Exchange of Geometric Information Between Applications

Pedro Quaresma
CISUC / Department of Mathematics
University of Coimbra, Portugal
pedro@mat.uc.pt

Vanda Santos
CISUC
University of Coimbra, Portugal
vsantos7@gmail.com

Nuno Baeta
CISUC
University of Coimbra, Portugal
nmsbaeta@gmail.com

The Web Geometry Laboratory (*WGL*) is a collaborative and adaptive e-learning Web platform integrating a well known dynamic geometry system. Thousands of Geometric problems for Geometric Theorem Provers (*TGTP*) is a Web-based repository of geometric problems to support the testing and evaluation of geometric automated theorem proving systems.

The users of these systems should be able to profit from each other. The *TGTP* corpus must be made available to the *WGL* user, allowing, in this way, the exploration of *TGTP* problems and their proofs. On the other direction *TGTP* could gain by the possibility of a wider users base submitting new problems.

Such information exchange between clients (e.g. *WGL*) and servers (e.g. *TGTP*) raises many issues: geometric search—someone, working in a geometric problem, must be able to ask for more information regarding that construction; levels of geometric knowledge and interest—the problems in the servers must be classified in such a way that, in response to a client query, only the problems in the user's level and/or interest are returned; different aims of each tool—e.g. *WGL* is about secondary school geometry, *TGTP* is about formal proofs in semi-analytic and algebraic proof methods, not a perfect match indeed; localisation issues, e.g. a Portuguese user obliged to make the query and process the answer in English; technical issues—many technical issues need to be addressed to make this exchange of geometric information possible and useful.

Instead of a giant (difficult to maintain) tool, trying to cover all, the interconnection of specialised tools seems much more promising. The challenges to make that connection work are many and difficult, but, it is the authors impression, not insurmountable.

1 Introduction

In the area of geometry there are now a large number of computational tools that can be used to perform many different tasks, dynamic geometry systems (DGS), computer algebra systems (CAS), geometry automatic theorem provers (GATP) among others [16]. These tools are being used in many different settings, in education, in research, etc. Apart from the set of examples that can be found in many of the systems currently available, there are several repositories of geometric knowledge (e.g. *Intergeo*,[1] *TGTP*,[2] *GeoGebra Materials*[3]).

When considering the current status, two questions arise:

- How systems that use geometric information and geometric information repositories can be put to work together?
- Should they be put to work together?

[1]The Interoperable Interactive Geometry for Europe.
[2]Thousand of Geometric problems for geometric Theorem Provers.
[3]GeoGebra Materials.

The answer to the second question is, in the authors opinion: *Yes*. For users of a system that manipulates information, the access to repositories of geometric information (problems, conjectures, proofs, etc.) will be very interesting. It will constitute a wide source of examples to work with, enlarging the (eventual) set of its own examples. For a collaborative build repository of geometric information enlarging its users/contributors base will be useful, thus enabling to open the repository to more, and of different types, contributions.

The answer to the first question lies on the answers to problems raised by the interconnection of those type of systems: search mechanisms (see Section 3.1); adaptive filtering of the geometric information (see Section 3.2); collaborative editing (see Section 3.3); technical issues raised by the geometric information interchange (see Section 3.4).

Overview of the paper. The paper is organised as follows: first, in Section 2, we introduce our test-case, the interconnection between the systems *WGL* and *TGTP*. In Section 3 we analyse the different problems and solutions in making the interconnection between clients and servers of geometric information. In Section 4 final conclusions are drawn.

2 Clients and Servers of Geometric Information

In this section two specific systems will be introduced. These systems are representative of the clients and servers systems in need of being connected. This will give an actual setting where the interconnection can and should be implemented.

The Web Geometry Laboratory (*WGL*) is a collaborative and adaptive e-learning Web platform integrating a well known DGS [19, 21]. Thousands of Geometric problems for geometric Theorem Provers (*TGTP*) is a Web-based repository of geometric problems to support the testing and evaluation of geometric automated theorem proving systems [15].

The interconnection of the two systems, *WGL* and *TGTP*, will reinforce the usefulness of each other.

The *WGL*$\rightarrow$*TGTP* interconnection will provide to the *WGL* users a large set of geometric constructions, giving them the possibility of browsing constructions and exploring conjectures and proofs. Teachers and students will gain the possibility of visualising geometric problems and link them with formal proofs [18].

The study of proofs in education is important as can be seen by the many contributions in proceedings of the ICMI Study 19 Conference: *Proof and Proving in Mathematics Education* [8]. A number of DGS already incorporates GATPs: *GCLC* incorporates four GATPs in it [11]; new versions of *GeoGebra* [10] already include a connection to GATPs allowing to give a formal answer to a given validation question [1]; *Java Geometry Expert* [26, 27, 28] incorporates several GATPs. Cinderella uses a technique called "randomised theorem checking", generating a large number of random examples, checking if the conjecture holds, establishing the truthfulness if the answer is yes for all examples generated [20].

None of above mentioned systems possess a repository of problems (some of them have unstructured lists of examples) open to users queries. The interconnection of clients and servers of geometric information will provide users with that open access to geometric information.

On the other hand, the *WGL*$\leftarrow$*TGTP* connection will allow to enlarge the users base of *TGTP*: teachers, and eventually students, could submit new conjectures. This will contribute to *TGTP*'s overall goal of providing a comprehensive repository of geometric problems, but will also enlarge its usefulness in education, given the fact that it is expected that secondary school teachers, and students, will contribute with problems close to the geometric subjects they are studying.

3 Issues in the Clients/Servers Interconnection

As in the case of *WGL*$\leftrightarrows$*TGTP* interconnection, the generic interconnection of clients and servers of geometric information raises many issues:

Searching: having repositories of geometric knowledge, users should be able to query the information in the most fruitful and easy way.

Adaptive Filtering: different users will have different goals when addressing a repository of geometric information: the user's level of geometric knowledge; the user's learning profile; the intended use of the geometric knowledge (e.g. educational setting, automatic deduction research).

Collaborative Editing: to allow repositories of geometric knowledge to grow, a collaborating editing interface, appealing and easy to use, should be implemented. The information introduced should contain keywords and other semantic information, allowing an easy querying.

Localisation: the *WGL* is an internationalised system, i.e. it has English as the base language and the Portuguese and Serbian translations (version 1.4). The *TGTP* is an English language only system. Setting aside any form of automatic translation, in this article that problem will not be approached, this is again an adaptive filtering issue, knowing the users preference the problems in the appropriated language(s) will be selected.

Technical issues: to make a connection between two different types of systems some technical issues must be addressed. The main ones are: specification of application interface protocols; common format for geometric information.

In the following these issues will be addressed.

3.1 Searching for Geometric Information

Having a repository of geometry knowledge a user should be able to browse the information in an easy and useful way. To a collaboratively built repository it is also important to avoid repetition and ensure consistency (e.g. non-contradictory facts). To answer both issues good search mechanisms are needed.

In *GeoGebra materials*[4] the search "Ceva" returned more than one hundred answers, most of them are about the same subject, "Ceva's Theorem". So a bit of excess of collaborative editing (many repetitions) and a simple text search mechanism. In the *Intergeo* site[5], there is a "simple text search" and a "complex text search". For the same text search ("Ceva") we get three results, two of them about "Ceva's Theorem" (a construction and an applet). The complex text search opens the possibility of adaptive filtering the information (e.g. choose the type activity: exercises; game; etc.). In *TGTP* [6] using the same simple text search returns one single result, "Ceva's Theorem". Also available is an extended text search and additionally, a novelty in the geometric information repositories, a geometric search mechanism, whose details will be explained below.

The above cited examples have different goals: generic geometric objects, for *GeoGebra materials* and *Intergeo*; conjectures to be proved by GATPs in *TGTP*. All systems share a common simple text search mechanism, *Intergeo* adds to that an adaptive filtering mechanism, and *TGTP* a geometric search mechanism.

[4]In 2018-01-11 there were 1041831 materials.
[5]In 2018-01-11 there were 3957 resources.
[6]In 2018-01-11 there were 236 problems.

A user of such systems will be looking for constructions, problems, conjectures, geometric information in a generic form. For example, *The circumcircle of a triangle*, *Ceva's Theorem*; *triangle concurrency points*, among many other possible queries.

Text Search The text search mechanisms are applied to one or more of the attributes that characterise the geometric objects in the repositories. This search mechanisms can be simple or complex, for example in *TGTP* the simple text query is done using *MySQL* regular expressions [13], over the `name` field in the database. A more powerful, text search mechanism is also available, using the *full-text search* of *MySQL* [13], over the fields `name`, `description`, `shortDescription` and `keywords`.

To facilitate the search, the information contained in the repositories should be enriched, manually or in some automatic way, with the use of geometric ontologies or with the addition of meta-information (e.g. keywords) that can characterise the geometric information [3].

TGTP does not have any automatic process for adding meta-information to the different entries in its repository. When inserting a new geometric conjecture the users can insert keywords characterising the new entry.

Geometric Search In geometry, apart from textual approaches common to other areas of mathematics, there is also the need for a geometric search approach, i.e. semantic searching in a corpus of geometric constructions.

We should be able to retrieve the geometric information contained in the many repositories of geometric knowledge. That is, we should be able to query for a given geometric construction having as result a set of similar geometric constructions, i.e. construction with, at least, the same geometric properties of the query construction.

Based on some preliminary work in geometric search [9], *TGTP*, apart from the two text queries mechanisms, has also a geometric search mechanism. The queries are constructed using a DGS and the geometric construction is semantically compared with the geometric constructions in the repository. The result of a query will be a list of geometric constructions, with the part matching the query highlighted.

To search, semantically, a given geometric constructions, it must be possible to perform inferential closure, starting with the geometric construction and obtaining a fix-point, the semantic description of that geometric construction, with all the properties that can be extracted (inferred) from it. The search mechanism will proceed by looking for other geometric constructions with a superset of these properties. The geometric construction, object of the query, is equal or a subfigure of those figures.[7]

The first version of the geometric semantic search mechanism is already implemented in *TGTP*, using the *GeoGebra* JavaScript applet for building the geometric query. The proposed geometric (semantic) search mechanism implemented in *TGTP* works as follows:

1. the user builds the geometric query using *GeoGebra*;
2. that geometric construction is transformed from the DGS format to a *predicate format*;
3. the *predicate format* is converted into a *conceptual graph*;
4. the inferential closure is performed;
5. the result is converted in a *global trail distribution* (GTD), a compact easy to manipulate representation of the conceptual graph;

[7] Yannis Haralambous and Pedro Quaresma, Geometric Figure Mining via Conceptual Graphs in preparation.

6. a database query is performed (almost all geometric constructions have a corresponding GTD), finding a list of all admissible candidates, i.e. constructions whose GTDs are a super-set of the query GTD;

7. using a subgraph isomorphism algorithm, identify the query construction as a subconstruction of the list of admissible candidates.

The last step is still to be implemented in *TGTP*. For now the process stops after getting the list of all admissible candidates and such list constitutes the output of the query.

In the current implementation the quality of the geometric search approach can be, somehow, misleading. If the user draws, three points, the lines connecting them and a circle, then the 96 (out of 193) constructions returned, are about the *incircle of a triangle*, *circumcircle*[8] and many other constructions involving circles, which is a very good fit (for the list of admissible candidates). Not all constructions on the previous geometric search are about triangles, but if we make a geometric search about three points and the lines connecting them, thus forming a triangle, the result is not the expected one, we get 193 (out of 193) answers, i.e. no useful selection was made. The reason for this is that the constructions in the database, and the geometric search also, are about points and lines, not having the triangle as object, so the match is made with all the constructions that have, at least, three points, and three lines. Not surprisingly all the constructions are more complex than that.

Summarising this section, we can conclude that some sort of complex text query mechanism (maybe a combination of *TGTP* and *Intergeo* text search) is needed, with filtering (see below). A geometric query mechanism will be also a very important addition.

3.2 Adaptive Filtering

The problems in a repository of geometric information must be classified in such a way that, in response to a user's query, only the problems in the user's level and/or learning style and/or interests and/or language(s) are returned.

Type of Users Different type of users will search for different type of information, e.g. the respective audience is different for *WGL* and *TGTP*: the former is constituted by secondary school teachers and students, the latter by computer science researchers, experts in geometric automated theorem proving. The same query made by *WGL* and by *TGTP* users should have different outcomes, adjusted to the user's expectations. Even if we consider only the *WGL* users, we should expect many different search goals, connected to the many different levels of geometric knowledge [5, 25] and also many different learning styles [2, 12, 14].

Localisation Issues A user of a client service, using a given language (e.g. Portuguese) may not want to receive the information from a server in a different language (e.g. English). This may be the case with *WGL* and *TGTP*, a secondary school student using *WGL* in Portuguese may have difficulties in dealing with an answer, in English, from the *TGTP* server. In our view, this is a complex problem and it will not be addressed in this article.

[8] *TGTP*'s geometric conjectures `GEO0281` and `GEO0328` respectively.

Taxonomies of Geometric Problems Taxonomies of geometric problems must be researched and implemented in the repositories of geometric knowledge in such a way that the users' queries can be filtered by a given criteria, adapting the result to the user. [9]

Current Status and Next Steps A first approach to adaptive filtering can be seen in the *Intergeo* project's "advanced search" mechanism. It has some adaptive filtering options, "Educational Levels" and "Trained Topics and Competencies", can be considered among others filters (see Figure 1). This allows a better fit to the different users needs.

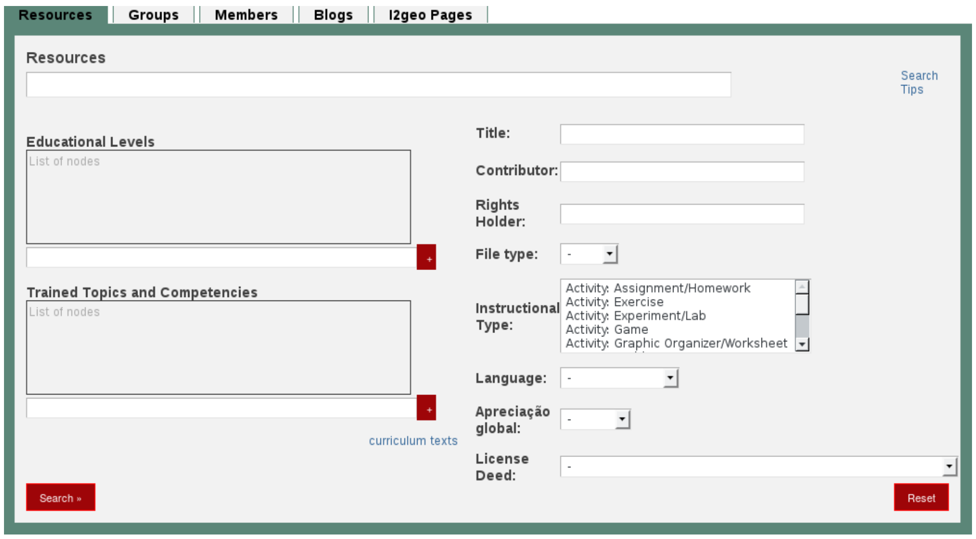


Figure 1: I2G Advanced Search Mechanism

The next step should be to implement adaptive features in such a way that the system would be capable of retaining information about the user, applying filters in an automatic way [4, 22, 24]. As far as we know none of the current systems has this feature.

3.3 Collaborative Editing

When building a repository with knowledge form a given area is now common practice to do it in a collaborative way, that is, open the manipulation of information contained in the repository to a large base of collaborators, enabling them to insert, update and delete "objects" of information.

All the cited repositories (see Section 1) work that way. The problem is how to avoid redundancy and incongruity. As the example in section 3.1 showed, we can fill a repository with many redundant information, reducing its usefulness. An even worst situation can occur when contradictory pieces of information are introduce into the repository.

Given the fact that the collaborative editing is not in question, the solution for the problems of redundancy and incongruity must be addressed by a powerful searching mechanism. Whenever a new piece of information is about to be introduced, the repository must be searched for an equal or similar piece of

[9]Pedro Quaresma, Vanda Santos, Pierluigi Graziani and Nuno Baeta, *Taxonomies of Geometric Problems*, to be submitted to a special issue of Journal of Symbolic Computation, on Dynamic Geometry and Automated Reasoning, full paper version of a extended abstract accepted at ThEdu'17.

information, giving to the collaborator the choice to give up (the “new” information is not new after all), or update the “old” information with new facts.

3.4 Technical Issues

Connecting “clients” and “servers”, i.e. geometric knowledge manipulator applications (e.g DGS, GATP) and repositories of such knowledge, can be done in two major ways.

- A one-to-one specific connection.

  For example we can use two systems, *WGL* and *TGTP*, and connect them in such a way that *WGL* will send the query in the exact format that *TGTP* is expecting and *TGTP* will reply with an answer specifically tailored to *WGL*.

  We can even think in a solution where the mechanisms (algorithms) implemented in one system are replicated in the other system. For example, opening the access to *TGTP*’s database to the *WGL* system and copying the search mechanisms of *TGTP* to *WGL*, with the smallest number of changes to make it work within *WGL*.

- A many-to-many generic connection.

  Like the *Open Data Base Connectivity* standard (ODBC), the application programming interface (API) for accessing database management systems [23], a *geometric API* is needed.

  The specification of such an API will allow to establish an easy (or at least standard) way to interconnect “clients” and “servers” of geometric knowledge in a generic way. The existence of such an API would allow the interconnection between systems developed by different groups, systems with different aims, systems with different types of users, but with the common “object”: geometry.

The one-to-one specific connection is just that. A connection specific to two specific systems. For *WGL* and *TGTP* such task would be easy, taking into account that the developer teams of both systems overlap. For two systems developed by two different teams it will be, in our opinion, a difficult negotiation to start with and, in the long run, an almost impossible interconnection to maintain, whenever both systems are still evolving.

In view of that, the many-to-many generic connection seems a more sensible investment. What are the difficulties in establishing such an API? What are the steps already done? What are the steps still to be done?

When interconnecting two systems, a “client” and a “server” we have to specify how the client will send the requests to the server and how the server answers. Two distinct situations: the client wants to query the server; the client wants to insert (or modify) information in the server.

The first question has to do with searching and adaptive filtering. The second with collaborative editing. In both cases the specification of an API is needed.

The geometric API should specify:

- how to send text queries (simple or complex) to the server;
- how to send geometric queries to the server;
- how to send filters to the server;
- how to combine filters and queries (filtered queries);
- how to send users profiles (for the adaptive features in the server);

- how to send new information, or update existing information, to the server;
- the server's response.

In the following, the description of a first experiment in the *WGL⇆TGTP* test-case will be describe.

3.5 WGL Meets TGTP

The connection between clients and a server is a question of sending the "sentence" or the geometric construction a user wants to query.

The text queries can be sent as a string. A format like *JSON*, a lightweight data-interchange format,[10] could be used to send the query.

The geometric queries are more complex. As both *WGL* and *TGTP* use the same DGS, *GeoGebra*, the ggb file could be sent by *WGL*, received by *TGTP*, processed and a list of ggb files would be sent back to *WGL*. But, in a more generic way, a common format like I2GATP should be used. The I2GATP format [17] is an extension of the I2G (Intergeo) common format[11] [6] that also supports conjectures and proofs produced by geometric automatic theorem provers. The I2GATP library is an open source project,[12] implemented in C++, to support the I2GATP common format. The library implements methods to manage the I2GATP container and filters between different GATP/DGS.

WGL* Querying *TGTP In order to make the connection *WGL→TGTP*, allowing *WGL* access the *TGTP* database, two applications were created, a *WGL* client and a *TGTP* server.

The flow of information between *WGL* and *TGTP* is as follow.

1. In the *List of Constructions WGL*'s Web page, the *WGL*'s user has a button *Query the TGTP server*, that opens a form, allowing the introduction of the query.
2. The *WGL* client program is called (a *system call*) with that query (string) as one of the command line arguments.[13]
3. The *WGL* client will create a *JSON* object with the query and the filter that asks for *GeoGebra* constructions only. The *JSON* object is then sent to the *TGTP* server.

   The format of the *JSON* object sent by the *WGL* client is:

   ```
   {
     "Query" : "<query>",
     "Filters" : "<filter> | <filter1 AND ... AND filterN>"
   }
   ```

   where the second component ("Filters") is optional.

4. The *TGTP* server receives the *JSON* object, parses it, builds the *SQL* query, sends it to the *TGTP* database server, receives the answer, produces the corresponding *JSON* object and sends it back to the *WGL* client.

   The format of the *JSON* object sent by the *TGTP* server is:

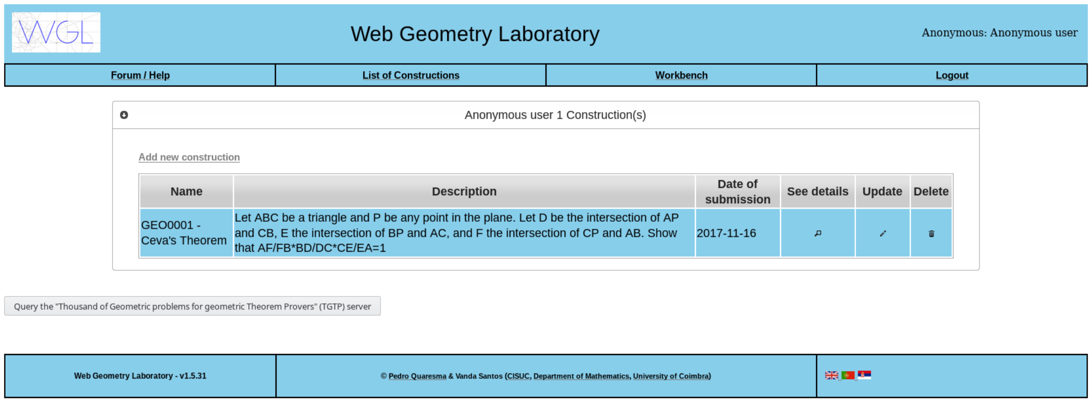
[10] http://www.json.org/
[11] http://i2geo.net
[12] https://github.com/GeoTiles/libI2GATP
[13] The command line call is "./clientWGL <TGTPserver_name> <port> <query>"

```
{
  "Theorem Identifier1" :
      "{"Name":"<name1>","Description":"<description1>","Code":"<code1>"
       ...,
  "Theorem IdentifierN" :
       "{"Name":"<nameN>","Description":"<descriptionN>","Code":"<codeN>"
}
```

5. The *WGL* client receives the *JSON* object (closing the network connection), parses it, builds a *SQL* instruction corresponding to the insertion of the new construction(s) and sends it to the *WGL* database server.
6. The *WGL* Web page is then reloaded (automatically), updating the user's list of constructions.

Figure 2 show the anonymous user's list of constructions, just after the query "ceva" was sent. The new construction was incorporated in the previously empty list of construction and it is now up to the user to make the best use of the new information.

The server is an application that is listening, in an infinite cycle, from a given socket (*IP:port*). The clients are applications that make requests to that socket, sending the query, and receiving the answer. The current experimental implementation uses the anonymous user of *WGL* and the number of successful queries is limited because we are looking for (native) *GeoGebra* constructions in *TGTP* without using (for now) the converters from other formats.


WGL
Web Geometry Laboratory
Anonymous: Anonymous user
Forum / Help
List of Constructions
Workbench
Logout

Anonymous user 1 Construction(s)

Add new construction

Name	Description	Date of submission	See details	Update	Delete
GEO0001 - Ceva's Theorem	Let ABC be a triangle and P be any point in the plane. Let D be the intersection of AP and CB, E the intersection of BP and AC, and F the intersection of CP and AB. Show that AF/FB*BD/DC*CE/EA=1	2017-11-16	⌕	✎	🗑

Query the "Thousand of Geometric problems for geometric Theorem Provers" (TGTP) server
Web Geometry Laboratory - v1.5.31
© Pedro Quaresma & Vanda Santos (CISUC, Department of Mathematics, University of Coimbra)

Figure 2: List of Constructions (after the query)

Both programs, `clientWGL` and `serverTGTP`, were written in *C++* with the libraries *cppconn*, *socket* and *ptree*, for the *MySQL* connection, the network connection between clients and servers and *BOOST* property trees library and *JSON* parser methods, respectively.

The client program is an open source program, contained in the *TGTP* project.[14] Anyone can write clients to access the *TGTP* server.

***TGTP* Enlarged Users Base** Assuming that the repository (server) uses the I2GATP format, any insertion and/or updating should be done by building the container, on the client side, and afterwards send it to the server. As the I2GATP associated library has methods to manipulate the I2GATP containers, it

[14] `https://github.com/GeoTiles/TGTP`

should be a matter of collecting all the information needed and then, using the appropriated methods, build the container and send it to the server.

Using, again, the test-case as an example: *TGTP* already uses the I2GATP format to store the problems in its database; from the *WGL* side it would be a question of collecting all the information needed, build the I2GATP container using the methods in the I2GATP library and send the information to the *TGTP* server. This is still to be implemented.

4 Conclusions and Future Work

Instead of a giant (heavy and difficult to use and maintain) tool, trying to cover all features of the many specialised tools, the interconnection of those specialised tools seems much more promising. The connection between *WGL* and *TGTP* is only an example of the connections that should be made between geometric tools. Efforts like *OpenGeoProver*, a open-source project of a library of GATPs,[15] to be used by different systems (e.g. *GeoGebra* [1]), or the connection between DGS and CAS [7], share the common approach of "mosaic tool", a tool composed of many small pieces forming a beautiful picture.

The main challenge faced by developers, when trying to connect different types of systems, is the definition (and implementation) of common formats. Having the I2GATP common format as a starting point and the corresponding open source library `libI2GATP` as a form to manipulate the I2GATP container, we should enlarge both as needed to allow different systems to exchange information.

The definition of an *ODBC* counterpart for geometry would allow to have a layer between clients and servers, isolating the nasty details from the developer and thus allowing an easy interconnection between systems.

The other big challenge is the retrieving of information, i.e. queries sent from the clients to the servers. The definition of a query language (an *SQL* counterpart) for geometry, allowing to explore text and geometric queries is needed.

Another issue is implementing adaptive features. Again a specification of what are the students of geometry learning profiles should be pursued.

We feel that having identified the main issues and having already done some steps, it is, now, a question of filling in the gaps to make it possible.

References

[1] Francisco Botana, Markus Hohenwarter, Predrag Janičić, Zoltán Kovács, Ivan Petrović, Tomás Recio & Simon Weitzhofer (2015): *Automated Theorem Proving in GeoGebra: Current Achievements*. *Journal of Automated Reasoning* 55(1), pp. 39–59, doi:10.1007/s10817-015-9326-4.

[2] Peter Brusilovsky et al. (1998): *Adaptive educational systems on the world-wide-web: A review of available technologies*. In: *Proceedings of Workshop "WWW-Based Tutoring" at 4th International Conference on Intelligent Tutoring Systems (ITS'98), San Antonio, TX*. Available at `http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.472.6285&rep=rep1&type=pdf`.

[3] Xiaoyu Chen & Dongming Wang (2012): *Management of geometric knowledge in textbooks*. *Data & Knowledge Engineering* 73(0), pp. 43–57, doi:10.1016/j.datak.2011.10.004.

[4] Konstantina Chrysafiadi & Maria Virvou (2013): *Student modeling approaches: A literature review for the last decade*. *Expert Systems with Applications* 40(11), pp. 4715–4729, doi:10.1016/j.eswa.2013.02.007.

[15] `https://github.com/ivan-z-petrovic/open-geo-prover/`

[5] Mary L. Crowley (1987): *The van Hiele Model of the Development of Geometric Thought*. In Mary Montegomery Lindquist, editor: *Learning and Teaching Geometry, K12*, chapter 1, Yearbook of the National Council of Teachers of Mathematics, National Council of Teachers of Mathematics, Reston, VA, USA, pp. 9–23. Available at `http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.456.5025&rep=rep1&type=pdf`.

[6] Santiago Egido, Maxim Hendriks, Yves Kreis, Ulrich Kortenkamp & Daniel Marquès (2010): I2G *Common File Format Final Version*. Technical Report D3.10, The Intergeo Consortium. Available at `http://i2geo.net/files/deliverables/D3.10-Common-File-Format.pdf`.

[7] Jesús Escribano, Francisco Botana & Miguel A. Abánades (2010): *Adding remote computational capabilities to Dynamic Geometry Systems*. *Mathematics and Computers in Simulation* 80(6), pp. 1177–1184, doi:10.1016/j.matcom.2008.04.019.

[8] Gila Hanna & Michael de Villiers, editors (2012): *Proof and Proving in Mathematics Education*. *NISS* 15, Springer, doi:10.1007/978-94-007-2129-6.

[9] Yannis Haralambous & Pedro Quaresma (2014): *Querying Geometric Figures Using a Controlled Language, Ontological Graphs and Dependency Lattices*. In S. Watt et al., editor: *CICM 2014*, *LNAI* 8543, Springer, pp. 298–311, doi:10.1007/978-3-319-08434-3_22.

[10] Markus Hohenwarter (2002): *GeoGebra - a software system for dynamic geometry and algebra in the plane*. Master's thesis, University of Salzburg, Austria.

[11] Predrag Janičić (2006): *GCLC — A Tool for Constructive Euclidean Geometry and More Than That*. In Andrés Iglesias & Nobuki Takayama, editors: *Mathematical Software - ICMS 2006*, *Lecture Notes in Computer Science* 4151, Springer, pp. 58–73, doi:10.1007/11832225_6.

[12] Myung-Geun Lee (2001): *Profiling students' adaptation styles in Web-based learning*. *Computers & Education* 36(2), pp. 121–132, doi:10.1016/S0360-1315(00)00046-4.

[13] Oracle (2011): *MySQL 5.5 Reference Manual*, 5.5 edition. Oracle. Available at `http://dev.mysql.com/doc/`. Revision: 24956.

[14] Kyparisia A. Papanikolaou, Maria Grigoriadou, George D. Magoulas & Harry Kornilakis (2002): *Towards new forms of knowledge communication: the adaptive dimension of a web-based learning environment*. *Computers & Education* 39(4), pp. 333–360, doi:10.1016/S0360-1315(02)00067-2.

[15] Pedro Quaresma (2011): *Thousands of Geometric Problems for Geometric Theorem Provers (TGTP)*. In Pascal Schreck, Julien Narboux & Jürgen Richter-Gebert, editors: *Automated Deduction in Geometry*, *Lecture Notes in Computer Science* 6877, Springer, pp. 169–181, doi:10.1007/978-3-642-25070-5_10.

[16] Pedro Quaresma (2017): *Towards an Intelligent and Dynamic Geometry Book*. *Mathematics in Computer Science* 11(3), pp. 427–437, doi:10.1007/s11786-017-0302-8.

[17] Pedro Quaresma & Nuno Baeta (2015): *Current Status of the I2GATP Common Format*, pp. 119–128. Springer International Publishing, doi:10.1007/978-3-319-21362-0_8.

[18] Pedro Quaresma & Vanda Santos (2016): *Visual Geometry Proofs in a Learning Context*. In Walther Neuper & Pedro Quaresma, editors: *Proceedings of ThEdu'15*, *CISUC Technical Reports* 2016001, CISUC, pp. 1–6. Available at `https://www.cisuc.uc.pt/ckfinder/userfiles/files/TR%202016-01.pdf`.

[19] Pedro Quaresma, Vanda Santos & Milena Marić (2017): *WGL, a web laboratory for geometry*. *Education and Information Technologies*, doi:10.1007/s10639-017-9597-y.

[20] Jürgen Richter-Gebert & Ulrich H. Kortenkamp (2000): *User Manual for the Interactive Geometry Software Cinderella*. Springer-Verlag Berlin Heidelberg, doi:10.1007/978-3-642-58318-6.

[21] Vanda Santos, Pedro Quaresma, Helena Campos & Milena Marić (2016): *Web Geometry Laboratory: Case Studies in Portugal and Serbia*. *Interactive Learning Environments* 26(1), pp. 3–21, doi:10.1080/10494820.2016.1258715.

[22] Valerie J. Shute & Joseph Psotka (2001): *The Handbook of Research for Educational Communications and Technology*, 1st edition, chapter Intelligent Tutoring Systems: Past, Present, and Future, pp. 570–600.

The Association for Educational Communications and Technology. Available at `https://www.aect.org/edtech/ed1/pdf/19.pdf`.

[23] Abraham Silberschatz, Henry Korth & S. Sudarshan (2010): *Database System Concepts*, 6th edition. McGraw-Hill Education - Europe, New York.

[24] Evangelos Triantafillou, Andreas Pomportsis & Stavros Demetriadis (2003): *The design and the formative evaluation of an adaptive educational system based on cognitive styles*. Computers & Education 41(1), pp. 87–103, doi:10.1016/S0360-1315(03)00031-9.

[25] Zalman Usiskin (1982): *van Hiele Levels and Achievement in Secondary School Geometry*. Technical Report, University of Chicago. Available at `http://ucsmp.uchicago.edu/resources/van_hiele_levels.pdf`.

[26] Zheng Ye, Shang-Ching Chou & Xiao-Shan Gao (2010): *Visually Dynamic Presentation of Proofs in Plane Geometry, Part 1*. J. Autom. Reason. 45, pp. 213–241, doi:10.1007/s10817-009-9162-5.

[27] Zheng Ye, Shang-Ching Chou & Xiao-Shan Gao (2010): *Visually Dynamic Presentation of Proofs in Plane Geometry, Part 2*. Journal of Automated Reasoning 45, pp. 243–266, doi:10.1007/s10817-009-9163-4.

[28] Zheng Ye, Shang-Ching Chou & Xiao-Shan Gao (2011): *An Introduction to Java Geometry Expert*. In Thomas Sturm & Christoph Zengler, editors: Automated Deduction in Geometry, Lecture Notes in Computer Science 6301, Springer Berlin Heidelberg, pp. 189–195, doi:10.1007/978-3-642-21046-4_10.