Current Status of the I2GATP Common Format

Conference Paper · July 2015

DOI: 10.1007/978-3-319-21362-0_8

2 authors:

Pedro Quaresma
University of Coimbra
43 PUBLICATIONS **182** CITATIONS
SEE PROFILE

Nuno Baeta
Instituto Superior de Engenharia de Coimbra
4 PUBLICATIONS **5** CITATIONS
SEE PROFILE

Current Status of the I2GATP Common Format

Pedro Quaresma[1] and Nuno Baeta[2]

[1] CISUC/Department of Mathematics, University of Coimbra
Coimbra, Portugal, `pedro@mat.uc.pt`
[2] ISEC, Polytechnic Institute of Coimbra
Coimbra, Portugal, `nmsb@isec.pt`

Abstract. The I2GATP format is an extension of the I2G (Intergeo) common format aimed to support conjectures and proofs produced by geometric automatic theorem provers. The goal in building such a format is to provide a communication channel between different tools from the field of geometry, allowing linking such tools, as well as allowing the use of geometric knowledge kept in different repositories.
In this article we report the current status of the I2GATP format and its accompanying components: the XSD files with the specification of the format; the C++ library to create the container with all the information regarding a geometric problem or to break it into its components; the filters to convert from/to geometric tools formats to/from I2GATP; the integration with repositories of geometric knowledge.

1 Introduction

The I2GATP format is an extension of the I2G (Intergeo) common format [14] aimed to support conjectures and proofs produced by geometric automatic theorem provers (GATP). As such, it is to be used by tools from the field of geometry, allowing its linking: in geometric knowledge repositories like TGTP [11] and GeoThms [4] as a common format; in a learning environment for geometry like the Web Geometry Laboratory (WGL) [13, 15, 16] connecting geometric tools, allowing the formal validation of the geometric constructions and even the introduction of formals proofs in a learning environment.

In this article we report the current status of the I2GATP format, i.e., the XSD[3] files, the programs to build the container and break it into its components; the filters between the format and the different geometric tools; integration within repositories of geometric knowledge, learning environments for geometry and other "clients" of geometric knowledge.

The XSD files contain the specification of the format: `information.xsd` with the meta-information about a given geometric problem; `intergeo.xsd` no more than the XSD for the I2G format; `conjecture.xsd` with the specification of the conjectures and `proofInfo.xsd` with the meta-information about the proof(s).

All the XML files containing the information about a geometric problem and also other auxiliary files, are packaged in the I2GATP container, an extension of

[3] An XML Schema (XSD) file describes the structure of an XML document.

the I2G container. The packaging and also the opposite operation of breaking-off the container into its components is a part of the auxiliary library built to support the I2GATP format.

To allow the linking of geometric tools, filters from/to the I2GATP need to be implemented. We begin by considering the filters needed to convert from/to languages of the GATPs contained in the TGTP repository to/from the I2GATP format.

Having made the previous steps, one last step is the integration of all this in repositories of geometric knowledge such as TGTP [11] and GeoThms [4]. It will allow the use of the common format to store all the information regarding one given geometric problem and then break it into components, e.g., to feed a conjecture to a given GATP. In the opposite direction, we can build the I2GATP container for any problem contained in the TGTP repository.

From the previous report about this project [12], the XSD and the container specifications have been improved; the open source library was built and provides now some of the filters needed alongside with the programs to manage the container file; and the integration with the TGTP system is underway.

Related Work The geometry description language (GDL) propose by Chen [1] aims to convert a natural (mathematical) language description of geometric problems, as found in the literature, to an equivalente in a formal language that can be automatically processed and convert to system-native representations. The interconections between this work and ours has to be explored.

Paper Overview. In Section 2 the overall structure and the status of the format are described. In Section 3 questions about the container are described. In Section 4 the implementation of the library and its integration with repositories of geometric problems and other systems are described. Finally in Section 5 some final conclusions are drawn and future work is discussed.

2 The I2GATP Format

The Intergeo (I2G) file format is a specification based on the markup language XML designed to describe constructions created with a DGS. It is one of the main results of the intergeo project, an eContentplus European project dedicated to the sharing of interactive geometry constructions across boundaries. For more information about the project, visit the site `http://i2geo.net` and look into the documentation available there, as well as to [6, 7].

An intergeo file takes the form of a compress file package. The main file is `intergeo.xml`, which provides a textual description of the construction in three parts, the elements part describing a (static) initial instance of the configuration, the constraints part where the geometric relationships are expressed and the display part where the details regarding the rendering of the construction are placed. For more details on the file format see [14].

The I2GATP Format is thought as an extension of the I2G common format aimed to support conjectures and proofs produced by GATPs and to be used in repositories of geometric knowledge. Using the information already contained in the TGTP database as a template, the I2GATP format is a combination of four XSD files (see Figure 1): `information.xsd`; `intergeo.xsd`; `conjecture.xsd` and `proofInfo.xsd`.

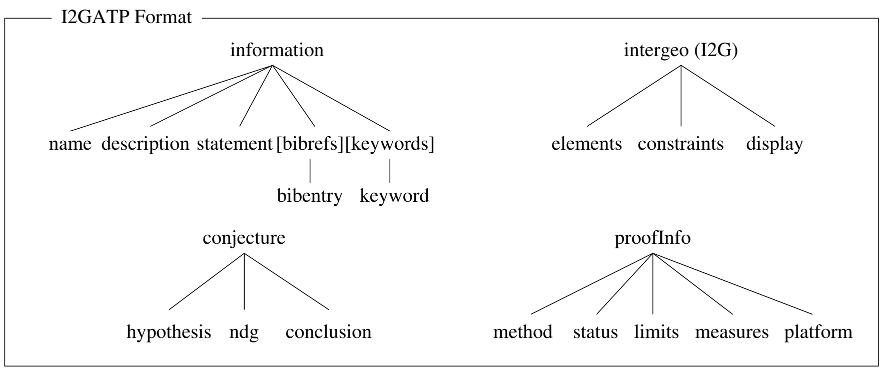

Fig. 1. Structure of the I2GATP File Format

Information The XSD file `information.xsd` (see Listing 1.1) contains all the information regarding the conjecture. This XSD format mirrors the information contained in the TGTP database.
For the `statement` tag the `MathML` XSD format is used. For the `bibrefs` tag, a list of bibliographic references, the file `bibtexml.xsd`[4] from the BibTeXml project is used. Conversion tools from LaTeX and BibTeX to the corresponding XML file are to be used, for example, *tex4ht*[5] and BibTeXml project converters respectively.

Listing 1.1. Fragment of `information.xsd`

```
<!-- information -->
 <xs:element name="information">
  <xs:complexType>
   <xs:all>
    <xs:element ref="conjecture_id"/>
    <xs:element ref="conjecture_name" minOccurs="0" maxOccurs="1"/>
    <xs:element ref="submission_date" minOccurs="0" maxOccurs="1"/>
    <xs:element ref="level" minOccurs="0" maxOccurs="1"/>
    <xs:element ref="description" minOccurs="0"  maxOccurs="1"/>
    <xs:element ref="statement" minOccurs="0" maxOccurs="1" />
    <xs:element ref="bibrefs" minOccurs="0"  maxOccurs="1"/>
    <xs:element ref="keywords" minOccurs="0" maxOccurs="1" />
   </xs:all>
  </xs:complexType>
 </xs:element>
```

[4] http://bibtexml.sourceforge.net/

[5] http://tug.org/applications/tex4ht/

Construction This is the `intergeo.xsd` file format from the Intergeo I2G format.

As we will explain below (see §4) one of the I2GATP project's goals is to provide filters from/to the DGS/GATP (at least the ones presented in TGTP) languages to/from this format.[6]

Conjecture The XSD file `conjecture.xsd` is a work-in-progress (see Listing 1.2). For now, it is a GCLC Area Method related format file, i.e. it is the translation of `conjecture` tag of the DTD file `geocons.dtd` from the XML suite incorporated in that GATP to the XSD format [2–4, 10].

Listing 1.2. Fragment of `conjecture.xsd`

```
<!-- conjecture -->
  <xs:element name="conjecture">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="prove" minOccurs="0"/>
      </xs:sequence>
      <xs:attribute ref="conjName" use="required"/>
    </xs:complexType>
  </xs:element>

 <xs:element name='prove'>
  <xs:complexType>
   <xs:sequence>
    <xs:element ref='xs:equality'/>
   </xs:sequence>
   <xs:attribute name='proof_limit' type='string' use='required'/>
   <xs:attribute name='proof_level' type='string' use='required'/>
  </xs:complexType>
 </xs:element>

 <xs:element name='equality'>
  <xs:complexType>
   <xs:sequence>
    <xs:element ref='xs:expression'/>
    <xs:element ref='xs:expression'/>
   </xs:sequence>
  </xs:complexType>
 </xs:element>

 <xs:element name='expression'>
  <xs:complexType>
   <xs:choice>
    <xs:element ref='xs:number'/>
    <xs:element ref='xs:constant'/>
    <xs:element ref='xs:sum'/>
    <xs:element ref='xs:mult'/>
    <xs:element ref='xs:fraction'/>
    <xs:element ref='xs:segment_ratio'/>
    <xs:element ref='xs:signed_area3'/>
    <xs:element ref='xs:pythagoras_difference3'/>
    <xs:element ref='xs:identical_points'/>
    <xs:element ref='xs:collinear'/>
   </xs:choice>
  </xs:complexType>
 </xs:element>
```

[6] Given the fact that we are more concern in the specification of a geometric construction and less in its rendering, the focus of the filters is in the construction's specification.

ProofInfo The XSD file `proofInfo.xsd` (see Listing 1.3) contains the meta-information regarding the proof generated by a given GATP on a given computing platform.
For the proofs themselves we are not considering a unique format. Given the fact that the GATPs using the area method or the full-angle method, the ones using coherent logic, or the ones using algebraic methods, all have very different proof formats, we do not see as possible to have some sort of confluence of those formats into a common format. Having that in mind the proof, if present, is included in the I2GATP container file, in the GATP own proof output format (see Section 3).

Listing 1.3. Fragment of `proofInfo.xsd`

```
<!--
 proofInfo - all the information about the proof, the gatp used
 (name & method) is mandatory all the other information is optional.
-->
<xs:element name="proof_info">
 <xs:complexType>
  <xs:sequence>
   <xs:element ref="gatp"/>
   <xs:element ref="status" minOccurs="0" maxOccurs="1"/>
   <xs:element ref="limits" minOccurs="0" maxOccurs="1"/>
   <xs:element ref="measures" minOccurs="0" maxOccurs="1"/>
   <xs:element ref="platform" minOccurs="0" maxOccurs="1"/>
  </xs:sequence>
 </xs:complexType>
</xs:element>
```

3 The I2GATP Container

The I2GATP container is an extension of the I2G container. In addition to the information in the I2G container, all the information regarding the geometric conjecture and all the proofs attempts are kept in the I2GATP container.

The public methods for the construction/deconstruction of the container are already implemented in the I2GATP library (see Section 4). The public methods allow deconstructing the container in its components and, in the opposite direction, building the container, having its components. The public method `addProofInfo` allow adding a new proof to an existing container (see Figure 2).

4 Implementation

The I2GATP library is an open source project,[7] implemented in C++, to support the I2GATP common format. The C++ classes are: `I2GATP`, to support the management of the I2GATP container; `Filter`*DGS/GATP*`toI2GATP` each implementing a filter for a given *DGS/GATP*; `TGTPtoI2GATP` containing the methods to deal with the integration of the I2GATP format in the TGTP repository.

[7] `http://itogatplibrary.sourceforge.net`

information/	mandatory
information/information.xml	optional
construction/	mandatory
(. . .)	
conjecture/	mandatory
conjecture/conjecture.xml	optional
proofs/	mandatory
proofs/proof<GATP><Version><Method>/	optional
proofs/proof<GATP><Version><Method>/proofInfo.xml	optional
proofs/proof<GATP><Version><Method>/proof.xml	optional
proofs/proof<GATP><Version><Method>/(. . .)	optional
metadata/	optional
(. . .)	
resources/	optional
resources/<image_files>	optional
resources/dgs<DGS><Version>/	optional
resources/dgs<DGS><Version>/<dgscode>	optional
resources/gatp<GATP><Version><Method>/	optional
resources/gatp<GATP><Version><Method>/<gatpcode>	optional
resources/(. . .)	optional
private/	optional
(. . .)	

Table 1. The I2GATP container

The `I2GATP` class aggregate all the filter classes. The `TGTPtoI2GATP` inherits from the `I2GATP` class, expanding this last one with the methods necessary to link TGTP and the I2GATP format (see Figure 2).

The I2GATP library depends on the following libraries: `libzip`,[8] a library to deal with `zip` archives, placing a list of files in a `zip` and in the opposite direction to open the `zip` archive; `boost_system` and `boost_filesystem`, libraries[9] to deal with files and directories. Given the fact that TGTP uses a MySQL database management system, the `TGTPtoI2GATP` class depends from the `mysqlcppconn` library[10] to make the connection with the MySQL database.

All the code is assembled in a dynamic library `libi2gatp.so`. Apart from the library, the program `callMakeI2GATP` is also created. It is used by the TGTP repository to build the I2GATP container for a given geometric conjecture and, in this way, to provide the TGTP's user all the info about that conjecture, packed in a unique file.

[8] `http://www.nih.at/libzip/`

[9] `http://www.boost.org/doc/libs/1_57_0/libs/filesystem/doc/index.htm`

[10] `http://dev.mysql.com/downloads/connector/cpp/`

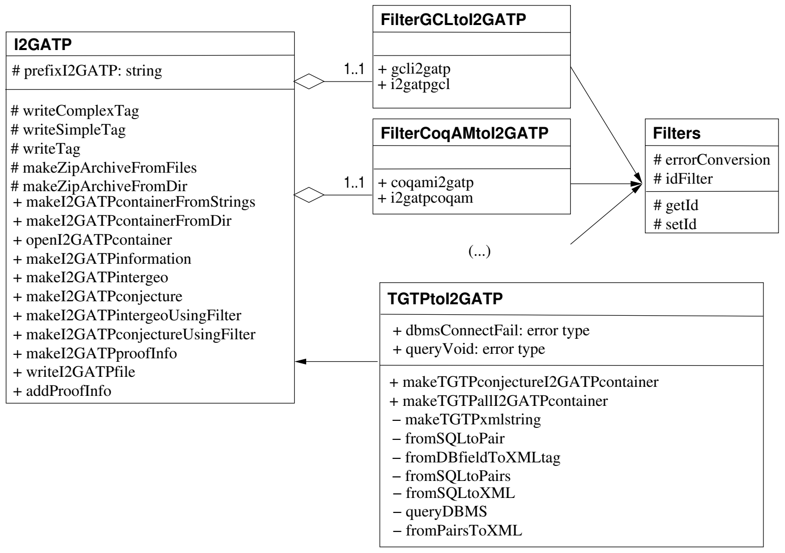

Fig. 2. UML Diagram for the I2GATP Library

4.1 I2GATP **Filters**

As a first step we aim to support the I2GATP format in TGTP [11], GeoThms [4] and WGL [13, 15, 16]. For that purpose we need filters from/to GCLC,[11] CoqAM[12] [2, 3, 8] and also GeoGebra.[13] The class `I2GATP` (see Figure 2) will use the filter's classes, e.g., `FilterGCLCtoI2GATP`, in order to convert from/to the DGS/GATP code to/from the I2GATP format. As said previously, we are more concern with the specification of the construction than with its rendering. Because of that the filters are built aiming to fully support the constructions' specification, disregarding, eventually, some of the rendering features.

The `FilterGCLtoI2GATP` and `FilterCoqAMtoI2GATP` are essentially wrappers for the filters built using lexical analyser and parser tools, e.g., `flex`[14] and `bison`.[15] The integration of other filters, eventually built with other tools, should follow the same guidelines, a function with one input string, with the DGS/GATP's code, and three output strings, corresponding to `intergeo.xml`, `conjecture.xml` and the report of the filter's execution (*log* string). The class `Filters` defines a set of error types common to all the filters and an identification number to identify, if needed, each of the filters.

[11] `http://poincare.matf.bg.ac.rs/~janicic/gclc/`
[12] GeoProof: `http://home.gna.org/geoproof/`
[13] `http://www.geogebra.org/`
[14] `http://flex.sourceforge.net/`
[15] `http://www.gnu.org/software/bison/`

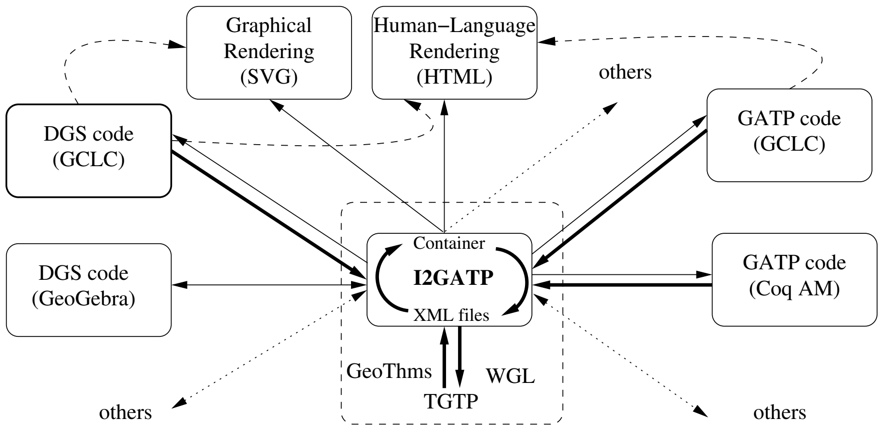

Fig. 3. Conversions From/To I2GATP To/From Geometric Tools

4.2 Integration Issues

The integration of this library into other programs is already being done (see Figure 3). The class `TGTPtoI2GATP` provides the public methods (see Figure 2) to build a container for a given problem or to do it for all the problems in TGTP.

The full integration will happen when the TGTP database will keep all the information using the I2GATP format. Then, using the filters and the methods already described, it will be possible to link the geometric information with the DGSs and GATPs.

The integration with GeoThms will be similar and will be done as soon as the integration with TGTP is completed.

A different set of goals are present when we consider the integration with a system like the Web Geometry Laboratory. This is a collaborative and adaptive blended-learning Web platform [13, 15, 16]. It is intended to allow verifying the geometric constructions deductively [5] and, because of that, the I2GATP format will be useful. In this case only the connection between the DGSs and the GATPs will be needed.

In figure 3 we can see the current status ("fat" arrows) and the planned work. The dashed arrows refer to a previous XML suite [10] which is integrated in GCLC.

5 Conclusions and Future Work

A format like I2GATP, allowing the connection between geometric tools, is important to open the geometric knowledge contained in repositories to all the community. It is important to be able to connect different tools in geometric environments, namely learning environments.

The main tasks ahead are:

– the (re)definition of the conjecture specification. We are planning to define a common format for this.

  Stojanović et. al. propose in "*A Vernacular for Coherent Logic*" [17], a simple yet expressive proof representation from which proofs for different proof assistants can easily be generated. This format is designed for conjectures in Coherent Logic and for forward chaining proofs (often used in common mathematical practice).

  Von Plato writes in *The axioms of constructive geometry* [9], that the general form of a geometric problem can be stated as: for a given data x of type A, find y such that condition $C(x, y)$ is fulfilled, with the type of y dependent of x, denoted $B(x)$.

  We are considering rewriting the XSD file in such a way that we can state the geometric problems in a more general form, following the ideas in [9, 17], but allowing specific XML tags, for different theories introduced by the different GATPs.

  Also the work of Chen [1] has to be considered in terms of exploting his geometric description language, and the transformation mechanisms provided, to allow complex changes in representations, e.g., as said in [1], to exchange from a predicate form to a constructive form.
– the definition of an application programming interface (API), fully documented, to support an easy integration of the I2GATP format in different platforms. This should be complemented by an Web interface to exemplify how to extend/apply the I2GATP library. We will use our experience with the TGTP integration to achieve this goal;
– the construction of filters to/from other geometric tools. Having set up the I2GATP library as a *sourceforge*[16] open source project we hope that other developers of geometric software can contribute writing their own filters and incorporating them in the library.

Having already done some work, much more is still in need to be done.

References

1. Chen, X.: Representation and automated transformation of geometric statements. Journal of Systems Science and Complexity 27(2), 382–412. Springer (2014)
2. Janičić, P.: GCLC — A tool for constructive euclidean geometry and more than that. In: Iglesias, A., Takayama, N. (eds.) Mathematical Software - ICMS 2006, Lecture Notes in Computer Science, vol. 4151, pp. 58–73. Springer (2006)
3. Janičić, P., Narboux, J., Quaresma, P.: The Area Method: a recapitulation. Journal of Automated Reasoning 48(4). Springer (2012)
4. Janičić, P., Quaresma, P.: System description: GCLCprover + GeoThms. In: Furbach, U., Shankar, N. (eds.) Automated Reasoning, Lecture Notes in Computer Science, vol. 4130, pp. 145–150. Springer (2006)

[16] `http://sourceforge.net/about`

5. Janičić, P., Quaresma, P.: Automatic verification of regular constructions in dynamic geometry systems. In: Botana, F., Recio, T. (eds.) Automated Deduction in Geometry, Lecture Notes in Computer Science, vol. 4869, pp. 39–51. Springer (2007)
6. Kortenkamp, U., Blessing, A.M., Dohrmann, C., Kreis, Y., Libbrecht, P., Mercat, C.: Interoperable Interactive Geometry for Europe – First technological and educational results and future challenges of the Intergeo Project. In: CERME 6 (2006)
7. Kortenkamp, U., Dohrmann, C., Kreis, Y., Dording, C., Libbrecht, P., Mercat, C.: Using the Intergeo platform for teaching and research. In: Proceedings of the 9th International Conference on Technology in Mathematics Teaching (ICTMT-9) (2009)
8. Narboux, J.: Formalization of the area method. Coq user contribution (2009), `http://dpt-info.u-strasbg.fr/~narboux/area\_method.html`
9. von Plato, J.: The axioms of constructive geometry. In: Annals of Pure and Applied Logic. vol. 76, pp. 169–200. Elsevier (1995)
10. Quaresma, P., Janičić, P., Tomašević, J., Vujošević-Janičić, M., Tošić, D.: Communicating Mathematics in The Digital Era, chap. XML-Bases Format for Descriptions of Geometric Constructions and Proofs, pp. 183–197. A. K. Peters, Ltd. (2008)
11. Quaresma, P.: Thousands of Geometric problems for geometric Theorem Provers (TGTP). In: Schreck, P., Narboux, J., Richter-Gebert, J. (eds.) Automated Deduction in Geometry, Lecture Notes in Computer Science, vol. 6877, pp. 169–181. Springer (2011)
12. Quaresma, P.: An XML-format for conjectures in geometry. In: Davenport, J., Jeuring, J., Lange, C., Libbrecht, P. (eds.) 24$^{\text{th}}$ OpenMath Workshop, 7$^{\text{th}}$ Workshop on Mathematical User Interfaces (MathUI), and Intelligent Computer Mathematics Work in Progress. pp. 54–65. No. 921 in CEUR Workshop Proceedings, Aachen (2012)
13. Quaresma, P., Santos, V., Bouallegue, S.: The Web Geometry Laboratory project. In: CICM 2013, Lecture Notes in Computer Science, vol. 7961, pp. 364–368. Springer (2013)
14. Santiago, E., Hendriks, M., Kreis, Y., Kortenkamp, U., Marquès, D.: i2g Common File Format Final Version. Tech. Rep. D3.10, The Intergeo Consortium (2010), `http://i2geo.net/xwiki/bin/view/I2GFormat/`
15. Santos, V., Quaresma, P.: Integrating DGSs and GATPs in an adaptative and collaborative blended-learning Web-environment. In: First Workshop on CTP Components for Educational Software (THedu'11). EPTCS, vol. 79, p. 111–123 (2012)
16. Santos, V., Quaresma, P.: Collaborative aspects of the WGL project. Electronic Journal of Mathematics & Technology 7(6) (2013), Mathematics and Technology, LLC
17. Stojanović, S., Narboux, J., Bezem, M., Janičić, P.: A vernacular for coherent logic. In: Watt, S., Davenport, J., Sexton, A., Sojka, P., Urban, J. (eds.) Intelligent Computer Mathematics, Lecture Notes in Computer Science, vol. 8543, pp. 388–403. Springer (2014)