Results of the First Phase of the OAF Project

Michael Kohlhase, Florian Rabe
FAU Erlangen-Nürnberg

January 18, 2021

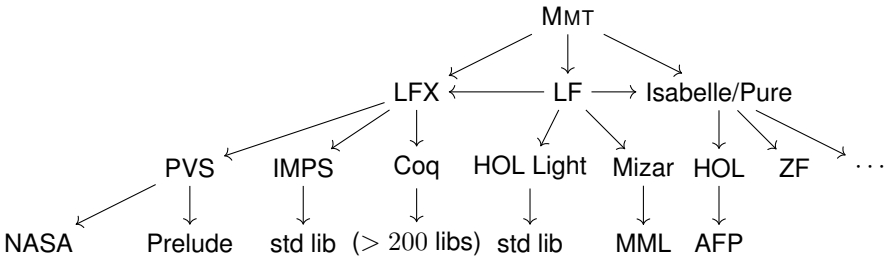

Figure 1: MMT as a universal framework for the libraries in OAF1

Summary

The goals of the first phase of the OAF project (we will write OAF1) were threefold: build a universal language and infrastructure for building an Open Archive of Formal libraries from mathematics and computer science (**WA 1**: "*Open Archive Infrastructure*"), seed it with imports of major libraries from a representative set of proof assistants (**WA 2**: "*Seeding the Archive with Libraries*"), and explore how to leverage such a universal archive for library integration (**WA 3**: "*Library Integration*") and library-independent services (**WA 4**: "*Global Services*").

The OAF-infrastructure is the first time ever that multiple diverse large libraries were available in a single universal representation language (specifically in our MMT language using OMDOC XML as the concrete representation for archival). Thus, the results from **WA 1+2** marked the first time that it was possible for anybody to investigate the research questions of **WA 3+4** at large scales. This part of OAF1 was a major success. Figures 1 and 2 gives overviews of the resulting overall theory graph inside the MMT framework and the involved libraries and their sizes. We expect spending most of the remaining time of OAF1 on polishing the system interface for viewing the imported libraries (**WP 1.2** and **WP 4.1**).

However, **WA 2** proved to be a critical bottleneck in practice: the library imports proved considerably harder than our (already very conservative) estimates — both in terms of the net OAF1 researcher time and (because it required enormous collaboration with external experts) in terms of overall elapsed time. They were very challenging in terms of theory (understanding advanced language features and designing new frameworks to represent them), practice (implementation, testing, and maintenance of the imports), and social logistics (find and coordinate with system expert collaborators, lobby proof assistant developer to add/improve export APIs). It is difficult to judge the time expenditure precisely because — except for Isabelle — none of the imports was done in a single chunk of time. We estimate we spent about 6 full-time person-months for each of the six imports. (The most basic Isabelle export took about 4 person-months, but it was extremely efficient as the language was relatively simple, and we could subcontract most of the work to the main Isabelle developer. The other exports took considerably longer, but we had preliminary work on HOL Light on Mizar.)

	Coq	HOL Light	IMPS	Isabelle	Mizar	PVS
Foundation	type th.	HOL	HOL + X	HOL	set th.	HOL + X
Collaborator	Sacerdoti Coen	Kaliszyk	Farmer	Wenzel	Urban	Owre
Export via[1]	instr.	instr.	instr.	instr.	gen.	gen.
External libraries[2]	> 200	-	-	AFP	MML	NASA lib.
Total # MMT theories	$2k$	0.2k	0.1k	10k	1k	1k
Total # MMT decls.	$170k$	$20k$	$1k$	$1,500k$	$70k$	$25k$
Src refs	statements	recoverable	recoverable	everywhere	none	everywhere

[1]: instr. = kernel instrumentation; gen. = via kernel-generated (XML) files

[2]: in addition to standard library; AFP = Archive of Formal Proof, MML = Mizar Mathematical Library

Figure 2: Overview of main library imports of OAF1

Because we are the first to carry out so many different imports and the first to do so at the involved scales, our experiences are very valuable to the community, and we are reporting on them in [KR20] (under review at JAR). The most important lesson we learned is that it is more practical to lobby, help with, and wait for upgrades to the proof assistant APIs (most importantly for Coq, PVS, and Isabelle) that enable scalable and maintainable imports, instead of developing ad-hoc solutions. This was the main reason why we opted for a cost-neutral extension of OAF1. That strategy paid off when we were able, in 2019, to export two of the largest and most valuable libraries, namely the ones of Isabelle and Coq. We had originally not promised those two because we had deemed them too difficult, and indeed neither export was feasible any earlier.

As anticipated in the original proposal, the full library integration problem requires far more time and resources than available within a single project. Therefore, **WA 3**, as planned, investigated and evaluated multiple different library integration approaches. Our main result is a clear understanding of how practical library integration solutions can and should be designed feasibly. Concretely, the community should use a star-pattern with a central lightweight formalization that focuses on introducing concepts and their properties but refrains from making definitions and proving theorems, thus avoiding a foundational commitment. Then alignments are used to relate its identifiers to those introduced in the various libraries. Sets of alignments can then induce (possibly approximate) translation functions across libraries. We reported on this approach in [Mül+17a; Koh+17b; Mül+17b]. A major part of the proposed second phase of the OAF project will be to develop this integration solution at large scale.

WA 4 developed a few services library-independent services. While we have developed these mostly successfully, our own applications and our collaborations with (potential) users made clear that some of the practically most important services are not actually in reach yet. In particular, navigating through, searching, and translating libraries can only be developed at practically relevant scales after building a large library of alignments needed for the library integration solution mentioned above.

People Employed on the Project

Dr. Rabe was employed for 36 months on his own position (Eigene Stelle). He contributed to all work packages, especially the overall architecture, the design of MMT, and the imports. He also co-supervised all employees on Prof. Kohlhase's on position. This setup proved critical especially for the imports in WA 2: these are extremely difficult even with senior supervision, but they become feasible if a PI can put in a month of full-time hands-on contribution to get the first version of an import working, at which point a PhD student or junior post-doc can take over.

The employees on Kohlhase's position (often billed at 50% of a position) were:

- Dr. Mihnea Iancu had already worked on preliminary results and carried out WP 1.2, 2.1, and 3.2 and WA 4 during the remainder of his PhD at (finished in 2017, grade: with distinction, title: "Towards Flexiformal Mathematics" [Ian17]).
- Dr. Dennis Müller was hired for OAF1 and was employed on it throughout his PhD (2014–2019, grade: very good, title: "Mathematical Knowledge Management Across Formal Libraries" [Mül19]). He was the key person for WP 2.4, 2.5, 2.6, and WA 3.

- Jonas Betzendahl is a recently hired junior PhD student and carried out WP 2.3.
- Dr. Christian Maeder was employed for 5 months as a senior research software developer for general system maintenance, e.g., improving the workflows in WP 1.3 and setting up the services for WA 4.
- Constantin Jucovschi was employed for 2 months as a PhD student for software development in WP 1.2.

Detailed Results by Work Package

WA 1: Open Archive Infrastructure All the infrastructure in this work has been developed as part of the MMT system. We have reported on the general setup and initial export facilities in [KR16].

WP 1.1: Back End We have extended the MMT system with memory management features as anticipated. However, due to the growth of main memory and the general switch to 64-bit machines (32-bit JVM applications performed badly when handling large amounts of data in memory), the practical bottlenecks have shifted to disk space management. As some of our imports now require triple-digit GB of OMDOC, we changed MMT to store all libraries in compressed files which decompress on the fly when loading data. With these optimizations, MMT's bottleneck for processing large libraries is now the time it takes to read and decompress OMDOC files.

Moreover, in order to better handle the growing number of libraries and using the fact that more MMT data structures can now be held in memory, we replaced the TNTBase system with a GitLab instance for versioned storage. This has met the OAF1 requirements very nicely, when coupled with the GIT LFS (Large File Storage) extension. Thus the specialized indexes could be replaced by in-memory data strutures, which were simpler to implement.

WP 1.2: Front End We have developed two web front-ends for OAF1: the regular, simplistic MMT web front-end (see e.g. `https://mmt.mathhub.info/`), and the MathHub system as an extended portal for mathematical libraries.

Originally, the MathHub user interface was based on Drupal, hoping to profit from an industry-strength platform with a large set of community-developed plugins, in particular user management, forums, and annotations. But the perceived advantage of using this Drupal-based front-end [Ian+14] turned into a liability because Drupal is a routine target for hackers, causing downtimes and maintenance headaches. Moreover, it turned out that almost all these services were better provided by more modern repository management infrastructures such as GitLab or in some case depended so much on logical aspects that they were easier to implement natively in MMT. Therefore, we dropped Drupal and developed a lightweight web application that complements the general purpose GitLab user interface with MMT-based logic-specific services. We describe it in detail together with OAF1 WA 4 below.

WP 1.3: Import/Export Workflows These have been implemented as planned and are the basis for the particular exports in **WA2**. For example, the export of the Isabelle library is regularly co-released with Isabelle and MMT [KRW19]. For the Coq library, which has now been split into hundreds of independently-maintained repositories, we developed workflows for automatically importing many libraries at once [MRS19].

WA 2: Seeding the Archive with Libraries As planned, we have developed exports for representative examples of all major proof assistant paradigms. We report extensively on the general challenges and our experiences in evaluating the various possible solutions in [KR20].

During the course of the project, we were able to slightly upgrade our goals by choosing more valuable examples (i.e., systems with more users and larger libraries) based on our growing expertise and the availability of cooperation partners. Concretely, we have imported six major proof assistant libraries: the ones of Coq and its many external libraries (based on type theory), HOL Light (based on higher-order logic), IMPS (based on heterogeneous reasoning), Isabelle and the Archive of Formal

Proof (a logical framework), Mizar (based on set theory), and PVS and the NASA library (using an undecidable type system). Additionally, we developed partial or initial exports for a variety of other systems including TPS and Specware, and an import for Metamath was contributed by others.

We changed our work program in regard to the *import of proofs*. The import of low-level proof terms is straightforward, but the proof objects are huge and have only limited value (independent proof checking being the main one). The high-level proofs seen by the user are much more interesting but lack the information inferred by the prover.

At this point, it is very difficult or impossible to export *high-level proofs* in a useful way. On the other hand, we can easily import low-level proof terms for all proof assistants with the caveat that in non-LCF systems (IMPS, Mizar, and PVS) only partial proof terms are ever available. However, the size of *low-level proofs* is often too big for current technologies: For example, for Isabelle, only the import of proof terms for small examples is feasible at all at this point despite major investments by main developer Makarius Wenzel, whom we subcontracted for the Isabelle export. For Coq, we imported all proof terms using a minimal representation that precludes the straightforward translation or rechecking of the proofs, and even so the biggest proof in the MathComp library (the one used in the formalization of the Odd Order Theorem [Gon+13]) is too big to export from Coq in any form. Due to the lack of implicit computation in Isabelle, we expect an export with proof terms for Isabelle to be even larger. We have identified two main technical problems that proof assistant development must tackle to make the export of low-level proof terms feasible: structure sharing must be preserved or re-introduced during the export (For example, Coq's structure sharing is syntactic rather than semantic and can therefore not be directly preserved.); and proof search using tactics and automated provers must be optimized for proof size as it is currently engineered exclusively to find some proof fast, which can occasionally unnecessarily explode the size of the found proof.

To find a good *compromise* between low- and high-level proof terms, we co-organized a Dagstuhl seminar on the issue of exporting and sharing proofs [Dow+17]. Here it became evident that the community has not found such a good compromise yet, and further research on a good proof interchange format is needed before it is worth investing heavily into the import of proofs into OAF. Therefore, we usually opted for exporting all proofs as dummy terms that carry only the information that the theorem was checked by the proof assistant and which dependencies were used. If possible, we also included, as an informal narrative text, the command-source of the proof.

WP 2.1: Mizar We maintained and expanded the Mizar import, which we had developed in preliminary work. We dropped the export of full proof terms: besides the size issue mentioned above, our collaborator Cezary Kaliszyk estimates at least 12 person-months would be needed to instrument Mizar to export proofs. This is because Mizar uses a big-kernel architecture with substantial complex reasoning in the kernel, and its code base is old and hard to understand for anybody but a few thinly-stretched core developers. (Mizar is maintained by a small underfunded developer community in Poland whose leader died shortly before OAF1 started.) For that reason, Mizar also lacks modern APIs that are becoming standard in other proof assistants and proved to be by far the hardest import to work with and maintain. After completion of OAF, the Mizar developers approached us with the news that they have redesigned their export interface to be more amenable to OAF1-style processing of the Mizar library, and we are currently working with them on upgrading the importer. This will also allow obtaining source references everywhere.

WP 2.2: HOL Light An initial export based on instrumenting the OCaml kernel structures had been established very early in the OAF project [KR14] in collaboration with Cezary Kaliszyk. In addition, we have fully implemented an export for Isabelle (whose main object logic Isabelle/HOL is very similar to the logic of HOL Light) and its huge AFP library, which we report on in [KRW19] (under review). This was the more challenging research goal because, while some HOL Light export existed before OAF1 [OS06], it was the first export for Isabelle. For the same reason, we explored the question of introducing structuring in the context of Isabelle rather than HOL Light.

WP 2.3: IMPS We have fully implemented the export and reported on it in [BK18; Bet18]. The main difficulty was reactivating the IMPS system, which has fallen out of use. But that also gave us better control of the source code so that it was easier to develop smooth export/import workflows for it.

WP 2.4: Matita Matita arose as a simpler reimplementation of Coq, and its developers and users now mostly see as an experimental system to study prover design, whereas Coq is a widely used production-ready system. We had initially chosen Matita because its export would be much simpler and happily switched to Coq when we realized we would be able to tackle the much harder Coq export. We have fully implemented an export for Coq and all its libraries and reported on it in [MRS19]. Particular difficulties were the enormous size of the export, and the split of the Coq library into hundreds of separately maintained libraries, which we had to export individually.

WP 2.5 PVS We have fully implemented the export and reported on it in [Koh+17a]. A particular difficulty was extending the logical framework LF with additional features (which we dub LF+X), specifically anonymous record types, predicate subtyping, and overloaded includes. We published some of these theoretical prerequisites separately in [MRK18].

WP 2.6: Standard Interface Library We have investigated two approaches for manually developing a standard interface library. Firstly, we developed a library of formal interface theories, which we call the Math-in-the-Middle (MitM) [MitM]. This curated library is written in the MitM foundation, a high-level logic defined in MMT that incorporates and integrates many of the features from the logics of the imported systems above. The MitM library covers algebra, arithmetics, set theory, set collections, graphs, measures/integrability, topology, and elementary calculus (55 files, 2600 LoC, 360 commits). MitM was already used in the OpenDreamKit EU project [Deh+16], where we extended MitM with interface theories for mathematical computation systems, and some of the MitM modules were written or extended in OpenDreamKit case studies.

Secondly, we developed a semi-formal *Semantic Multilingual Glossary of Mathematics* (SMGloM), see `https://gl.mathhub.info/smglom` written in sTeX, a semantic variant of LaTeX. The SMGloM framework was started in 2013 before OAF1, but throughout OAF1 we extended the data model and system functionality, and a significant part of the glossary content and the alignments were developed for the OAF1 case studies. The glossary contains about $1k$ glossary modules with more than 2k concepts and more than 4k verbalizations in English, German, Chinese, etc.

Both have complementary advantages: while formal interface languages tend to be too rigid to align with *all* proof assistant libraries, and informal ones tend to be too weak to build *all* desirable services for aligned proof assistant libraries As a lightweight integration of the two approaches, we use a couple of hundreds of alignments to cover the overlap between the two interface libraries. A main result of OAF1 is the initial design of new languages for writing an interface library that would allow a tight integration of the advantages of the two approaches.

WA 3: Library Integration In this work area, we explored and evaluated various approaches to library integration. Most of this work was conducted by Dennis Müller. Because some approaches worked better than others, the depth of the results is not the same in each work package.

WP 3.1: Interface Logics As anticipated we have developed the theoretical basis in terms of theory morphisms, but — also as anticipated — theory morphisms can be too heavyweight and rigid for practical applications, unless the logics have been designed for compatibility. Thus our approach has been twofold: we have developed a powerful meta-logical framework, which provides many of the salient features of the theorem prover logics. This allows expressing the object logics modularly, and keep the theory morphisms simple. Where this is impossible, we resort to alignments and alignment-based translation [Mül+17b; Mül+17a]. This approach is central in Müller's thesis [Mül19].

WP 3.2 Pragmatic Interfaces High-level pragmatic language features (such as module system, data types, interpretation hints) are difficult not only because they are often elaborated into kernel language and then hard to access in exports — there are also no strong frameworks in which these features can be described. Therefore, we have extended the MMT language and system with what we call *structural features*. That is an abstract logic-independent definition of a language feature that concrete language definitions can instantiate. We used this throughout our imports, i.e., to represent Mizar and HOL Light definitions, PVS inheritance, Coq sectioning, and Isabelle locales. We reported on this in [Mül+20].

WP 3.3: Interface Theory Generation We have conducted an initial case study, in which we manually extracted interfaces from proof assistant libraries [Mül+17b]. An automated extraction of abstract interfaces from our library exports would be straightforward (It amounts to throwing away all definitions.), but our experience showed that the libraries are too idiosyncratic to make automatically extracted interfaces valuable. Instead, we learned that interfaces should be written manually in a way that ignores the choice of type, module, and proof system made by the respective proof assistants.

WP 3.4 Theory Refactoring Theory refactoring is often critical for introducing structure in homogeneous libraries, which then improves interoperability. We have developed support in MMT for refactoring operations based on theory graphs, e.g., operations that work with a pair of inverse partial morphisms from a large theory to itself and then factor out their domains into separate, smaller theories. We have also developed the facility to compute theory intersections for factoring out the shared part of two theories [MK15]. We have also added MMT surface syntax generation for the generated theories and integrated it into the jEdit MMT IDE [IntV15].

WP 3.5: View Finder Library interoperability depends on identifying overlap, the automation of which is a very challenging long-term task. As a first major case study, we have developed a view finder algorithm and tested it both library-internally and in an intra-library mode. We have reported the results in [MKR18], and experiments on the larger libraries that have become available since then are ongoing. First results are encouraging in that we found several non-trivial views including some that humans were not aware of. But they also showed the limitations of logically rigorous approaches because idiosyncrasies of the logics (e.g., hard vs. soft typing) and library conventions (e.g., modules vs. records) often prevent the existence of views to begin with. This was a main reason that let us to develop the more flexible alignment-based methods. We have considered machine learning as well but the size of our imports is only now beginning to reach the scale where it is promising, and is a topic of Müller's current stay as DAAD scholar in Innsbruck.

Linked Data Moreover, expanding on the original proposal, we put work package–level resources into one approach that we had not considered originally: the use of Linked Data–style representations of the libraries. While these abstract from the mathematical meaning, they are conceptually easier and already provide strong results for many of the more shallow integration tasks. But theorem provers are even less engineered for linked data exports than they are for the exports done in OAF1 WA 2, and therefore, this approach became feasible only towards the end of the project. We reported on this in [Con+19], where we generated linked data representations comprising 0.4 individuals and 11.4 relations for a sample of 50 Coq libraries resp. $1,7$M individuals and 37M relations for Isabelle library and AFP. We used these for example for the cross-library querying we mention in OAF1 WP 4.1.

WA 4: Global Services All our contributions in this work area are integrated in and centered around our MathHub system at `http://mathhub.info`. MathHub is meant to become a general portal for reading, writing, managing, and interacting with formal mathematical libraries. It provides *i*) versioned hosting of mathematical data, knowledge, and documents via git repositories, *ii*) a community for sharing tools, results, and projects, *iii*) an online and offline document development infrastructure. Over the course of the OAF1 project, MathHub has become the central integration platform for the

OAF1 libraries and services. After a few prototypes, we have now settled on a stable and mature architecture based on GitLab for community and non-logical aspects of library management and MMT as the knowledge management engine (see Figure 3). Users can browse GitLab directly at `http://gl.mathhub.info` or use can MathHub, which uses GitLab's API as a backend and extends it MMT-based services. MathHub is based on React.JS, a docker swarm for the backends, and webpack for deployment mostly developed by Wiesing.

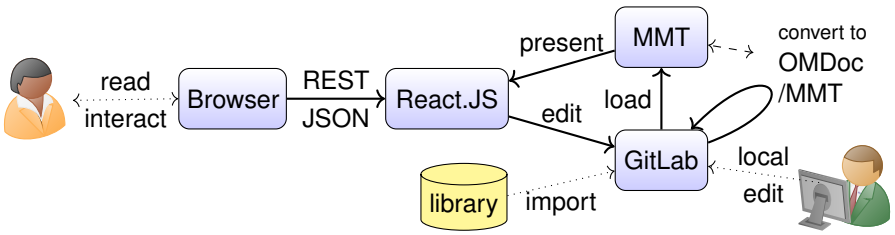

Figure 3: The MathHub Architecture

This development has refocused some of the research and development tasks in the work packages below:

WP 4.1 Search & Querying We have explored two basic avenues for search and querying of formal libraries. Firstly, we expanded on the MathWebSearch system for efficient formula retrieval [KMP12], which had been developed in an earlier project, largely as student projects. On the technological side, we have now consolidated the software infrastructure and made it deployable via docker compose [MWS]. MathWebSearch now consists of three parts: the harvester collects formulas from a corpus and transforms them into content MathML, the front-end allows composing queries in a familiar syntax and displays results, and the search daemon answers unification queries on them. Finally, we have completely re-developed the front-end in REACT.js, so that it can be integrated into MathHub [Alt19]; see `http://nlabsearch.mathweb.org` for an external deployment of this interface for the nLab. On the library side, we have developed harvesters for MMT content, which allows indexing and searching all the libraries imported in WA 2. We have also developed index completion methods that use induced content.

The second avenue is cross-library querying based on Linked Data–style relational properties of formal libraries (as mentioned at the beginning of the results of WA 3). For this approach we have developed a suggested standard ontology for formal library properties, which we call the *upper library ontology* (ULO) [ULO]. We can use ULO to express properties (e.g., which assertions are used in a proof of a theorem) as ULO-triples, collect them in a triple-store, and use that to answer SPARQL-based queries independently of the library. We have evaluated this approach for the Coq and Isabelle Libraries and reported on this and the applications to querying it enables in [Con+19].

Moreover, we have started the design of a federated query language that combines MathWebSearch, relational search, and text search. We reported on this in [BKR20].

WP 4.2 Versioning and Management of Change By switching the storage backend to Gitlab, we obtained highly scalable versioning and change management. The originally envisioned fine-grained change management proved to be less important: as proof assistant libraries are not released very frequently (typically a few times per year at most) and each import is expensive even when fully automated, the need for advanced logic-specific change management has not arisen in practice so far. Therefore, we have rededicated the resources of this WP towards maintaining the GitLab architecture (which proved to be more expensive than envisioned due to the size of the libraries that we were able to import).

WP 4.3: Generic Type/Proof Inference We have significantly extended the MMT system to allow for the rapid prototyping of formal systems. MMT is now able to perform type reconstruction in

a maximally logic-independent way and is able to mimic major implementations of formal systems, including e.g., dependent type theory, rewriting, or advanced data types. These language features are organized modularly in MMT theories so that system developers can mix and match language features when designing new systems. We reported on the theory in [Rab18], the implementation in a system description [Rab20], and on a number of case studies in [MR19]. Going beyond the core aim of OAF, we have started generating theorem provers from logic specifications in MMT. We reported on this in [Koh+20].

WP 4.4: Active Libraries MathHub serves the imported libraries from OAF1 WA 2 as active documents, i.e., documents that can be explored interactively. The general idea is that the primary documents are machine-oriented and users interact with them through multiple viewers that focus on different aspects. For example, http://mmt.mathhub.info⇝Isabelle⇝Distribution shows the Isabelle standard library in source-near form. Here the backend services (e.g., type inference based on WP 4.3) are developed in MMT, and MathHub integrates them into the user interface. The size of our imported libraries proved to be an unexpected bottleneck as casual users are unable to explore their high-level structure effectively. Therefore, we have developed special viewers that operate at the high level of the theory graphs (which are still huge graphs with up to thousands of node in our case): a browser-based viewer [Rup19; TGV] and a virtual-reality based one [MKR20] for VR goggles (see [T3W] for a WebGL-based web interface). The different viewers are interlinked with other, e.g., the VR viewer can project the source-near formal definition of a selected theory onto a virtual screen in the VR environment.

Outreach and Community Building

In many ways, the objectives of the overall project are so grand that they require a major community effort. Therefore, we have spent significant effort on outreach to raise awareness in the community and prepare for the current proposal. Together, we gave almost 40 overview presentations on OAF1 in general (not counting presentations of papers at workshops or conferences), including invited talks at dedicated week-long workshops at Schloss Dagstuhl, the Edinburgh International Centre for Mathematical Sciences, and the Hausdorff Center for Mathematics in Bonn.

We mention three collaborations of particular importance. We worked extensively with Gilles Dowek's Dedukti group in Paris, which pursues similar goals to ours (focusing more on proof verification than on library integration). This has recently culminated in a European proof assistant integration initiative that brings all stakeholders together and aims at developing an EU infrastructure proposal in March 2020.

We worked with Cezary Kaliszyk's group on the HOL Light import and library integration methods based on alignments. We will continue an active collaboration in the proposed OAF project phase. In fact, Dr. Müller, who has written his dissertation in OAF1, is currently on a six-month post-doc leave for a DAAD visit to Innsbruck working with Kaliszyk's group on the integration of narrative/inference aspects.

Within the Horizon 2020 project OpenDreamKit, we extended the ideas of OAF to systems for mathematical computation (including SageMath, GAP) and databases (including LMFDB, OEIS). We saw that all our results for framework, library export, integration, and services carry over to these domain, and that was a major motivation for the direction we are taking in OAF.

While impossible to quantify, this has had noticeable impact on the research interests of the community. For example, we were surprised to see the above-mentioned EU infrastructure proposal include a whole work area on alignments for proof assistant libraries.

References

[Alt19] Christoph Alt. "Diagram Chasing Done Meta – Formula Search for nLab". B.Sc. Thesis. FAU Erlangen-Nürnberg, Dec. 2019. URL: https://gl.kwarc.info/supervision/BSc-archive/blob/master/2019/Alt_Christoph.pdf.

[Bet18] Jonas Betzendahl. “Translating the IMPS Theory Library to MMT / OMDoc”. Master’s Thesis. Informatik, Universität Bielefeld, Apr. 2018. URL: https://gl.kwarc.info/supervision/MSc-archive/blob/master/2018/jbetzendahl/thesis_imps2omdoc.pdf.

[BK18] Jonas Betzendahl and Michael Kohlhase. “Translating the IMPS theory library to OMDoc/MMT”. In: *Intelligent Computer Mathematics*. Conferences on Intelligent Computer Mathematics. Ed. by Florian Rabe, William M. Farmer, Grant O. Passmore, and Abdou Youssef. LNAI 11006. Springer, 2018. ISBN: 978-3-319-96811-7. DOI: 10.1007/978-3-319-96812-4. URL: http://kwarc.info/kohlhase/papers/cicm18-imps.pdf.

[BKR20] Katja Berčič, Michael Kohlhase, and Florian Rabe. “Towards a Heterogeneous Query Language for Mathematical Knowledge”. submitted. 2020. URL: http://kwarc.info/kohlhase/submit/cicm20-search.pdf.

[Con+19] A. Condoluci, M. Kohlhase, D. Müller, F. Rabe, C. Sacerdoti Coen, and M. Wenzel. “Relational Data Across Mathematical Libraries”. In: *Intelligent Computer Mathematics*. Ed. by C. Kaliszyk, E. Brady, A. Kohlhase, and C. Sacerdoti Coen. Springer, 2019, pp. 61–76.

[Deh+16] Paul-Olivier Dehaye et al. “Interoperability in the OpenDreamKit Project: The Math-in-the-Middle Approach”. In: *Intelligent Computer Mathematics*. Conferences on Intelligent Computer Mathematics (Bialystok, Poland, July 25–29, 2016). Ed. by Michael Kohlhase, Moa Johansson, Bruce Miller, Leonardo de Moura, and Frank Tompa. LNAI 9791. Springer, 2016. ISBN: 978-3-319-08434-3. URL: https://github.com/OpenDreamKit/OpenDreamKit/blob/master/WP6/CICM2016/published.pdf.

[Dow+17] G. Dowek, C. Dubois, B. Pientka, and Florian Rabe. “Universality of Proofs (Dagstuhl Seminar 16421)”. In: *Dagstuhl Reports by Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik* 6.10 (2017). see http://drops.dagstuhl.de/opus/volltexte/2017/6951/, pp. 75–98.

[Gon+13] G. Gonthier et al. “A Machine-Checked Proof of the Odd Order Theorem”. In: *Interactive Theorem Proving*. Ed. by S. Blazy, C. Paulin-Mohring, and D. Pichardie. 2013, pp. 163–179.

[Ian+14] Mihnea Iancu, Constantin Jucovschi, Michael Kohlhase, and Tom Wiesing. “System Description: MathHub.info”. In: *Intelligent Computer Mathematics*. Conferences on Intelligent Computer Mathematics (Coimbra, Portugal, July 7–11, 2014). Ed. by Stephan Watt, James Davenport, Alan Sexton, Petr Sojka, and Josef Urban. LNCS 8543. Springer, 2014, pp. 431–434. ISBN: 978-3-319-08433-6. URL: http://kwarc.info/kohlhase/papers/cicm14-mathhub.pdf.

[Ian17] Mihnea Iancu. “Towards Flexiformal Mathematics”. PhD thesis. Bremen, Germany: Jacobs University, 2017. URL: https://opus.jacobs-university.de/frontdoor/index/index/docId/721.

[IntV15] Dennis Müller. *Theory Intersections in MMT*. URL: https://www.youtube.com/watch?v=qXKaGuV7kLY (visited on 06/29/2015).

[KMP12] Michael Kohlhase, Bogdan A. Matican, and Corneliu C. Prodescu. “MathWebSearch 0.5 – Scaling an Open Formula Search Engine”. In: *Intelligent Computer Mathematics*. Conferences on Intelligent Computer Mathematics (CICM) (Bremen, Germany, July 9–14, 2012). Ed. by Johan Jeuring, John A. Campbell, Jacques Carette, Gabriel Dos Reis, Petr Sojka, Makarius Wenzel, and Volker Sorge. LNAI 7362. Berlin and Heidelberg: Springer Verlag, 2012, pp. 342–357. ISBN: 978-3-642-31373-8. URL: http://kwarc.info/kohlhase/papers/aisc12-mws.pdf.

[Koh+17a] Michael Kohlhase, Dennis Müller, Sam Owre, and Florian Rabe. “Making PVS Accessible to Generic Services by Interpretation in a Universal Format”. In: *Interactive Theorem Proving 8th International Conference, ITP 2017*. Ed. by Mauricio Ayala-Rincón and César A. Muñoz. Vol. 10499. LNCS. Springer, 2017. ISBN: 978-3-319-66107-0. URL: http://kwarc.info/kohlhase/submit/itp17-pvs.pdf.

[Koh+17b] Michael Kohlhase, Dennis Müller, Markus Pfeiffer, Florian Rabe, Nicolas Thiéry, Victor Vasilyev, and Tom Wiesing. “Knowledge-Based Interoperability for Mathematical Software Systems”. In: *MACIS 2017*. Ed. by Johannes Blömer, Temur Kutsia, and Dimitris Simos. LNCS 10693. Springer Verlag, 2017, pp. 195–210. URL: https://github.com/OpenDreamKit/OpenDreamKit/blob/master/WP6/MACIS17-interop/crc.pdf.

[Koh+20] Michael Kohlhase, Florian Rabe, Claudio Sacerdoti Coen, and Jan Frederik Schaefer. “Logic-Independent Proof Search in Logical Frameworks (short paper)”. In: *10th International Joint Conference on Automated Reasoning*. accepted. Springer Verlag, 2020.

[KR14] Cezary Kaliszyk and Florian Rabe. “Towards Knowledge Management for HOL Light”. In: *Intelligent Computer Mathematics*. Conferences on Intelligent Computer Mathematics (Coimbra, Portugal, July 7–11, 2014). Ed. by Stephan Watt, James Davenport, Alan Sexton, Petr Sojka, and Josef Urban. LNCS 8543. Springer, 2014, pp. 357–372. ISBN: 978-3-319-08433-6. URL: http://kwarc.info/frabe/Research/KR_hollight_14.pdf.

[KR16] Michael Kohlhase and Florian Rabe. “QED Reloaded: Towards a Pluralistic Formal Library of Mathematical Knowledge”. In: *Journal of Formalized Reasoning* 9.1 (2016), pp. 201–234. URL: http://jfr.unibo.it/article/download/4570/5733.

[KR20] Michael Kohlhase and Florian Rabe. “Experiences from Exporting Major Proof Assistant Libraries”. 2020. URL: https://kwarc.info/people/frabe/Research/KR_oafexp_20.pdf.

[KRW19] Michael Kohlhase, Florian Rabe, and Makarius Wenzel. “Making Isabelle Content Accessible in Knowledge Representation Formats”. 2019. URL: https://kwarc.info/kohlhase/submit/isabelle-export.pdf.

[MitM] *MitM: The Math-in-the-Middle Ontology*. URL: https://mathhub.info/library/group?id=MitM (visited on 12/05/2019).

[MK15] Dennis Müller and Michael Kohlhase. “Understanding Mathematical Theory Formation via Theory Intersections in MMT”. In: *FMM - Formal Mathematics for Mathematicians*. 2015. URL: http://cicm-conference.org/2015/fm4m/FMM_2015_paper_2.pdf.

[MKR18] D. Müller, M. Kohlhase, and F. Rabe. “Automatically Finding Theory Morphisms for Knowledge Management”. In: *Intelligent Computer Mathematics*. Ed. by F. Rabe, W. Farmer, G. Passmore, and A. Youssef. Springer, 2018, pp. 209–224.

[MKR20] Richard Marcus, Michael Kohlhase, and Florian Rabe. “3-Dimensional Graph Visualization of Mathematical Knowledge”. submitted. 2020. URL: http://kwarc.info/kohlhase/submit/cicm20-tgview3D.pdf.

[MR19] D. Müller and F. Rabe. “Rapid Prototyping Formal Systems in MMT: Case Studies”. In: *Logical Frameworks and Meta-languages: Theory and Practice*. Ed. by D. Miller and I. Scagnetto. 2019, pp. 40–54.

[MRK18] Dennis Müller, Florian Rabe, and Michael Kohlhase. “Theories as Types”. In: *9th International Joint Conference on Automated Reasoning*. Ed. by Didier Galmiche, Stephan Schulz, and Roberto Sebastiani. Springer Verlag, 2018. URL: http://kwarc.info/kohlhase/papers/ijcar18-records.pdf.

[MRS19] D. Müller, F. Rabe, and C. Sacerdoti Coen. “The Coq Library as a Theory Graph”. In: *Intelligent Computer Mathematics*. Ed. by C. Kaliszyk, E. Brady, A. Kohlhase, and C. Sacerdoti Coen. Springer, 2019, pp. 171–186.

[Mül+17a] Dennis Müller, Thibault Gauthier, Cezary Kaliszyk, Michael Kohlhase, and Florian Rabe. “Classification of Alignments between Concepts of Formal Mathematical Systems”. In: *Intelligent Computer Mathematics*. Conferences on Intelligent Computer Mathematics. Ed. by Herman Geuvers, Matthew England, Osman Hasan, Florian Rabe, and Olaf Teschke. LNAI 10383. Springer, 2017. ISBN: 978-3-319-62074-9. DOI: 10.1007/978-3-319-62075-6. URL: http://kwarc.info/kohlhase/papers/cicm17-alignments.pdf.

[Mül+17b] Dennis Müller, Colin Rothgang, Yufei Liu, and Florian Rabe. “Alignment-based Translations Across Formal Systems Using Interface Theories”. In: *Fifth Workshop on Proof eXchange for Theorem Proving - PxTP 2017*. 2017. URL: http://jazzpirate.com/Math/AlignmentTranslation.pdf.

[Mül+20] Dennis Müller, Florian Rabe, Colin Rothgang, and Michael Kohlhase. “Representing Structural Language Features in Formal Meta-Languages”. submitted. 2020. URL: http://kwarc.info/kohlhase/submit/cicm20-features.pdf.

[Mül19] Dennis Müller. “Mathematical Knowledge Management Across Formal Libraries”. PhD thesis. Informatics, FAU Erlangen-Nürnberg, Dec. 2019. URL: https://opus4.kobv.de/opus4-fau/files/12359/thesis.pdf.

[MWS] *MathWebSearch*. URL: https://github.com/MathWebSearch (visited on 01/05/2020).

[OS06] S. Obua and S. Skalberg. “Importing HOL into Isabelle/HOL”. In: *Automated Reasoning*. Ed. by N. Shankar and U. Furbach. Vol. 4130. Springer, 2006.

[Rab18] F. Rabe. “A Modular Type Reconstruction Algorithm”. In: *ACM Transactions on Computational Logic* 19.4 (2018), pp. 1–43.

[Rab20] F. Rabe. “MMT: The Meta Meta Tool”. see https://kwarc.info/people/frabe/Research/rabe_mmtsys_20.pdf. 2020.

[Rup19] Marcel Rupprecht. “Visualization of Theory Graphs”. M.Sc. Thesis. FAU Erlangen-Nürnberg, May 2019. URL: https://gl.kwarc.info/supervision/MSc-archive/blob/master/2019/Rupprecht_Marcel.pdf.

[T3W] *TGView3D Web Version*. URL: https://tgview3d.mathhub.info (visited on 03/07/2019).

[TGV] *TGView@MathHub*. URL: https://mmt.mathhub.info/graphs/tgview.html (visited on 03/07/2019).

[ULO] *ULO Documentation*. URL: https://ulo.mathhub.info/ (visited on 03/11/2019).

[Wat+14] Stephan Watt, James Davenport, Alan Sexton, Petr Sojka, and Josef Urban, eds. *Intelligent Computer Mathematics*. Conferences on Intelligent Computer Mathematics (Coimbra, Portugal, July 7–11, 2014). LNCS 8543. Springer, 2014. ISBN: 978-3-319-08433-6.