it in the original survey form, the last passage before the form data can be transferred to MfN metadata database (see Figure 2). If the DOI has been generated for a submission number already present in the database associated to the Django admin interface, the user receives a warning message and can check for possible duplicates (Figure 5 (right)). On the success page, a link is available to return to the submission form, so that the data manager can restart the process immediately, if more DOIs need to be minted at the same time. By using the Django application developed, the DOI process can be carried out by the MfN data manager without using DataCite Fabrica user interface. This reduces at least a few manual steps in the DOI workflow described in Figure 2. As a result, there is not only some time-saving, but also better quality control

because the metadata are now associated to the DOI at the time of creation eliminating any risk for mismatching of metadata and DOI.

All the technologies used in the development of the web application (Docker, Django, etc.) are free to use for educational and non-commercial organisations. The DOI request process implemented by the application uses the available DataCite python client and it is portable and reusable by any organisation that needs to register/update a DOI via DataCite API. However, the way metadata are read from the survey database is specific to MfN and cannot be generalised to other institutions. Nonetheless, the tools used to read the metadata (sqlalchemy, mySQL connectors, etc.) are widely popular to integrate database queries in python scripts and can be recommended as a handy solution to work with DataCite client, regardless of the database type and configuration. The documentation for the sqlalchemy library (`https://www.sqlalchemy.org`) provides all the required information for connection to popular databases, even databases not explicitly considered here, such as PostgreSQL.

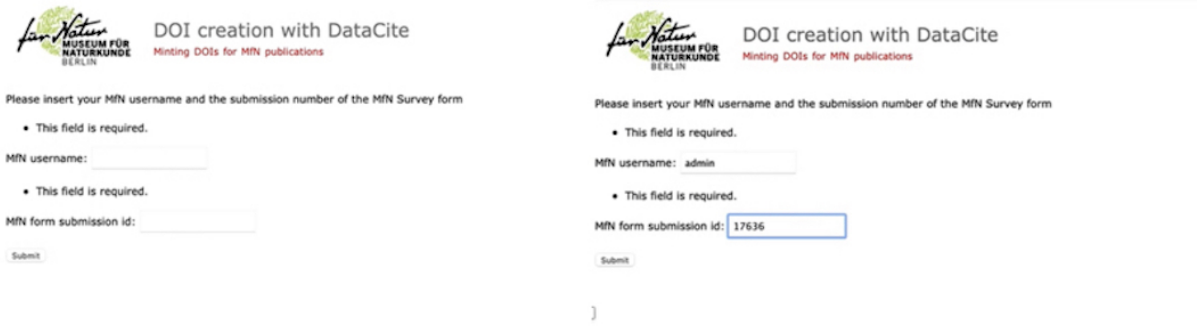

Figure 4: User interface. From left to right: 1) Django form for requesting a DOI; 2) Django form filled in. The images were taken during code development.

5 Research data management and automation

More demanding funder policies regarding research data and growing needs on the side of researchers, who are dealing with scientific data in quantities never experienced before, are constantly increasing the workload of research data management teams. Surveys conducted among library staff in several countries suggest that, alongside traditional advisory roles, such as preparation of data management plans or consultations on copyright and intellectual property, research data management teams are devoting more and more of their time to run data repositories, compile data catalogues, create metadata, rescue data, invest in long-term data preservation and data quality checks (Cox et al. 2019).

Usually, the workload increase has not been matched by a corresponding increase in staff and financial resources for research data management teams. Automation becomes, therefore, a necessary choice to avoid limiting or reducing the services that research data management teams provide to users. For services that are constantly requested, as it

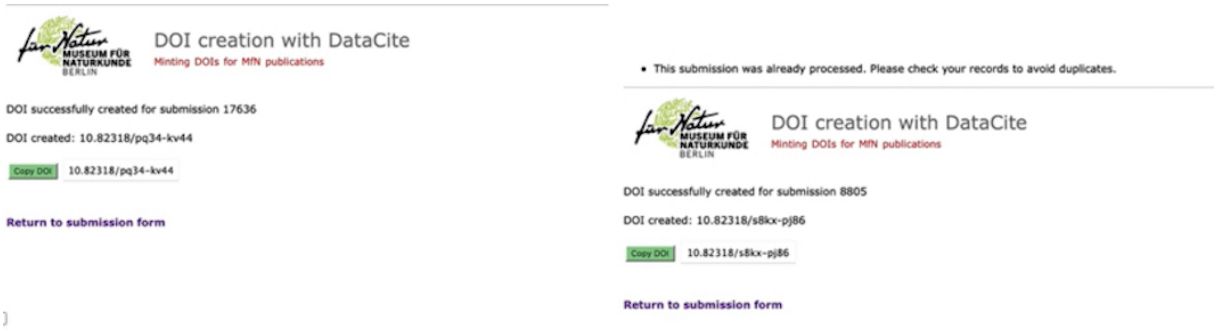

Figure 5: User interface. From left to right: 1) Success message for DOI creation related to the submission in Figure 4 (right); 2) Success message for DOI creation related to a submission number already present in the Django admin database. The user is warned about a possible duplicate. The images were taken during code development. The DOI prefix refers to the test repository made available to MfN by DataCite.

is the case of DOI registration at MfN, the initial time invested in developing code for automation is easily paid back by the time saved during daily operations. In addition, code development can be greatly facilitated by taking advantage of software already available and by following development strategies that are easy to implement and highly portable. This has been the philosophy behind the code development for automating the DOI process at MfN. Calling the DataCite API has been straightforward thanks to the already available API Python client. Furthermore, the microservices approach followed by the institution has made easier the integration of the new web application in the already existing IT ecosystem at MfN. The use of the web framework Django for creating the application has further accelerated the coding work. Each Django project already contains a template for the back end and the front end and this template can be customised and reused without the necessity to write all the code from scratch.

The arguments in favour of automation, however, are not confined to the limited resources of research data management teams and their increased workload. The concept of automation is interlinked with the concept of quality and this not just in manufacturing, which is the industrial sector in which modern automation procedures have their roots (Nof 2009). Key features of quality control, such as efficiency, productivity, and reliability, are all at stake in automation and this holds true also for automation in research data management. By shifting tasks from people to technologies in DOI registration we have set in motion a process that, on the one hand, allows data managers to be more efficient and productive in carrying out this task because they are able to achieve the goal with fewer steps, and, on the other hand, we have removed potential sources of mistakes that exist in manual operations, such as associating the wrong metadata to a DOI. The time that the data manager saves because there is no need to log in and out of multiple IT systems and to perform manual operations (like copying and pasting data from one system to the other) can be devoted to a much better use. At MfN the initial request of the user is the key source of metadata information for the DOI. To ensure the highest

quality in a FAIR publication process, it is, therefore, crucial that the data manager has sufficient time to review the request form in detail and, if necessary, interact with the user pointing out mistakes or inconsistencies in the metadata provided.

6 Conclusions

We have described an example of process automation in research data management implemented to meet the significant increase in DOI registration requests received at MfN. This is just an example of the benefits that task automation allows in research data management. Over time, we expect that there will be more interventions of this kind for shifting recurrent tasks, which do not require human control, from people to technologies.

We conceive this automation process as going through incremental steps, as suggested by an Agile approach to software development (Shore and Warden 2021). The web application described here is not a final product, but the first step of a more ambitious plan. Once the current version has satisfactorily performed in production, we plan to automate further. Currently, the data manager needs to manually copy the DOI minted into the request form in the MfN Survey Database (Figure 2. Step 3) and trigger the process of metadata transfer to the MfN metadata storage by filling in a web form with the submission number (Figure 2. Step 4). These two steps can be further automated by giving writing rights to the databases to the Django application that carries out the DOI registration process. In a further refinement, we also envision to automatically update the metadata registered in DataCite if there are changes in the MfN metadata database for the information related to a DOI. Further developments on the table also include the possibility to mint a DOI with a specific suffix and not just to mint a random DOI. This is a convenient feature when the data manager needs to create DOIs for a collection of related items or when a DOI must be assigned to versions of the same item that only differ in language.

Overall, we feel that there is a compelling argument in terms of time economy and improved quality control for automating tasks in research data management. At all stages of the data management process, there is both an increase in the quantity of research data available and in the tasks that need to be performed on these data to comply with funders' regulation, legal requirements, best practices to ensure FAIRness, etc. It is hardly realistic to think that a manual approach can be a solution to this transformation, given also that this trend is not going to stop, but rather it will continue with possibly an even more sustained growth in coming years. Only technologies can make a difference and free up valuable time for carrying out operations that really require human intervention in research data management, such as advisory tasks.

Acknowledgements

The authors thank the participants in the E-Science-Tage conference for the comments and suggestions received in Heidelberg. They are also grateful to the MfN colleagues, Mareike

Petersen and Caitlin Thorn, for the feedback received while developing the application user interface and user experience. The code development and the participation of one of the authors in the E-Science-Tage conference were kindly sponsored by MfN with internal funding.

References

Bucchiarone, Antonio, Nicola Dragoni, Schahram Dustdar, Patricia Lago, Manuel Mazzara, Victor Rivera, and Andrey Sadovykh, editors. 2020. *Microservices.* Springer International Publishing. DOI: https://doi.org/10.1007/978-3-030-31646-4.

Chard, Ryan, Kyle Chard, Jason Alt, Dilworth Y. Parkinson, Steve Tuecke, and Ian Foster. 2017. "Ripple: Home Automation for Research Data Management". In *2017 IEEE 37th International Conference on Distributed Computing Systems Workshops (ICDCSW),* 389–394. IEEE. DOI: https://doi.org/10.1109/ICDCSW.2017.30.

Cox, Andrew M., Mary Anne Kennan, Liz Lyon, Stephen Pinfield, and Laura Sbaffi. 2019. "Maturing research data services and the transformation of academic libraries". *Journal of Documentation* 75 (6): 1432–1462. DOI: https://doi.org/10.1108/JD-12-2018-0211.

Hobart, Mark. 2020. "The Business Case for Automating Data Management". *Journal of ICT Standardization* 8 (3): 199–216. DOI: https://doi.org/10.13052/jicts2245-800X.832.

Liu, Jia. 2021. "Digital Object Identifier (DOI) and DOI Services: An Overview". *Libri* 71 (4): 349–360. DOI: https://doi.org/10.1515/libri-2020-0018.

Masse, Mark. 2011. *REST API Design Rulebook: Designing Consistent RESTful Web Service Interfaces.* O'Reilly Media, Incorporated. ISBN: 9781449319908.

Mele, Antonio, and Bob Belderbos. 2022. *Django 4 by Example: Build Powerful and Reliable Python Web Applications from Scratch: Build Powerful and Reliable Python Web Applications from Scratch.* 4th edition. Packt Publishing, Limited. ISBN: 9781801813051.

Miksa, Tomasz, Simon Oblasser, and Andreas Rauber. 2021. "Automating Research Data Management Using Machine-Actionable Data Management Plans". *ACM Transactions on Management Information Systems* 13 (2): 1–22. DOI: https://doi.org/10.1145/3490396.

Mouat, Adrian. 2015. *Using Docker Developing and Deploying Software with Containers: Developing and Deploying Software with Containers.* O'Reilly Media, Incorporated. ISBN: 9781491915929.

Nof, Shimon Y., editor. 2009. *Springer Handbook of Automation.* Springer Handbooks. Springer Berlin Heidelberg. DOI: https://doi.org/10.1007/978-3-540-78831-7.

Shore, James, and Shane Warden. 2021. *Art of Agile Development.* 2nd edition. 530. O'Reilly Media, Incorporated. ISBN: 9781492080695.