Cognitive Technologies

Georg Rehm *Editor*

European Language Grid

A Language Technology Platform for Multilingual Europe

EUROPEAN LANGUAGE GRID

OPEN ACCESS

Springer

Cognitive Technologies

Georg Rehm
Editor

European Language Grid

A Language Technology Platform
for Multilingual Europe

Springer

EUROPEAN
LANGUAGE
GRID

Editor
Georg Rehm
Deutsches Forschungszentrum
für Künstliche Intelligenz GmbH (DFKI)
Berlin, Germany

The European Language Grid has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no. 825627.

ISSN 1611-2482 ISSN 2197-6635 (electronic)
Cognitive Technologies
ISBN 978-3-031-17257-1 ISBN 978-3-031-17258-8 (eBook)
https://doi.org/10.1007/978-3-031-17258-8

This Springer imprint is published by the registered company Springer Nature Switzerland AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Foreword

I was proud to have the opportunity to present my report on *Language Equality in the Digital Age* to the European Parliament in 2018 and even prouder to see the overwhelming support it received. It was one of my final achievements as a Member of the European Parliament and I am delighted that it contributed to the groundbreaking work being done on the European Language Grid project. Despite it not being a legislative report, the level of cross-party support it received meant its recommendations could not be ignored.

When I first proposed a report on language equality in the digital age to the European Parliament's Culture and Education Committee it provoked a great deal of interest, as it did in the Industry Committee. This was due to the clear language inequality in Europe but also to the huge opportunities it presented for the digital industries. As both committees laid claim to the report, there was some debate before it was resolved that it would be a Culture Committee report with a written opinion from the Industry Committee. The latter's participation strengthened the report and its impact. It widened the scope to emphasise the importance of the role of private companies alongside public bodies and of facilitating cross border trade in the Digital Single Market.

It was clear from the early days that the European Commission was keen to support the report and take the proposals forward. As a spokesperson for the Commissioner stated in a conference I organised in parliament in September 2018, "You are never so wealthy as when you can speak in your own language". The European Language Equality project is currently developing a roadmap to achieve language equality by 2030, which will be presented to the European Institutions later this year.

Minority languages in particular have most to lose but also most to gain from the digital age, given the right support. Cultural and linguistic diversity depends largely on the technological resources available to all languages.

It was a report by the EU Panel for the Future of Science and Technology, STOA, that sparked the idea of a parliamentary report. STOA highlighted the social and economic consequences of language barriers and the widening of the technological gap. As someone who had long campaigned for equal status for the Welsh language, I was inspired by the potential that a major EU project could offer.

Even though the dominance of a few well-resourced languages in the digital world was obvious, the impact of this on other languages had not been adequately explored. When the discussion began, the interest grew. The increase in technology presented new threats and new opportunities. This was an issue which literally affected everyone, and most notably children growing up in this digital world. The role of education is crucial in teaching and understanding language technologies but also in raising awareness of career opportunities in this industry across Europe.

The European Union itself, of course, could could play a major role. The institutional framework for the provision of language technology could be improved considerably. I believed that this was such a crucial issue that it deserved the specific allocation of the portfolio to a European Commissioner. This did not materialise in the appointment of a new Commission following the European elections in 2019, but I believe the proposal should be maintained and could be adopted in future.

The strong support given to my report by the European Parliament was an indication of support for the exciting language equality work that is taking place now. The report proposed a dedicated funding programme for research, development and innovation in language technologies with the aim of closing the gap between European languages.

This suggestion was a direct result of seeing the existing research being done in many countries. Identifying the problem went hand in hand with discovering that there were many individuals and organisations already addressing it and working to overcome it. They had the information and expertise but needed far more support and a higher profile. It was clear that the EU could become a trailblazer in research on digital language technology, given the political will.

As a politician, the rights of minority languages like my own, Welsh, were at the heart of my work for justice and equality. For me, language was not merely a means of communication but central to our culture and identity. The EU claims equality in diversity but when it came to language equality it fell far short. So in my role as a Member of the European Parliament I saw an opportunity to help correct this. I could play my role in parliament but to ensure the report was effective in achieving its aim it needed the input of the experts, the practitioners and the pioneers in this field of work to ensure that it was accurate and informed.

I never fail to be inspired by their work and their dedication and I repeat my thanks to all those who contributed to the success of the *Language Equality in the Digital Age* report and to the remarkable European Language Grid project which established the primary platform for “language technologies *for* Europe built *in* Europe”.

Rhondda Valley, April 2022 *Jill Evans*

Preface

The origins of this book date back to 2012. Back then, under the umbrella of the EU Network of Excellence META-NET, we prepared the recommendations and priority research themes specified in the first Strategic Research Agenda (SRA) for the European Language Technology (LT) field in a complex, community-driven process.[1] While the European LT community is quite extensive, with hundreds of commercial and academic organisations working on a large and heterogeneous set of technologies, it is also extremely fragmented with many community members operating only in narrow niches and limited regions, on very specific topics and quite often only taking into account one or two regionally confined languages. Through the META-NET SRA process, we have been able to identify the community's need for a joint technology platform that brings the European LT community together, that fosters collaboration and synergies, that acts as a marketplace and deployment platform, that functions like the "yellow pages" of the European LT community and through which essentially all European resources, corpora, datasets and grammars as well as tools, services and source code can be discovered and actually used, straight from the platform itself. Back in 2013, in the published META-NET SRA, we called this concept the *European Service Platform for Language Technologies*. The SRA document only contained a rather coarse-grained description of this ambitious technology vision, which has been demanded, for a number of different reasons, by an overwhelming majority of the members of the LT community.

Later on, in the three Strategic Research and Innovation Agendas prepared under the umbrella of the EU project CRACKER (Cracking the Language Barrier; 2015-2017), we refined the notion of the European LT Service Platform and we extended the possible use cases and a large number of LT-driven applications, primarily focusing the multilingual digital single market. Further boosted by the scientific breakthroughs produced in the area of Artificial Intelligence, Machine Learning and Deep Learning, early on applied to LT applications such as Machine Translation, not only the topic of Language Technology but also the vision of a joint European Language Technology Platform became more and more relevant. The topic was mentioned in

[1] http://www.meta-net.eu/sra

a prominent way in the STOA study *Language equality in the digital age*[2], commissioned by the European Parliament and also in a European Parliament resolution[3] with the same title, adopted by the European Parliament in a landslide vote in 2018 (cf. Jill Evans' foreword).

Roughly at the same time, in late 2017, we started preparing a project proposal for the Horizon 2020 ICT call, topic ICT-29 a), *European Language Grid*, which fortunately reflected the vast interest within the community in such a platform. After various unsuccessful attempts at coming up with a good title for the proposal, we decided to use the title of the actual call because it fit perfectly. Having passed the evaluation with a positive result, the project started in January 2019. We had an enthusiastic kick-off meeting, exciting hackathons and developed the first prototype of the platform in a fast and agile way. It was first presented to the public at META-FORUM 2019, which took place in Brussels in October that year.

Only a few weeks later, the global SARS-CoV-2 pandemic hit. The whole world was affected and so was our project plan. We were unable to have face-to-face project meetings or additional hackathons, we were unable to organise any on-site workshops with our 32 ELG National Competence Centres as part of the "ELG European Roadshow". All meetings, including our annual META-FORUM conferences in 2020 and 2021, had to go virtual, which was new to us at first and quickly became the new normal. Recently, in early June 2022, we had our last META-FORUM conference under the umbrella of the ELG EU project. META-FORUM 2022 went back, at least partially, to the *old* normal with approx. 100 participants in the conference centre in Brussels and hundreds more participating remotely.

It was nothing but a pleasure to act as Coordinator of the European Language Grid project and to work together with such a strong and dedicated team. Our original plan in this Innovation Action was already quite ambitious yet we managed to exceed our joint expectations in terms of the technology platform and its features, in terms of the services and resources developed, collected and ingested into the platform, in terms of the acceptance and feedback by the community and also in terms of the various collaborations we conducted with other projects. Many of the features envisioned for the *European Service Platform for Language Technologies* in 2013 are in fact now finally available in the *European Language Grid*, which is, by a large margin, the biggest all-purpose Language Technology platform on the planet covering the whole breadth and technology spectrum of the field.

All of the activities and results produced by the nine partners of the ELG consortium during the project's runtime are described in this book in detail. I would like to thank all consortium partners and team members for their extremely hard and dedicated work towards our common goal of developing and establishing the ELG platform, community and marketplace. In addition, I would like to thank the 15 selected pilot projects for their innovative proposals and the more than 200 organisations who applied for funding through one of our pilot projects. Thanks are also due to the projects ELG collaborated with, especially, in 2021/2022, the European

[2] https://www.europarl.europa.eu/stoa/en/document/EPRS_STU(2017)598621

[3] https://www.europarl.europa.eu/doceo/document/TA-8-2018-0332_EN.html

Language Equality project, the results of which will also be documented in the form of a book in the same series, but also others such as Bergamot, COMPRISE, ELITR, EMBEDDIA, Gourmet, Prêt-à-LLOD, AI4EU, HumanE AI Net, VISION, TAILOR, WeVerify, NTEU, Microservices at your Service, MAPA, QURATOR, PANQURA, SPEAKER and many others.

This book is the definitive documentation of the EU project European Language Grid.[4] I would like to thank all colleagues from the ELG consortium and also from the ELG pilot projects wholeheartedly for the chapters they contributed, without which this book would not have been possible.

While this book can only cover the results achieved during the project's runtime (January 2019 until June 2022), the ELG initiative will continue. In the second half of 2022 we will establish a legal entity that will take over maintenance and operation of the platform. We hope that ELG will serve its many purposes and, among others, address the stark community fragmentation and contribute to digital language equality in Europe, functioning indeed as one joint umbrella platform for the whole European LT community. Furthermore, while none of these can be considered a direct follow-up just yet, in a few projects (including OpenGPT-X, NFDI4DataScience and AI as well as the EU projects DataBri-X and SciLake) we will have the opportunity to continue our work with and on the ELG platform.

Berlin, July 2022 *Georg Rehm*

Acknowledgements The European Language Grid EU project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no. 825627.

[4] https://european-language-grid.readthedocs.io provides more details with regard to technical aspects of the ELG platform. The online documentation is actively maintained and kept up to date.

Contents

List of Contributors

European Language Grid EU Project (Parts I, II and III)

Victoria Arranz
ELDA, France, arranz@elda.org

Gerhard Backfried
HENSOLDT Analytics GmbH, Austria, gerhard.backfried@hensoldt.net

Cristian Berrìo Aroca
Expert AI, Spain, cberrio@expert.ai

Kalina Bontcheva
University of Sheffield, UK, k.bontcheva@sheffield.ac.uk

Rémi Calizzano
Deutsches Forschungszentrum für Künstliche Intelligenz GmbH (DFKI), Germany, remi.calizzano@dfki.de

Khalid Choukri
ELDA, France, choukri@elda.org

Miltos Deligiannis
Institute for Language and Speech Processing, R. C. "Athena", Greece, mdel@athenarc.gr

Dimitris Galanis
Institute for Language and Speech Processing, R. C. "Athena", Greece, galanisd@athenarc.gr

Andres Garcia Silva
Expert AI, Spain, agarcia@expert.ai

Ulrich Germann
University of Edinburgh, UK, ulrich.germann@ed.ac.uk

Maria Giagkou
Institute for Language and Speech Processing, R. C. "Athena", Greece, mgiagkou@athenarc.gr

Katerina Gkirtzou
Institute for Language and Speech Processing, R. C. "Athena", Greece, katerina.gkirtzou@athenarc.gr

Dimitris Gkoumas
Institute for Language and Speech Processing, R. C. "Athena", Greece, dgkoumas@athenarc.gr

Jose Manuel Gómez-Pérez
Expert AI, Spain, jmgomez@expert.ai

Annika Grützner-Zahn
Deutsches Forschungszentrum für Künstliche Intelligenz GmbH (DFKI), Germany, annika.gruetzner-zahn@dfki.de

Jan Hajič
Charles University, Czech Republic, hajic@ufal.mff.cuni.cz

Jana Hamrlová
Charles University, Czech Republic, hamrlova@ufal.mff.cuni.cz

Stefanie Hegele
Deutsches Forschungszentrum für Künstliche Intelligenz GmbH (DFKI), Germany, stefanie.hegele@dfki.de

Miroslav Jánošík
HENSOLDT Analytics GmbH, Austria, miroslav.janosik@hensoldt-analytics.com

Lukáš Kačena
Charles University, Czech Republic, kacena@ufal.mff.cuni.cz

Florian Kintzel
Deutsches Forschungszentrum für Künstliche Intelligenz GmbH (DFKI), Germany, florian.kintzel@dfki.de

Athanasia Kolovou
Institute for Language and Speech Processing, R. C. "Athena", Greece, akolovou@athenarc.gr

Ondřej Košarko
Charles University, Czech Republic, kosarko@ufal.mff.cuni.cz

Jens-Peter Kückens
Deutsches Forschungszentrum für Künstliche Intelligenz GmbH (DFKI), Germany, jens_peter.kueckens@dfki.de

Penny Labropoulou
Institute for Language and Speech Processing, R. C. "Athena", Greece, penny@athenarc.gr

Andis Lagzdiņš
Tilde, Latvia, andis.lagzdins@tilde.lv

Valérie Mapelli
ELDA, France, mapelli@elda.org

Katrin Marheinecke
Deutsches Forschungszentrum für Künstliche Intelligenz GmbH (DFKI), Germany, katrin.marheinecke@dfki.de

Stelios Piperidis
Institute for Language and Speech Processing, R. C. "Athena", Greece, spip@athenarc.gr

Katja Prinz
HENSOLDT Analytics GmbH, Austria, katja.prinz@hensoldt.net

Georg Rehm
Deutsches Forschungszentrum für Künstliche Intelligenz GmbH (DFKI), Germany, georg.rehm@dfki.de

Mickaël Rigault
ELDA, France, mickael@elda.org

Ian Roberts
University of Sheffield, UK, i.roberts@sheffield.ac.uk

Milan Straka
Charles University, Czech Republic, straka@ufal.mff.cuni.cz

Andrejs Vasiļjevs
Tilde, Latvia, andrejs@tilde.lv

Leon Voukoutis
Institute for Language and Speech Processing, R. C. "Athena", Greece, leon.voukoutis@athenarc.gr

European Language Grid FSTP Pilot Projects (Part IV)

Begoña Altuna
Fondazione Bruno Kessler, Italy, HiTZ Centre, University of the Basque Country, Spain, begona.altuna@ehu.eus

Sebastian Andersson
Lingsoft, Finland, sebastian.andersson@lingsoft.fi

Arkaitz Anza
Skura Mobile, Spain, arkaitz@skuramobile.com

Mikko Aulamo
University of Helsinki, Finland, mikko.aulamo@helsinki.fi

Valerio Basile
University of Turin, Italy, valerio.basile@unito.it

Andrea Bolioli
CELI, Italy, andrea.bolioli@h-farm.com

Alessio Bosca
CELI, Italy, alessio.bosca@h-farm.com

Cristina Bosco
University of Turin, Italy, cristina.bosco@unito.it

Mark Breuker
EDIA b. v., The Netherlands, mark@edia.nl

Cesare Campagnano
Sapienza University of Rome, Italy, campagnano@di.uniroma1.it

Li-Hsin Chang
University of Turku, Finland, lhchan@utu.fi

Simone Conia
Sapienza University of Rome, Italy, conia@di.uniroma1.it

Ander Corral
Elhuyar Fundazioa, Spain, a.corral@elhuyar.eus

Michael Fell
University of Turin, Italy, michael.fell@unito.it

Filip Ginter
University of Turku, Finland, figint@utu.fi

Dagmar Gromann
University of Vienna, Austria, dagmar.gromann@univie.ac.at

Sam Hardwick
University of Helsinki, Finland, sam.hardwick@helsinki.fi

Barbara Heinisch
University of Vienna, Austria, barbara.heinisch@univie.ac.at

Gregor Jarisch
Labs.ai, Austria, gregor@labs.ai

Pavel Jedlička
University of West Bohemia, Czech Republic, jedlicka@ntis.zcu.cz

Beñat Jimenez
Talaios Koop., Spain, jimakker@talaios.coop

Jenna Kanerva
University of Turku, Finland, jmnybl@utu.fi

Jemina Kilpeläinen
University of Turku, Finland, jemina.j.kilpelainen@utu.fi

Svetla Koeva
Institute for Bulgarian Language, Bulgarian Academy of Sciences, Bulgaria, svetla@dcl.bas.bg

Zdeněk Krňoul
University of West Bohemia, Czech Republic, zdkrnoul@ntis.zcu.cz

Hanna-Mari Kupari
University of Turku, Finland, hmknie@utu.fi

Christian Lang
University of Vienna, Austria, christian.lang@univie.ac.at

Alberto Lavelli
Fondazione Bruno Kessler, Italy, lavelli@fbk.eu

Igor Leturia
Elhuyar Fundazioa, Spain, i.leturia@elhuyar.eus

Helmut Ludwar
Sign Time GmbH, Austria, helmut.ludwar@signtime.media

Bernardo Magnini
Fondazione Bruno Kessler, Italy, magnini@fbk.eu

Jaione Martinez
Skura Mobile, Spain, jaione@skuramobile.com

Anne-Lyse Minard
Université d'Orléans, France, anne-lyse.minard@univ-orleans.fr

Luděk Müller
University of West Bohemia, Czech Republic, muller@kky.zcu.cz

Roberto Navigli
Sapienza University of Rome, Italy, navigli@diag.uniroma1.it

Tommi Nieminen
University of Helsinki, Finland, tommi.nieminen@helsinki.fi

Riccardo Orlando
Sapienza University of Rome, Italy, orlando@diag.uniroma1.it

Viviana Patti
University of Turin, Italy, viviana.patti@unito.it

Aurora Piirto
University of Turku, Finland, aurora.e.piirto@utu.fi

Silvia Portela
Skura Mobile, Spain, silvia@skuramobile.com

Jenna Saarni
University of Turku, Finland, jensaay@utu.fi

Xabier Sarasola
Elhuyar Fundazioa, Spain, x.sarasola@elhuyar.eus

Julia Schuster
Sign Time GmbH, Austria, julia.schuster@signtime.media

Maija Sevón
University of Turku, Finland, maija.suonpaa@gmail.com

Valtteri Skantsi
University of Turku, Finland, valtteri.skantsi@oulu.fi

Manuela Speranza
Fondazione Bruno Kessler, Italy, manspera@fbk.eu

Michael Stormbom
Lingsoft, Finland, michael.stormbom@lingsoft.fi

Otto Tarkka
University of Turku, Finland, ohitar@utu.fi

Steffen Thoma
FZI Research Center for Information Technology, Germany, thoma@fzi.de

Jörg Tiedemann
University of Helsinki, Finland, jorg.tiedemann@helsinki.fi

Rossella Varvara
University of Turin, Italy, rossella.varvara@unito.it

Alena Vasilevich
Coreon GmbH, Germany, alena@coreon.com

Lennart Wachowiak
University of Vienna, Austria, lennart.wachowiak@univie.ac.at

Franz Weber
Labs.ai, Austria, franz@labs.ai

Michael Wetzel
Coreon GmbH, Germany, michael@coreon.com

Patrick Wiener
FZI Research Center for Information Technology, Germany, wiener@fzi.de

Roberto Zanoli
Fondazione Bruno Kessler, Italy, zanoli@fbk.eu

Miloš Železný
University of West Bohemia, Czech Republic, zelezny@ntis.zcu.cz

Acronyms

AI	Artificial Intelligence
AMR	Abstract Meaning Representation
API	Application Programming Interface
ASL	American Sign Language
ASR	Automatic Speech Recognition
ATE	Automated Term Extraction
BMC	Business Model Canvas
CAS	Common Analysis System
CAT	Computer-assisted Translation
CC	Creative Commons
CD	Continuous Deployment
CEF	Connecting Europe Facility
CEFR	Common European Framework of Reference
CI	Continuous Integration
CLAIRE	Confederation of Laboratories for AI Research in Europa
CLARIN	Common Language Resources and Technology Infrastructure
CLI	Command-Line Interface
CMDI	Component Metadata Infrastructure
CMS	Content Management System
COAR	Controlled Vocabularies for Repositories
COMPRISE	Cost-effective, Multilingual, Privacy-driven, Voice-enabled Services
CPU	Central Processing Unit
CRACKER	Cracking the Language Barrier
CSE	Czech Sign Language
CSS	Cascading Style Sheets
CURLICAT	Curated Multilingual Language Resources for CEF AT
DC	Data Controller
DC	Dublin Core
DCAT	Data Catalog Vocabulary
DMP	Data Management Plan

DOI	Digital Object Identifier
DSDE	Development of Slovene in a Digital Environment
EEA	European Economic Area
EEIG	European Economic Interest Grouping
EFNIL	European Federation of National Institutions for Language
ELE	European Language Equality
ELG	European Language Grid
ELG R1	European Language Grid Release 1
ELG R2	European Language Grid Release 2
ELG R3	European Language Grid Release 3
ELITR	European Live Translator
ELRA	European Language Resource Association
ELRC	European Language Resource Coordination
ELT	European Language Technology
EMBEDDIA	Cross-lingual Embeddings for Less-Represented Languages in European News Media
EOSC	European Open Science Cloud
EUCPT	EU Council Presidency Translator
FAIR	Findable, Accessible, Interoperable, Reusable
FSTP	Financial Support to Third Parties
GATE	General Architecture for Text Engineering
GDPR	General Data Protection Regulation
GPU	Graphics Processing Unit
GUI	Graphical User Interface
HF	Hugging Face
HLT	Human Language Technology
HMM	Hidden Markov Models
HPA	Horizontal Pod Autoscaler
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
IAA	Inter-Annotator Agreement
ICT	Information and Communication Technology
IE	Information Extraction
IRI	Internationalised Resource Identifier
JSON	JavaScript Object Notation
JVM	Java Virtual Machine
KG	Knowledge Graph
KPI	Key Performance Indicator
KWS	Keyword-Spotting
LCR	Lexical or Conceptual Resource
LLOD	Linguistic Linked Open Data
LOD	Linked Open Data
LR	Language Resource
LRT	Language Resources and Language Technologies
LSDISCO	Lingsoft Solutions as Distributable Containers

LT	Language Technology
MAPA	Multilingual Anonymisation for Public Administrations
MARCELL	Multilingual Resources for CEF.AT in the Legal Domain
META	Multilingual Europe Technology Alliance
META-NET	Network of Excellence forging the Multilingual Europe Technology Alliance
MIME	Multipurpose Internet Mail Extensions
MT	Machine Translation
MVP	Minimum Viable Product
NAP	National Anchor Point
NCC	National Competence Centre
NCP	National Contact Point
NER	Named Entity Recognition
NLP	Natural Language Processing
NLU	Natural Language Understanding
NMT	Neural Machine Translation
NTEU	Neural Translation for the European Union
OA	Open Access
OAI-PMH	Open Archives Initiative Protocol for Metadata Harvesting
OCR	Optical Character Recognition
OLAC	Open Language Archives Community
OWL	Web Ontology Language
PB	Pilot Board
PID	Persistent Identifier
POS	Part of Speech
PRINCIPLE	Providing Resources in Irish, Norwegian, Croatian and Icelandic for Purposes of Language Engineering
PROVENANCE	Providing Verification Assistance for New Content
QURATOR	QURATOR – Curation Technologies
RDF	Resource Description Framework
REST	Representational State Transfer
SDK	Software Development Kit
SEO	Search Engine Optimisation
SKOS	Simple Knowledge Organisation System
SL	Sign Language
SME	Small and Medium Size Enterprises
SOA	Service-Oriented Architecture
SPDX	Software Package Data Exchange
SQL	Structured Query Language
SR	Subword Regularisation
SRL	Semantic Role Labeling
TBX	Termbase Exchange
TC	Text Classification
TCS	Terminological Concept System
TMX	Translation Memory Exchange

TTS	Text To Speech Synthesis
UI	User Interface
UML	Unified Modeling Language
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
USP	Unique Selling Proposition
WFST	Weighted Finite State Transducer
WMT	Workshop/Conference on Machine Translation
WSD	Word Sense Disambiguation
XGAPP	XML GATE Application
XMI	XML Metadata Interchange
XML	Extensible Markup Language
XSD	XML Schema Definition
YAML	YAML Ain’t Markup Language

Chapter 1
European Language Grid: Introduction

Georg Rehm

Abstract Europe is a multilingual society with 24 European Union Member State languages and dozens of additional languages including regional and minority languages as well as languages spoken by immigrants, trade partners and tourists. While languages are an essential part of our cultural heritage, language barriers continue to be unbreachable in many situations. The only option to enable and to benefit from multilingualism is through Language Technologies (LTs) including Natural Language Processing (NLP), Natural Language Understanding (NLU) and Speech Technologies. The commercial European LT landscape is dominated by hundreds of SMEs that develop many different kinds of LTs. While the industrial and also the academic European LT community is world-class, it is also massively fragmented. This chapter is an introduction to the present volume, which describes the European Language Grid (ELG) cloud platform, initiative and EU project. The ELG system is targeted to evolve into the primary platform and marketplace for LT in Europe by providing one umbrella platform for the entire European LT community, including research and industry, enabling all stakeholders to showcase, share and distribute their services, tools, products, datasets and other resources. At the time of writing, the ELG platform provides access to more than 13,000 commercial and non-commercial language resources and technologies covering all official EU languages and many national, co-official, regional and minority languages.

1 Overview and Context

Europe is a multilingual society with 24 EU Member State languages and dozens of additional languages including regional and minority languages as well as languages spoken by immigrants, trade partners and tourists. While languages are an important part of our cultural heritage, language barriers continue to be unbreachable in many situations. The only option to enable and to benefit from multilingualism is through Language Technologies (LTs) including Natural Language Processing (NLP), Nat-

Georg Rehm
Deutsches Forschungszentrum für Künstliche Intelligenz GmbH, Germany, georg.rehm@dfki.de

G. Rehm (ed.), *European Language Grid*, Cognitive Technologies,
https://doi.org/10.1007/978-3-031-17258-8_1

ural Language Understanding (NLU), and Speech Technologies. The commercial European LT landscape is dominated by hundreds of SMEs and a few larger enterprises (Rehm et al. 2020b). While the European LT community is world class, it is also very fragmented, significantly holding back its impact (Vasiljevs et al. 2019).

This book is the definitive documentation[1] of the EU project *European Language Grid*, which has developed the ELG cloud platform (Figure 1), available online at:

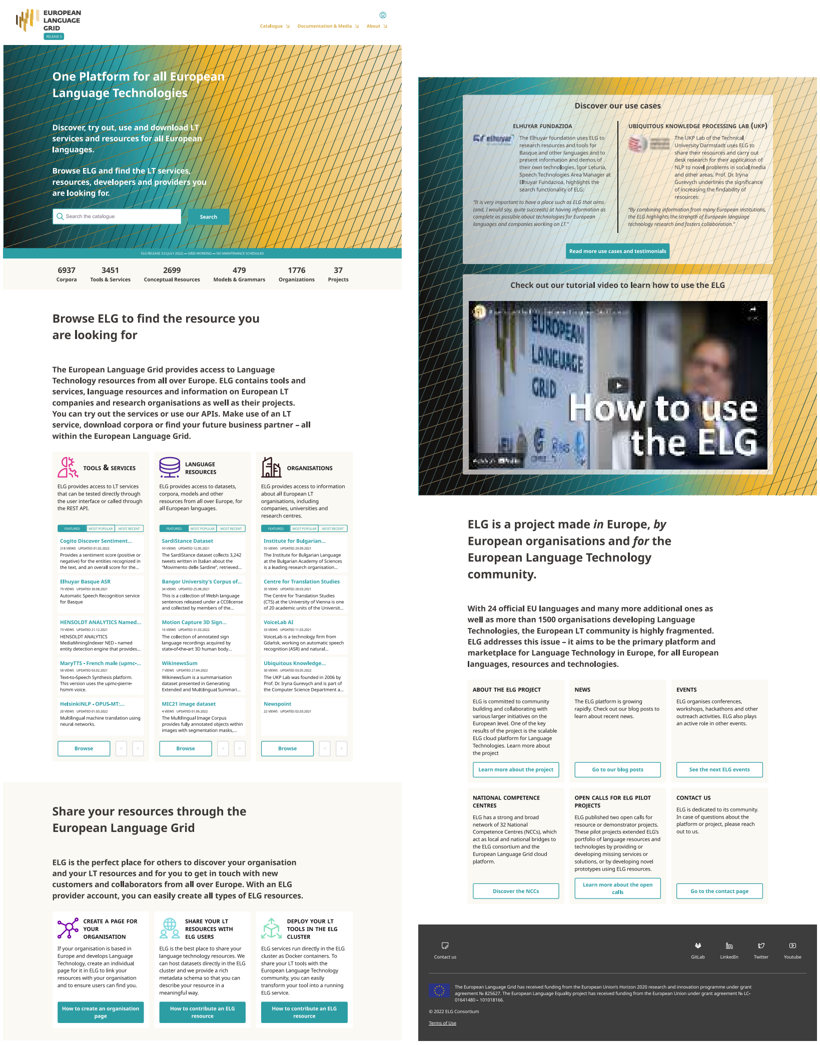
https://www.european-language-grid.eu

The European Language Grid is targeted to evolve into the primary platform for Language Technology in Europe. We provide one umbrella platform for all LTs and LRs developed by the whole European LT landscape, including research and industry, addressing a major gap, i. e., the lack of a common LT platform, that has been repeatedly raised by the whole community for many years (Rehm and Uszkoreit 2013; Rehm et al. 2016; STOA 2018; Rehm and Hegele 2018; European Parliament 2018). The ELG platform is also meant to be a virtual home and marketplace for all products, services and organisations active in this space in Europe, significantly boosting the EU Digital Single Market by helping to make it multilingual. ELG is an initiative *from* the European LT community *for* the European LT community. It provides one platform that can be used by all stakeholders to showcase, share and distribute their products, services, tools, datasets, corpora and other relevant resources. At the time of writing, the ELG platform enables access to more than 13,000 commercial and non-commercial language resources and technologies for all official EU languages and many national, co-official, regional and minority languages.

The European LT community had been demanding a dedicated LT platform for years – the ELG cloud platform fills this gap. The ambition of the ELG project and initiative is to unite a strong and extensive network of European experts and concentate on *commercial* as well as *non-commercial LTs*, both *functional* (analysis, processing and generation for written and spoken language) and *non-functional* (datasets, corpora, lexicons, models etc.). A related goal is to establish the ELG as a marketplace for the fragmented European LT landscape (Vasiljevs et al. 2019; Rehm et al. 2020b) to connect demand and supply, strengthening Europe's position in this field. The ELG platform enables the whole European LT community to upload their services and datasets, to deploy them, connect with, and make use of those resources made available by others (taking into account IPR and licences, as soon as the ELG legal entity is in place, including payment and billing options, especially with regard to commercial services and resources).

ELG is also meant to support *digital language equality* in Europe (STOA 2018; European Parliament 2018), i. e., bringing about a situation in which *all* languages are supported through technologies equally well. Currently, there is still an extreme imbalance, characterised by a stark predominance of LRTs for English, while almost all other languages are only marginally supported (Gaspari et al. 2022; Grützner-Zahn and Rehm 2022). In fact, many of these languages are in danger of digital

[1] The ELG cloud platform is actively being used, i. e., new services, tools and resources are made available on or through ELG on a daily basis. The data, numbers and statistics presented in this book regarding the use of ELG reflect the respective time of writing.



One Platform for all European Language Technologies

Discover, try out, use and download LT services and resources for all European languages.

Browse ELG and find the LT services, resources, developers and providers you are looking for.

ELG RELEASE 3.0 (JULY 2022) — GRID WORKING — NO MAINTENANCE SCHEDULED

6937	3451	2699	479	1776	37
Corpora	Tools & Services	Conceptual Resources	Models & Grammars	Organizations	Projects

Browse ELG to find the resource you are looking for

The European Language Grid provides access to Language Technology resources from all over Europe. ELG contains tools and services, language resources and information on European LT companies and research organisations as well as their projects. You can try out the services or use our APIs. Make use of an LT service, download corpora or find your future business partner – all within the European Language Grid.

TOOLS & SERVICES

ELG provides access to LT services that can be tested directly through the user interface or called through the REST API.

Cogito Discover Sentiment...
218 VIEWS UPDATED 01.02.2022
Provides a sentiment score (positive or negative) for the entities recognized in the text, and an overall score for the...

Elhuyar Basque ASR
79 VIEWS UPDATED 30.08.2021
Automatic Speech Recognition service for Basque

HENSOLDT ANALYTICS Named...
73 VIEWS UPDATED 21.12.2021
HENSOLDT ANALYTICS MediaMiningIndexer NED - named entity detection engine that provides...

MaryTTS - French male (upmc-...
58 VIEWS UPDATED 03.02.2021
Text-to-Speech Synthesis platform. This version uses the upmc-pierre-hsmm voice.

HelsinkiNLP - OPUS-MT:...
20 VIEWS UPDATED 01.03.2022
Multilingual machine translation using neural networks.

LANGUAGE RESOURCES

ELG provides access to datasets, corpora, models and other resources from all over Europe, for all European languages.

SardiStance Dataset
59 VIEWS UPDATED 12.05.2021
The SardiStance dataset collects 3,242 tweets written in Italian about the "Movimento delle Sardine", retrieved...

Bangor University's Corpus of...
34 VIEWS UPDATED 25.08.2021
This is a collection of Welsh language sentences released under a CC0license and collected by members of the...

Motion Capture 3D Sign...
16 VIEWS UPDATED 31.03.2022
The collection of annotated sign language recordings acquired by state-of-the-art 3D human body...

WikinewsSum
7 VIEWS UPDATED 27.04.2022
WikinewsSum is a summarisation dataset presented in Generating Extended and Multilingual Summari...

MIC21 image dataset
4 VIEWS UPDATED 01.06.2022
The Multilingual Image Corpus provides fully annotated objects within images with segmentation masks,...

ORGANISATIONS

ELG provides access to information about all European LT organisations, including companies, universities and research centres.

Institute for Bulgarian...
55 VIEWS UPDATED 29.09.2021
The Institute for Bulgarian Language at the Bulgarian Academy of Sciences is a leading research organisation...

Centre for Translation Studies
35 VIEWS UPDATED 28.03.2021
The Centre for Translation Studies (CTS) at the University of Vienna is one of 20 academic units of the Universit...

VoiceLab AI
33 VIEWS UPDATED 11.03.2021
VoiceLab is a technology firm from Gdańsk, working on automatic speech recognition (ASR) and natural...

Ubiquitous Knowledge...
30 VIEWS UPDATED 03.05.2022
The UKP Lab was founded in 2006 by Prof. Dr. Iryna Gurevych and is part of the Computer Science Department a...

Newspoint
22 VIEWS UPDATED 02.03.2021

Share your resources through the European Language Grid

ELG is the perfect place for others to discover your organisation and your LT resources and for you to get in touch with new customers and collaborators from all over Europe. With an ELG provider account, you can easily create all types of ELG resources.

CREATE A PAGE FOR YOUR ORGANISATION

If your organisation is based in Europe and develops Language Technology, create an individual page for it in ELG to link your resources with your organisation and to ensure users can find you.

How to create an organisation page

SHARE YOUR LT RESOURCES WITH ELG USERS

ELG is the best place to share your language technology resources. We can host datasets directly in the ELG cluster and we provide a rich metadata schema so that you can describe your resource in a meaningful way.

How to contribute an ELG resource

DEPLOY YOUR LT TOOLS IN THE ELG CLUSTER

ELG services run directly in the ELG cluster as Docker containers. To share your LT tools with the European Language Technology community, you can easily transform your tool into a running ELG service.

How to contribute an ELG resource

Discover our use cases

ELHUYAR FUNDAZIOA

The Elhuyar foundation uses ELG to research resources and tools for Basque and other languages and to present information and demos of their own technologies. Igor Leturia, Speech Technologies Area Manager at Elhuyar Fundazioa, highlights the search functionality of ELG:

"It is very important to have a place such as ELG that aims (and, I would say, quite succeeds) at having information as complete as possible about technologies for European languages and companies working on LT."

UBIQUITOUS KNOWLEDGE PROCESSING LAB (UKP)

The UKP Lab of the Technical University Darmstadt uses ELG to share their resources and carry out desk research for their application of NLP to novel problems in social media and other areas. Prof. Dr. Iryna Gurevych underlines the significance of increasing the findability of resources:

"By combining information from many European institutions, the ELG highlights the strength of European language technology research and fosters collaboration."

Check out our tutorial video to learn how to use the ELG

ELG is a project made *in* Europe, *by* European organisations and *for* the European Language Technology community.

With 24 official EU languages and many more additional ones as well as more than 1500 organisations developing Language Technologies, the European LT community is highly fragmented. ELG addresses this issue – it aims to be the primary platform and marketplace for Language Technology in Europe, for all European languages, resources and technologies.

ABOUT THE ELG PROJECT

ELG is committed to community building and collaborating with various larger initiatives on the European level. One of the key results of the project is the scalable ELG cloud platform for Language Technologies. Learn more about the project

NEWS

The ELG platform is growing rapidly. Check out our blog posts to learn about recent news.

EVENTS

ELG organises conferences, workshops, hackathons and other outreach activities. ELG also plays an active role in other events.

NATIONAL COMPETENCE CENTRES

ELG has a strong and broad network of 32 National Competence Centres (NCCs), which act as local and national bridges to the ELG consortium and the European Language Grid cloud platform.

OPEN CALLS FOR ELG PILOT PROJECTS

ELG published two open calls for resource or demonstrator projects. These pilot projects extended ELG's portfolio of language resources and technologies by providing or developing missing services or solutions, or by developing novel prototypes using ELG resources.

CONTACT US

ELG is dedicated to its community. In case of questions about the platform or project, please reach out to us.

The European Language Grid has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement № 825627. The European Language Equality project has received funding from the European Union under grant agreement № LC-01641480 – 101018166.

© 2022 ELG Consortium

Terms of Use

Fig. 1 The European Language Grid cloud platform

language extinction (Rehm and Uszkoreit 2012; Kornai 2013). With an initial consortium of 52 partners, ELG's sister project ELE (European Language Equality; Jan. 2021 – June 2022) and its immediate follow-up project ELE 2 (July 2022 – June 2023) are developing a strategic agenda and roadmap for digital language equality in Europe by 2030 to address this issue by means of a coordinated, pan-European research, development and innovation programme (Rehm and Way 2023).[2]

[2] https://european-language-equality.eu

2 The European Language Grid EU Project

The original proposal for the Innovation Action "European Language Grid" (ELG) was prepared by a consortium of nine partners (Table 1) and submitted on 17 April 2018, responding to the European Commission Horizon 2020 call topic ICT-29-2018 ("A multilingual Next Generation Internet", sub-topic "European Language Grid").[3] The ELG EU project[4] started in January 2019 and finished in June 2022.[5]

1	Deutsches Forschungszentrum für Künstliche Intelligenz GmbH (Coordinator)	DFKI	Germany
2	Athena Research and Innovation Center in Information, Communication and Knowledge Technologies, Institute for Language and Speech Processing	ILSP	Greece
3	University of Sheffield	USFD	UK
4	Charles University	CUNI	Czech Republic
5	Evaluations and Language Resources Distribution Agency	ELDA	France
6	Tilde SIA	TILDE	Latvia
7	HENSOLDT Analytics GmbH	HENS	Austria
8	Expert System Iberia SL	EXPSYS	Spain
9	University of Edinburgh	UEDIN	UK

Table 1 Consortium of the ELG EU project

The project was structured into three broader *areas*. The *ELG Platform* area (WP 1, WP 2, WP 3) took care of developing the technology platform, which was built with robust, scalable, reliable and widely used open source technologies, enabling it to scale with the growing demand and supply. As an important part of the platform, the ELG catalogue contains metadata records of all resources (including services, datasets etc.), service and application types, languages as well as records of LT companies, research organisations, projects, etc. This is where the first area overlapped with the second, i. e., *ELG Content* (WP 4, WP 5), referring to the actual content of the European Language Grid in terms of processing or generation services, tools, datasets, corpora, models, language resources etc. We distinguished between *functional* content (running services that can be uploaded into and deployed from the ELG cloud platform and integrated into other systems) and *non-functional* content (datasets, corpora, lexicons, etc.). Functional LT services are created by containerising and ingesting them into ELG. One of our key goals was to make this process as easy and efficient as possible for commercial and non-commercial LT providers. These are two of the main classes of users of the third area, i. e., *ELG Community* (WP 6, WP 7), which includes all stakeholders of the ELG. Apart from commercial or academic developers of LT, these stakeholders also include companies, NGOs or

[3] https://ec.europa.eu/info/funding-tenders/opportunities/portal/screen/opportunities/topic-details/ict-29-2018

[4] https://cordis.europa.eu/project/id/825627

[5] The original runtime of 36 months was extended by six months due to the COVID-19 pandemic.

public administrations interested in purchasing or integrating Language Technologies into their own systems and applications. The ELG project collaborated – and still collaborates – with various other EU-supported research and innovation projects as well as with international networks and associations. Furthermore, ELG established a network of 32 National Competence Centres (NCCs) in as many European countries, who acted as national bridges to the project, generating interest in participating in the ELG initiative amongst relevant stakeholders from their own regions. In 2020, ELG published two open calls through which a total of 15 pilot projects were financially supported. These pilot projects extended ELG's catalogue with relevant services or datasets and realised innovative applications based on the ELG platform and available services and resources, demonstrating the usefulness of the platform. Table 2 shows all work packages of the ELG EU project.

Area	Work Package	Lead
ELG Platform	WP 1 Base Infrastructure	DFKI
	WP 2 Language Grid	ILSP
	WP 3 Interactive Interface and Information System	TILDE
ELG Content	WP 4 Services, Tools, Components	USFD
	WP 5 Language Resources, Data Sets and Models	ELDA
ELG Community	WP 6 Piloting the ELG	CUNI
	WP 7 Communication and Competence Centres	DFKI
	WP 8 Project Management and Coordination	DFKI

Table 2 Work packages of the ELG EU project

The ELG project resulted in more than 40 deliverables, the public ones of which are available online.[6] In addition to what had been originally specified in the project plan in early 2018, the project also worked on a number of activities that were not foreseen to be executed in the project proposal or grant agreement. For example, ELG organised the First International Workshop on Language Technology Platforms (IWLTP 2020).[7] Driven by the success of this workshop (Rehm et al. 2020a), a special issue of the *Language Resources and Evaluation* journal focusing on LT Platforms is currently in preparation, scheduled to be published in 2023. Motivated by the very positive feedback we have received from many different stakeholders since the beginning of the project, we decided, in 2020, to compile the present book as the definitive documentation of the project.

[6] https://www.european-language-grid.eu/deliverables

[7] https://www.european-language-grid.eu/iwltp-2020

3 Beyond the ELG EU Project

Throughout the years it has been repeatedly argued that Europe should not outsource its multilingual communication and digital language infrastructure to other continents and markets since the European demands are complex, challenging and above all unique. Instead, Europe should support and make use of its own LT community. One of the obstacles to overcome along the way has been the development of a shared technology and community platform for all European stakeholders. Now that the ELG cloud platform is finally in place, it is able to foster Language Technologies *for Europe* built *in Europe*, tailored to our languages and cultures and to our societal and economical demands, benefitting European citizens, society, innovation and industry. ELG plays the role of a shared, scalable cloud platform for the whole European LT community and it also functions as a joint marketplace and broker for a broad variety of services, products and datasets.

The ELG EU project was successfully completed in June 2022, and Release 3 of the ELG platform is ready to be used. At the time of writing, ELG provides access to more than 13,000 commercial and non-commercial language resources and technologies for all official EU languages and many national, co-official, regional and minority languages. In addition, the ELG project has contributed to validating and extending the platform with 15 pilot projects, building a pan-European community of users and providers, establishing communication and outreach channels and organising a number of large-scale conferences and smaller workshops.

Since the start of the project, we have been collaborating with the European AI on demand platform, especially with the AI4EU project, to ensure compatibility of our approaches in terms of describing resources semantically. Furthering these collaborative efforts will facilitate cross-platform search and discovery enabling ELG resources and other assets to be visible, discoverable and usable by the wider AI community. Considering the EU's plan to deploy the emerging European AI on demand platform, ELG is ready to act as the central language-related AI hub and marketplace providing access to and direct use of several thousands of LT services and datasets.

The ELG legal entity will take over further development and maintenance of ELG in the second half of 2022. At the same time, the ELG platform plays a role in several new funded projects. ELE (Jan. 2021 – June 2022) and ELE 2 (July 2022 – June 2023) have already been mentioned – ELG's sister projects are developing a strategic agenda and roadmap for achieving full digital language equality in Europe by 2030.[8] The ELG platform was and is heavily used in ELE – of special importance is the ELE dashboard, which provides a number of visualisations of the ELG catalogue, enabling various comparisons of the technology support of Europe's languages.[9] The project OpenGPT-X (Jan. 2022 – Dec. 2024), funded by the German Federal Ministry for Economic Affairs and Climate Action, develops large language models that will enable new data-driven business solutions, specifically address-

[8] https://european-language-equality.eu

[9] https://live.european-language-grid.eu/catalogue/dashboard

ing European needs.[10] In this project, many different language resources provided by ELG are used for research and development purposes. In addition, ELG will be further extended so that it complies to the specifications of the emerging Gaia-X[11] infrastructure and ecosystem, eventually integrating ELG into Gaia-X, making available many of the OpenGPT-X results (and *all* ELG resources) through Gaia-X. The project NFDI4DataScience and Artificial Intelligence (Oct. 2021 – Sept. 2026) is part of the initiative *Nationale Forschungsdateninfrastruktur* (German Research Data Infrastructure).[12] In this project, the ELG platform will be integrated into the emerging NFDI[13] infrastructure. A similar goal will be addressed by the upcoming EU project SciLake (Jan. 2023 – Dec. 2025), in which we will establish technical bridges between the ELG platform and the European Open Science Cloud (EOSC).[14] Finally, the upcoming EU project DataBri-X (Oct. 2022 – Sept. 2025) will interlink ELG and the emerging DataBri-X platform.

4 Summary of this Book

This book is structured into four different parts. Parts I, II and III describe the main results of the ELG project, while Part IV focuses on the ELG open calls and the 15 pilot projects. Below we include short summaries of the four parts.

4.1 Part I: ELG Cloud Platform

Part I provides an in-depth description of the *European Language Grid Cloud Platform*. First, Chapter 2 (p. 13 ff.) introduces the architecture and setup of the ELG cloud platform, including fundamental concepts such as the user and provider roles, the semantic metadata scheme and the different types of technologies currently supported by the platform. Afterwards, Chapter 3 (p. 37 ff.) concentrates on using ELG as a *consumer*. For this purpose, the web-based user interface, the public-facing APIs and the ELG Python SDK can be used. The complementary Chapter 4 (p. 67 ff.) examines using ELG as a *provider* of Language Technologies and Language Resources including the corresponding dashboard, service integration and various helper tools. Chapter 5 (p. 95 ff.) goes even deeper and provides a description of the ELG cloud infrastructure, e. g., the Kubernetes cluster, the storage solution etc. Finally, Chapter 6 (p. 107 ff.) examines the relation between ELG and other projects and infrastructures in terms of various technical collaborations (e. g., metadata harvesting).

[10] https://opengpt-x.de

[11] https://gaia-x.eu

[12] https://www.nfdi4datascience.de

[13] https://www.nfdi.de

[14] http://eosc.eu, https://eosc-portal.eu

4.2 Part II: ELG Inventory of Technologies and Resources

Part II focuses on the actual content of the ELG platform, i. e., it examines the *ELG Inventory of Technologies and Resources*. First, Chapter 7 (p. 131 ff.) describes the hundreds of functional Language Technology tools and services available in the ELG platform, covering machine translation, automatic speech recognition, text-to-speech synthesis as well as text analysis tools, among others. These tools and services have been and are being provided by companies as well as academic organisations. Chapter 8 (p. 151 ff.) then takes a look at the diverse set of Language Resources covering datasets, corpora, language models and other types of resources for all European languages. Many of these are hosted in ELG, available for direct download, while for others metadata records are collected from external repositories, enabling discovery through ELG as a one-stop-shop platform for the European LT community. Chapter 9 (p. 171 ff.) concludes Part II and describes the organisations, i. e., companies and research institutions, as well as projects currently represented in ELG. Our vision is for ELG to become the primary platform for Language Technology in Europe and, thus, for all organisations that develop LT to actively maintain their ELG pages, provide language tools and services as well as language resources, linking them to their own ELG pages.

4.3 Part III: ELG Community and Initiative

Part III provides an in-depth look at four different dimensions of the *ELG Community and Initiative*. First, Chapter 10 (p. 189 ff.) describes the main group of stakeholders that the EU project ELG collaborated with including various LT providers, different EU and national research projects as well as several wider initiatives. This chapter also describes the different ELG communication channels including social media. Chapter 11 (p. 205 ff.) focuses on the 32 National Competence Centres (NCCs) that the ELG project set up. The NCCs function as an international network of national networks, they support the overall mission of the ELG project. On a more abstract level, Chapter 12 (p. 219 ff.) provides a glimpse at various aspects and processes that revolve around open innovation and the marketplace concept as one of the main visions we have for the European Language Grid. Finally, Chapter 13 (p. 233 ff.) describes the ELG legal entity – including setup, challenges, products etc. – as the main instrument to sustain the ELG initiative beyond the EU project.

4.4 Part IV: ELG Open Calls and Pilot Projects

Part IV is dedicated to the *ELG Open Calls and Pilot Projects*. A considerable amount of the overall budget of the EU project European Language Grid was set aside to support a number of pilot projects that either make use of the technologies and resources provided by ELG or that extend the ELG inventory and portfolio by contributing additional technologies or resources. First, Chapter 14 (p. 257 ff.) describes the setup of the ELG open calls including designed and implemented procedures, boards, evaluation criteria etc. The following 15 chapters – Chapter 15 (p. 271 ff.) to Chapter 29 (p. 355 ff.) – report on the 15 pilot projects, selected from more than 200 project proposals in an expert-driven evaluation procedure.

References

European Parliament (2018). *Language Equality in the Digital Age. European Parliament resolution of 11 September 2018 on Language Equality in the Digital Age (2018/2028(INI)*. URL: http://www.europarl.europa.eu/doceo/document/TA-8-2018-0332_EN.pdf.

Gaspari, Federico, Owen Gallagher, Georg Rehm, Maria Giagkou, Stelios Piperidis, Jane Dunne, and Andy Way (2022). "Introducing the Digital Language Equality Metric: Technological Factors". In: *Proceedings of the Workshop Towards Digital Language Equality (TDLE 2022; co-located with LREC 2022)*. Ed. by Itziar Aldabe, Begoña Altuna, Aritz Farwell, and German Rigau. Marseille, France, pp. 1–12. URL: http://www.lrec-conf.org/proceedings/lrec2022/workshops/TDLE/pdf/2022.tdle-1.1.pdf.

Grützner-Zahn, Annika and Georg Rehm (2022). "Introducing the Digital Language Equality Metric: Contextual Factors". In: *Proceedings of the Workshop Towards Digital Language Equality (TDLE 2022; co-located with LREC 2022)*. Ed. by Itziar Aldabe, Begoña Altuna, Aritz Farwell, and German Rigau. Marseille, France, pp. 13–26. URL: http://www.lrec-conf.org/proceedings/lrec2022/workshops/TDLE/pdf/2022.tdle-1.2.pdf.

Kornai, Andras (2013). "Digital Language Death". In: *PLoS ONE* 8.10. DOI: 10.1371/journal.pone.0077056. URL: https://doi.org/10.1371/journal.pone.0077056.

Rehm, Georg, Kalina Bontcheva, Khalid Choukri, Jan Hajic, Stelios Piperidis, and Andrejs Vasiljevs, eds. (2020a). *Proc. of the 1st Int. Workshop on Language Technology Platforms (IWLTP 2020, co-located with LREC 2020)*. Marseille, France. URL: https://www.aclweb.org/anthology/volumes/2020.iwltp-1/.

Rehm, Georg and Stefanie Hegele (2018). "Language Technology for Multilingual Europe: An Analysis of a Large-Scale Survey regarding Challenges, Demands, Gaps and Needs". In: *Proceedings of the 11th Language Resources and Evaluation Conference (LREC 2018)*. Ed. by Nicoletta Calzolari, Khalid Choukri, Christopher Cieri, Thierry Declerck, Sara Goggi, Koiti Hasida, Hitoshi Isahara, Bente Maegaard, Joseph Mariani, Hélène Mazo, Asuncion Moreno, Jan Odijk, Stelios Piperidis, and Takenobu Tokunaga. Miyazaki, Japan: ELRA, pp. 3282–3289. URL: https://aclanthology.org/L18-1519.pdf.

Rehm, Georg, Katrin Marheinecke, Stefanie Hegele, Stelios Piperidis, Kalina Bontcheva, Jan Hajic, Khalid Choukri, Andrejs Vasiļjevs, Gerhard Backfried, Christoph Prinz, José Manuel Gómez Pérez, Luc Meertens, Paul Lukowicz, Josef van Genabith, Andrea Lösch, Philipp Slusallek, Morten Irgens, Patrick Gatellier, Joachim Köhler, Laure Le Bars, Dimitra Anastasiou, Albina Auksoriūtė, Núria Bel, António Branco, Gerhard Budin, Walter Daelemans, Koenraad De Smedt, Radovan Garabík, Maria Gavriilidou, Dagmar Gromann, Svetla Koeva, Simon Krek, Cvetana Krstev, Krister Lindén, Bernardo Magnini, Jan Odijk, Maciej Ogrodniczuk, Eiríkur Rögnvaldsson, Mike Rosner, Bolette Pedersen, Inguna Skadina, Marko Tadić, Dan Tufiş, Tamás Váradi, Kadri Vider, Andy Way, and François Yvon (2020b). "The European Language Technology Landscape in 2020: Language-Centric and Human-Centric AI for Cross-Cultural Communication in Multilingual Europe". In: *Proceedings of the 12th Language Resources and Evaluation Conference (LREC 2020)*. Ed. by Nicoletta Calzolari, Frédéric Béchet, Philippe Blache, Christopher Cieri, Khalid Choukri, Thierry Declerck, Hitoshi Isahara, Bente Maegaard, Joseph Mariani, Asuncion Moreno, Jan Odijk, and Stelios Piperidis. Marseille, France: ELRA, pp. 3315–3325. URL: https://www.aclweb.org/anthology/2020.lrec-1.407/.

Rehm, Georg and Hans Uszkoreit, eds. (2012). *META-NET White Paper Series: Europe's Languages in the Digital Age*. 32 volumes on 31 European languages. Heidelberg etc.: Springer.

Rehm, Georg and Hans Uszkoreit, eds. (2013). *The META-NET Strategic Research Agenda for Multilingual Europe 2020*. Heidelberg, New York, Dordrecht, London: Springer. URL: http://www.meta-net.eu/vision/reports/meta-net-sra-version_1.0.pdf.

Rehm, Georg, Hans Uszkoreit, Sophia Ananiadou, Núria Bel, Audronė Bielevičienė, Lars Borin, António Branco, Gerhard Budin, Nicoletta Calzolari, Walter Daelemans, Radovan Garabík, Marko Grobelnik, Carmen García-Mateo, Josef van Genabith, Jan Hajič, Inma Hernáez, John Judge, Svetla Koeva, Simon Krek, Cvetana Krstev, Krister Lindén, Bernardo Magnini, Joseph Mariani, John McNaught, Maite Melero, Monica Monachini, Asunción Moreno, Jan Odjik, Maciej Ogrodniczuk, Piotr Pęzik, Stelios Piperidis, Adam Przepiórkowski, Eiríkur Rögnvaldsson, Mike Rosner, Bolette Sandford Pedersen, Inguna Skadiņa, Koenraad De Smedt, Marko Tadić, Paul Thompson, Dan Tufiş, Tamás Váradi, Andrejs Vasiļjevs, Kadri Vider, and Jolanta Zabarskaite (2016). "The Strategic Impact of META-NET on the Regional, National and International Level". In: *Language Resources and Evaluation* 50.2, pp. 351–374. DOI: 10.1007/s10579-015-9333-4. URL: http://link.springer.com/article/10.1007/s10579-015-9333-4.

Rehm, Georg and Andy Way, eds. (2023). *European Language Equality: A Strategic Agenda for Digital Language Equality*. Cognitive Technologies. Forthcoming. Springer.

STOA (2018). *Language equality in the digital age – Towards a Human Language Project*. STOA study (PE 598.621), IP/G/STOA/FWC/2013-001/Lot4/C2. URL: https://data.europa.eu/doi/10.2861/136527.

Vasiljevs, Andrejs, Khalid Choukri, Luc Meertens, and Stefania Aguzzi (2019). *Final study report on CEF Automated Translation value proposition in the context of the European LT market/ecosystem*. DOI: 10.2759/142151. URL: https://op.europa.eu/de/publication-detail/-/publication/8494e56d-ef0b-11e9-a32c-01aa75ed71a1/language-en.

Part I
ELG Cloud Platform

Chapter 2
The European Language Grid Platform: Basic Concepts

Stelios Piperidis, Penny Labropoulou, Dimitris Galanis, Miltos Deligiannis, and Georg Rehm

Abstract In the fragmented Language Technology (LT) landscape of multilingual Europe, ELG has set out to bring together language resources and technologies (LRTs) and boost the LT sector and its activities. The primary goal is to build a scalable and comprehensive cloud platform for providers, developers, integrators and consumers of language resources and technologies. We describe the basic concepts of the ELG platform in terms of its architecture, the functionalities and services offered to its types of users and the policies it implements. We present the ELG repository, its catalogue features, the LT services execution environment as well as the metadata model underlying the platform operations and the resources life cycle, from creation to publication. We also discuss the compliance of ELG with the FAIR principles and the relation to other platforms and infrastructure initiatives which have inspired certain aspects and with which ELG has been establishing strong links.

1 Introduction

The overarching objective of the European Language Grid (ELG, Rehm et al. 2021) is to tackle the observed fragmentation in the European Language Technology (LT) landscape by bringing together Language Resources and Technologies (LRTs), commercial and non-commercial, and through multiple multi-level services support and boost the LT sector and LT activities in Europe. The primary technological goal is to build a scalable cloud-based platform through which developers and providers of language resources and technologies can not only deposit and upload their resources and technologies into ELG, but also deploy them through the platform and make use of the services, technologies and resources made available by others. ELG is a marketplace through which consumers and integrators of LRTs can discover, try out

Stelios Piperidis · Penny Labropoulou · Dimitris Galanis · Miltos Deligiannis
Institute for Language and Speech Processing, R. C. "Athena", Greece, spip@athenarc.gr, penny@athenarc.gr, galanisd@athenarc.gr, mdel@athenarc.gr

Georg Rehm
Deutsches Forschungszentrum für Künstliche Intelligenz GmbH, Germany, georg.rehm@dfki.de

G. Rehm (ed.), *European Language Grid*, Cognitive Technologies,
https://doi.org/10.1007/978-3-031-17258-8_2

and integrate the resources and technologies they require for their own research and application development.

The primary services of the platform are dedicated to the deposition, discovery, distribution and deployment of Language Resources and Technologies. ELG already offers access to thousands of commercial and non-commercial LTs and ancillary LRs for all European languages and more. These include processing and generation services, tools, applications for written and spoken language, as well as datasets, corpora, lexical resources, language models and computational grammars.

ELG also supports the promotion and collaboration of LT stakeholders through an extensive catalogue of organisations (companies, SMEs, academic and research organisations and groups, etc.) active in the LT community. Organisations can describe, promote and distribute their services and resources all in one place. Complemented with an expanding catalogue of European and national projects that have funded the production of LRTs and related activities, the catalogue of the ELG platform offers an overview of the European LT landscape. ELG, therefore, also acts as an observatory of LT, consolidating existing and legacy tools, services, LRs, and information about them, as well as newly emerging ones. This, in turn, enables the identification of gaps and imbalances between the LRTs offered for all European languages, a valuable instrument for the support of digital language equality in Europe.

ELG is conceived as a platform for the whole LT community. Primarily for Europe, ELG is a platform built *by* the European LT community *for* the European LT community, including industry, innovation and research. For the population of the catalogue of its platform, ELG builds bridges to existing initiatives and reaches agreements for harvesting and importing information (i. e., metadata) and resources from other infrastructures, platforms and repositories under mutually agreed conditions, business policies, acknowledgement and attribution of the source, and collaborates in joint initiatives and crowdsourcing campaigns.

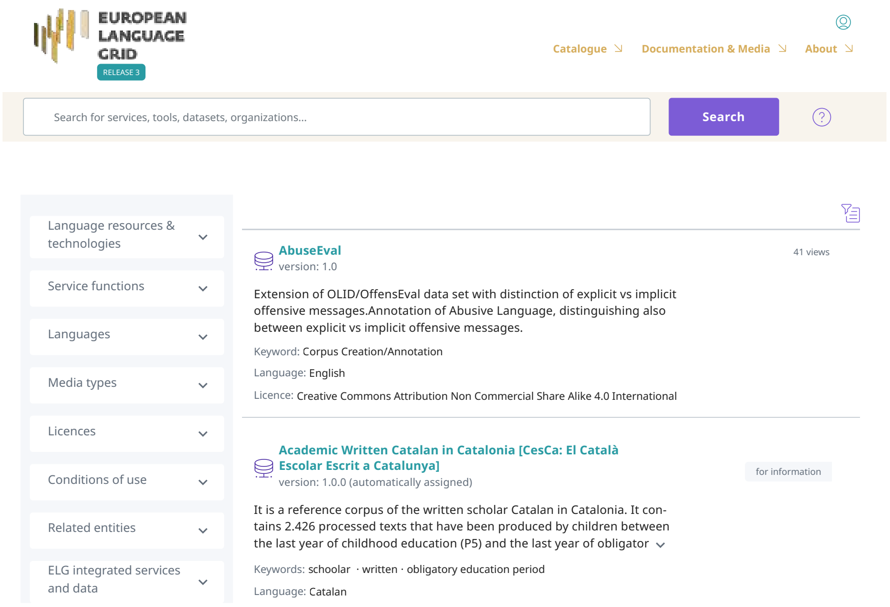
This chapter introduces the basic concepts of the ELG platform, while the subsequent chapters go into more detail with regard to functionalities offered to consumers (Chapter 3) and providers (Chapter 4), the cloud infrastructure (Chapter 5) and the synergies with other initiatives (Chapter 6). We first give an overview of the platform features (Section 2) and its users (Section 3). Section 4 presents the architecture of ELG. Sections 5 and 6 present the models and policies that influence the design and operations of the ELG platform, i. e., the metadata model, and the publication life cycle of catalogue entries. Section 7 positions the ELG platform with regard to the FAIR principles (Wilkinson et al. 2016).

2 Overview of the ELG Platform

The ELG platform combines the features of a catalogue (Section 2.1), a repository (Section 2.2), and an execution environment for running services (Section 2.3).

2.1 Catalogue

All LRTs are accessed through their metadata records in the catalogue (Figure 1). Providers can describe and share their LRTs; they can upload them to be hosted in ELG, or they can only describe them and provide access to them through other locations, such as institutional or national repositories, or private repositories of commercial organisations. They can also create dedicated pages for their organisations, describe their offerings and services and interlink all their LRTs through their own pages.



AbuseEval
version: 1.0

Extension of OLID/OffensEval data set with distinction of explicit vs implicit offensive messages.Annotation of Abusive Language, distinguishing also between explicit vs implicit offensive messages.

Keyword: Corpus Creation/Annotation
Language: English
Licence: Creative Commons Attribution Non Commercial Share Alike 4.0 International

Academic Written Catalan in Catalonia [CesCa: El Català Escolar Escrit a Catalunya]
version: 1.0.0 (automatically assigned)

It is a reference corpus of the written scholar Catalan in Catalonia. It contains 2.426 processed texts that have been produced by children between the last year of childhood education (P5) and the last year of obligator

Keywords: schoolar · written · obligatory education period
Language: Catalan

Fig. 1 Browse/Search page of the ELG catalogue

Additionally, the ELG catalogue includes metadata records imported automatically from other sources, through standard harvesting protocols and dedicated converters, thus resulting in an extensive and continuously growing inventory of LRTs as well as of organisations and projects in the LT domain.

LRT consumers, i. e., users, and other interested parties can search for and discover LRTs using free text search and faceted views of the catalogue. Users can select and view the detailed descriptions of LRTs to see if they fit the users' needs. Users can access the resources, either directly if hosted in ELG, or be re-directed to the URL from where the resources are accessible. Users can also search for organisations, browse them, and view their activities on their profile pages. If these organisations have also described the LRTs they developed, users can navigate to the respective pages for more details. Last, users can also discover the LT-related projects in which organisations participated and that have helped fund the organisations' LRT development. Finally, users can export and download the metadata descriptions or share the pages on social media.

2.2 Repository of Language Resources and Technologies

LRT providers can upload their resources to be hosted in the ELG cloud infrastructure, and to be made available to consumers for direct download. Providers must specify the licensing conditions under which the resources can be used. Depending on the terms, ELG will allow immediate download (for open access resources) or impose further measures (authentication and authorisation). Commercial LRTs, distributed for download at a fee, will be available for purchase using a user-friendly billing service.

ELG as a repository is committed to making data, services and their metadata FAIR, i. e., findable, accessible, interoperable and reusable (Wilkinson et al. 2016). The assignment of persistent identifiers in the form of Digital Object Identifiers (DOIs)[1] for the data and services hosted in ELG is among the main steps towards this objective; the FAIR principles, detailed in Section 7, form an integral part of the ELG policies aiming to support the requirements posed by research results reproducibility objectives and practices.

2.3 Running Language Technology Cloud Services

To benefit from the advanced features of ELG, providers can integrate LT tools as ready-to-deploy services, following our specifications (Chapter 4). In this case, consumers can test the tools and services using the trial UIs or APIs offered by ELG, and, ultimately, integrate them in their workflows and systems. For commercial services, billing services will be available to allow pay-for-use services with seamless access and use in the minimum possible number of steps.

ELG provides a set of standard APIs which cover all principal service types (see Chapter 3, Section 3, p. 50 ff., for more details): *information extraction and annotation services* for *text and speech*, *text-to-text* services (most notably machine translation services, but also summarisers, anonymisers, etc.), *classification services for text or image*, such as language identifiers, fake news detectors, sentiment analysers, etc., *speech recognition* services, *text-to-speech synthesis* services, and *image OCR* (optical character recognition) services.

The technical specifications give service providers a set of easy-to-implement integration options from which they can select the one that best fits their needs. All that is required is that they upload an image of their tool or service using one of these options in a container registry and provide access to ELG.

[1] https://www.doi.org

ELG maintains a dedicated container registry for LT services.[2] As the images of LT services are partly pulled from registries external to the ELG project, this registry serves as a point to collect LT service images when they are ingested into the ELG and to apply versioning. This approach enables us to ensure that older versions of images remain available even if their original site no longer provides them.

To provide easy access and interaction with the ELG platform also for programmers, a Python SDK has been developed on top of the various ELG programmatic interfaces providing simple methods to easily interact with the platform and consume resources in Python (see Chapter 3, Section 4, p. 55 ff., for more details).

3 User Types and User Model

Specified by its mission, ELG targets various types of users, broadly classified into:

Providers of LRTs, both commercial and academic, albeit with different requirements (the former seek to promote and sell their products and activities, while the latter wish to make their resources available for research or look for cooperation to further develop them in new projects or commercialize them),

Consumers of LRTs, including companies developing LT tools, services and applications, integrators, researchers using LRT for their studies, etc.,

LT laypersons interested in finding out more about LT and its uses,

Funding authorities and stakeholders that wish to get an overview of the LT field and landscape, trends and prospects with regard to languages, domains etc.

All users can browse the catalogue and access, view and inspect the detailed descriptions of the assets listed in the catalogue, and download resources available with open access licences. For further interactions with the ELG platform, registration is required and can be performed with a simple and user-friendly self-service procedure. The types of permitted actions and access level are determined by the user role: *registered consumers* can run integrated services and download resources that are available for free download to authenticated users; *providers* can, in addition, describe all types of assets, upload content files, and integrate services according to the ELG technical requirements; two specific user roles (*validator* and *administrator*) are reserved for ELG team members responsible for the management of the catalogue, metadata records and data files, in accordance with the ELG policies (Section 6) including the overall platform maintenance and administrative operations.

[2] registry.european-language-grid.eu

4 Architecture

The ELG platform uses state-of-the art technologies and is designed to evolve over time to address new requirements or technological advancements. The choices made in the architectural design and implementation allow for scaling with the growing demand and supply for compute resources and lay the foundation for interoperable data and service spaces.

All subsystems are built with robust, scalable, reliable, widely used open source technologies, as described below. Docker containers[3] are used for all services and applications which comprise the ELG platform, while Kubernetes[4] is used for container orchestration. Conceptually, ELG takes the form of a three-layered platform, with each layer grouping together the main subsystems responsible for the platform's functionalities: *base infrastructure*, *platform back end*, *platform front end* (Figure 2).

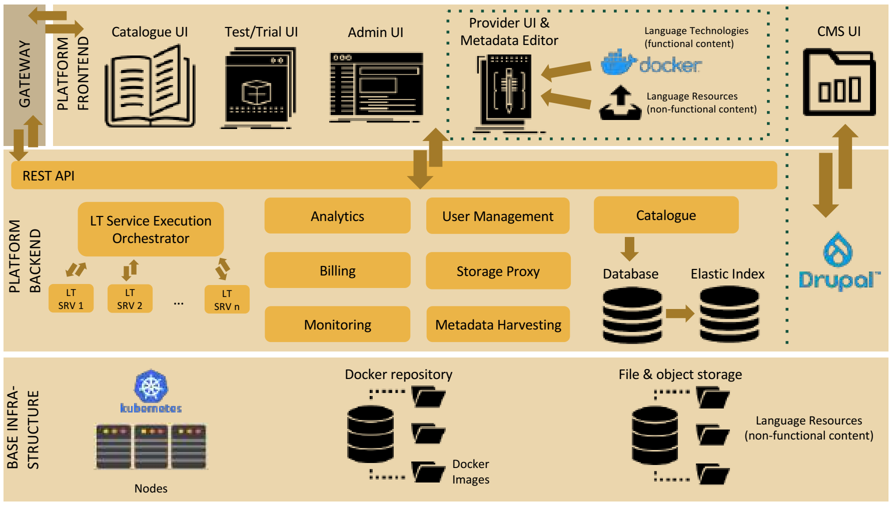


Fig. 2 ELG platform architecture

The *base infrastructure* is the layer on which all ELG software components are deployed and run. It includes the supporting tools that facilitate development and management of the ELG platform software. It is composed, first and foremost, of the compute nodes running the platform, alongside their respective volume storage and networking facilities; these are organised in two different clusters, one for development and one for production purposes. It also comprises public and private

[3] A Docker image of an application contains its actual code and all required dependencies required to run it; e. g., the operating system, frameworks, settings, configuration files, libraries, etc. Containers are instantiations of images and can be thought of as lightweight virtual machines.

[4] Kubernetes is a framework that enables and simplifies the deployment, scaling and management of containers, see https://kubernetes.io.

container registries, which host all images for the ELG platform components and for the LT services integrated in the platform. In addition, it includes an S3-compatible file and object storage, through which data resources uploaded by providers as well as backups of core platform components are persisted. This layer also includes a set of Git[5] repositories for the source code of the platform software apps and for the individual LT services implementations of specific providers. Chapter 5 (p. 95 ff.) provides more information on the base infrastructure.

The *platform back end* consists of all the components that enable the operation of the ELG platform, i. e., the catalogue core components, the component for processing LT services and platform support as well as management components. The catalogue component, implemented using Django[6], interfaces with a PostgreSQL[7] database for storing the metadata records and an index, which uses ElasticSearch[8]. The LT service execution server offers a common REST API for calling LT services integrated in the platform, and handles failures, time-outs, etc. Finally, separate modules are used for the user management and authentication module (based on Keycloak[9], an identity and access management solution), the analytics, monitoring, metadata harvesting and the proxy for interacting with the S3-compatible storage.

The *platform front end* layer consists of the static pages maintained in a Content Management System (CMS). These provide information on the ELG project and initiative, and the platform UIs for the different types of users, i. e., consumers, providers, validators, and administrators. These include the catalogue pages (browse, search, view), and the dashboard pages customised for the different user types, UIs for registering (describing and uploading) LRTs and other assets and supporting the publication life cycle, implemented using React[10], and the trial UIs for services integrated in ELG. The catalogue UI consumes REST services exposed by the ELG platform back end (e. g., catalogue application, LT Service execution server).

Chapters 3 (p. 37 ff.) and 4 (p. 67 ff.) provide more information on the back end and front end layers of the European Language Grid platform.

5 Catalogue Contents and Metadata Model

All types of LT assets as well as all LT-related meta-information are brought together, aligned and interlinked. This set of information[11] is formally structured and harmonised in ELG using the ELG-SHARE metadata model[12] catering for the full

[5] https://git-scm.com

[6] https://www.djangoproject.com

[7] https://www.postgresql.org

[8] https://www.elastic.co

[9] https://www.keycloak.org

[10] https://reactjs.org

[11] https://european-language-grid.readthedocs.io/en/stable/all/A2_Metadata/Metadata.html

[12] https://gitlab.com/european-language-grid/platform/ELG-SHARE-schema

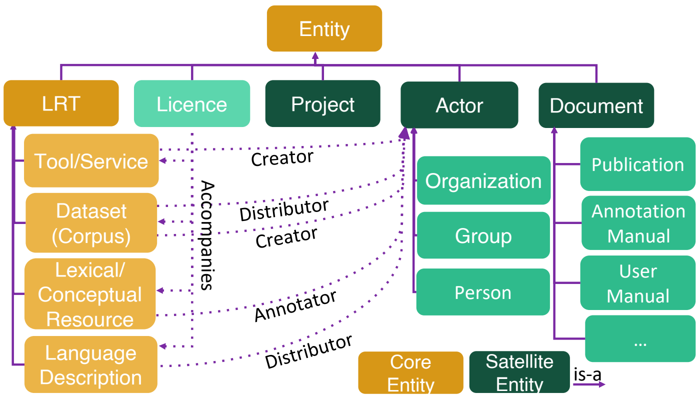


Fig. 3 ELG entities

language data and services life cycle and their related entities (Labropoulou et al. 2020). The ELG model covers the following types of entities (Figure 3).

- *Language resources and technologies (LRTs)*, further classified into:
  - *Corpora*, i. e., datasets of mono/bi/multilingual text documents, audio/video recordings, multimedia datasets, parallel corpora, translation memories, etc.
  - *Lexical/conceptual resources*, including lexica, ontologies, gazetteers, term lists, computational dictionaries, etc.
  - *Language descriptions*, which mainly refer to computational grammars, statistical and machine learning models
  - *Tools/services*, i. e., pieces of software offered as locally executable code or web services, hosted and running in the ELG cloud platform or remotely
- Related/satellite entities, such as actors, be it *persons* or *organizations* that have created or that curate resources, *projects* that have funded them or in which they have been used, as well as *licences* and accompanying *documents* (e. g., publications related to the resource, user manuals, technical documents, etc.)

The ELG model lies at the heart of the platform and supports its key operations. In particular, it aims to 1. support the discoverability of all catalogue contents; 2. enable accessibility by human users and, where possible or required, machines (e. g., including links to URLs that offer direct access to a resource or service); 3. address (at the metadata level) interoperability requirements of resources belonging to the same types and media, but coming from different sources with different descriptions, as well as between resources of different types and media (e. g., between datasets and services to be used for their processing); and, 4. finally, satisfy documentation needs at different levels of granularity, ranging from the strict enforcement of technical metadata required for the deployment of ELG-compatible services to rather loose descriptions of resources imported from general purpose catalogues.

The metadata model builds upon previous work from the META-SHARE metadata model (Gavrilidou et al. 2012), which caters for the description of language resources and language-processing technologies, and its application profiles, i. e., ELRC-SHARE (Piperidis et al. 2018a), OMTD-SHARE (Labropoulou et al. 2018), CLARIN-SHARE (Piperidis et al. 2018b), which extend, restrict and adapt the basic model to specific domains and areas (e. g., public domain resources, text and data mining domain, etc.), and the MS-OWL ontology[13] (McCrae et al. 2015; Khan et al. 2022), which is the RDF/OWL representation of the model.

The model builds along three key concepts, each of which is associated with a distinctive set of metadata elements:

- *resource type*, with the four subtypes described above;
- *media type*, which specifies the form or physical medium of the resource. The notion of medium is preferred over the written, spoken or multimodal distinction, as it has clearer semantics and allows us to view LRs as a set of modules, each of which can be described through a distinctive set of features. Thus, the following media type values are foreseen: *text*, *audio*, *image*, *video* and *numerical text* (referring to numerical data, such as biometrical, geospatial data, etc.). To cater for multimedia and multimodal language resources (e. g., a corpus of videos and subtitles, or a corpus of audio recordings and transcripts, a sign language corpus with videos and texts, etc.), language resources are represented as *consisting* of at least one media part;
- *distribution*, which, following the DCAT[14] model (Albertoni et al. 2020; Maali and Erickson 2014), refers to any physical form of the resource that can be distributed and deployed by end-users.

These elements give rise to a modular structure, in which metadata elements are attached to the appropriate level ("class"). The "LanguageResource" class includes properties common to all resource and media types, such as those used for identification purposes (title, description, etc.), recording provenance (creation, publication dates, creators, providers, etc.), contact points, etc. More technical features and classification elements differ across resource and media types and are, thus, attached to combinations thereof; for example, a corpus may take elements specific to annotation processes, while the description of a computational lexicon encodes, e. g., whether it includes lemmas, examples, grammatical information, translation equivalents, etc. Technical features, such as format, size, information on licensing and mode of access are properties of the distribution. They can also differ across resource type. For example, corpora can be distributed as PDF files or as simple text files, lexical resources in tabular form or queried through an interface, while tools may be available as source code, executable files or web services. Each of these forms can be licensed under different terms: source code may be available at a price for integration in other applications, while an API may be offered for research purposes without any fee. Figure 4 illustrates a subset of the elements for a tool/service.

[13] http://w3id.org/meta-share/meta-share

[14] https://www.w3.org/TR/vocab-dcat-3/

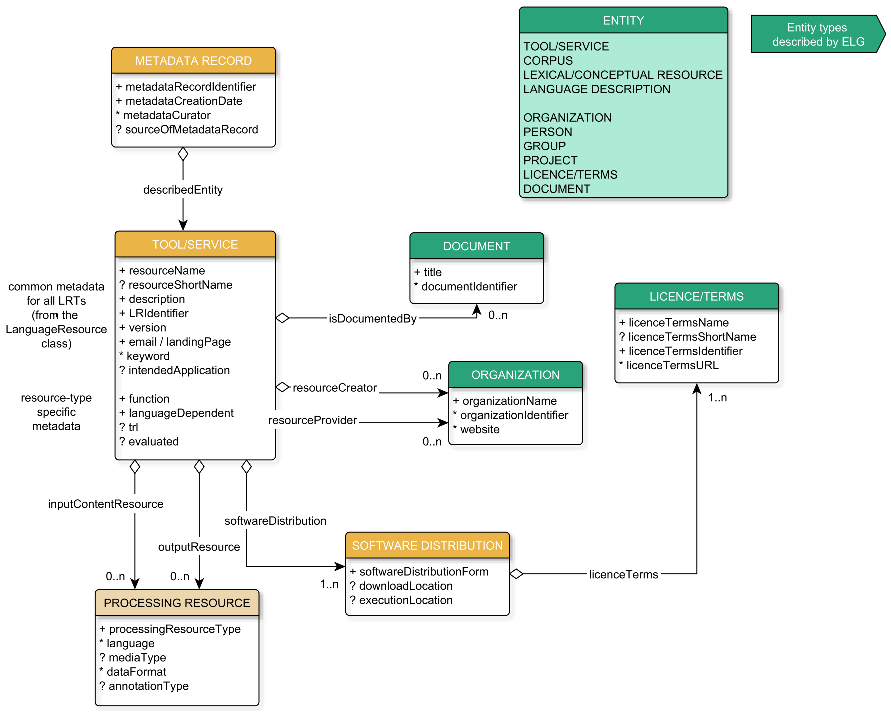


Fig. 4 Excerpt of the minimal schema for tools/services

The schema allows for the description of the full life cycle of language resources (see, e. g., Rehm 2016), from conception and creation to integration in applications and usage. All this information leads to a complex and demanding schema; to ensure flexibility and uptake by resource providers, the elements are classified into three levels of optionality:

- *mandatory:* elements that are necessary
- *recommended:* elements that can help the current or future use of the resource, or useful information that providers have not yet standardised
- *optional:* all remaining information

The minimal schema comprises all mandatory elements which must be filled for a metadata record to be considered ELG-compliant and eligible to be registered in the platform. Recently, a "relaxed" version of the ELG schema was introduced as a way of handling metadata records with "lighter" information imported from other catalogues in ELG, but this version of the schema is allowed only under specific circumstances. Chapter 6 discusses this in more detail. Below, we summarise the metadata categories considered mandatory for the description of resources (Figures 6 to 10 in the Appendix provide an overview for each resource type).

- Administrative information: these features are important for the identification of an LRT (resource name, version, description which includes information on the contents, provenance information, any other information deemed useful and helpful for consumers, etc.), contact information (landing page with additional information or a contact email).
- Classification information: one or more free text keywords that support the findability of the resource.
- Usage information: separate distributions for each distributable form of the resource, with the following elements: the distribution form (i. e., whether it can be downloaded, accessed through an interface, deployed as a web service, etc.), the licensing terms under which it can be used (licence name and URL); if the resource is not uploaded in ELG, an access or download link.
- Legal/ethical information for data resources: whether personal or sensitive data is included and, if applicable, information on anonymisation.
- Technical information: depending on the resource type
  - for tools/services: the function (i. e., the task it performs, e. g., named entity recognition, machine translation, speech recognition, etc.), the technical specifications of its input (at least the resource type it processes, e. g., corpus, text, etc.), whether it is language independent and, if not, the input languages; depending on the function, further information may be required (e. g., the languages of the output resource for machine translation services);
  - for all data resources[15]: features on the language following the BCP 47[16] guidelines, multilinguality type, resource subtype with different values (e. g., terminological glossary, ontology, etc. for lexical/conceptual resources, raw or annotated for corpora); size and format information must also be added separately for each distribution and media part;
  - in addition, specifically for models: the intended application (e. g., machine translation, named entity recognition, etc.), the model function (e. g., zero-shot classification), and model type (e. g., embeddings, Bayesian model, n-gram model, etc.);
  - specifically for grammars and lexical/conceptual resources: the encoding level of their contents (i. e., whether they contain morphological, syntactic, semantic, etc. information).

For organisations and projects, all that is required is the name (official title). However, we also recommend a free text description with the activities of the organisation or the project summary respectively, and the URL of its website. The LT area(s) in which the organisation/project activities are related to and one or more keywords increase its visibility and findability. For big organisations with multiple divisions (e. g., academic institutions with schools, faculties, departments, or multinational

[15] A resource can consist of one or more media parts, which must be described separately, for example, for a corpus of video recordings and their subtitles in various languages, the language value must be indicated separately for each part.

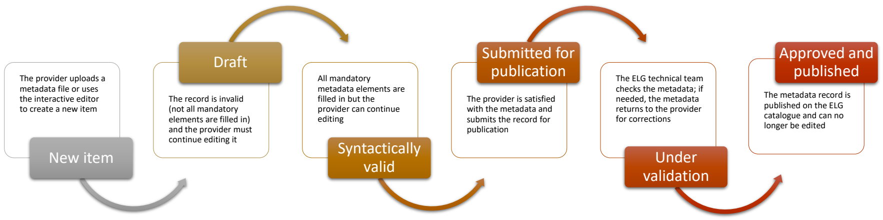
[16] https://www.rfc-editor.org/info/bcp47

companies with branches), both the parent organisation and division(s) can be registered and a link between them added.

For standardisation purposes, the ELG schema favours controlled vocabularies over free-text fields, especially when these are associated with internationally acknowledged standards, best practices or widespread vocabularies, e. g., ISO 3166 for region codes (ISO 2020), RFC 5646 for languages[17] (Phillips and Davis 2009), etc. The implementation in the form of an XML Schema Definition (XSD) imports elements from two ontologies, i. e., the MS-OWL ontology, which includes most elements and controlled vocabularies, and the OMTD-SHARE ontology[18] (Labropoulou et al. 2018) reserved for the controlled vocabularies of LT categories (also referred to as "LT taxonomy"), data formats, annotation types and methods.

6 Publication Life Cycle

ELG considers the quality of metadata records to be of primary importance as it contributes to the discovery and usage of resources. We defined a set of policies that take into account the source and the process through which a record has been entered in the ELG catalogue.



The provider uploads a metadata file or uses the interactive editor to create a new item
New item
Draft
The record is invalid (not all mandatory elements are filled in) and the provider must continue editing it
All mandatory metadata elements are filled in but the provider can continue editing
Syntactically valid
Submitted for publication
The provider is satisfied with the metadata and submits the record for publication
The ELG technical team checks the metadata; if needed, the metadata returns to the provider for corrections
Under validation
Approved and published
The metadata record is published on the ELG catalogue and can no longer be edited

Fig. 5 ELG publication life cycle

The ELG publication life cycle consists of a set of states through which an entry progresses, from its creation in the ELG platform until it is published (Figure 5). A *new item* is created each time a provider adds a new metadata record. The record can remain at the *draft* status as long as the provider wishes, in which case no validation checks are made – apart from validation of the data types of the metadata elements (e. g., that a URL is properly formulated). At the *syntactically valid* status, a metadata record must comply with the minimal version of the ELG schema (i. e., all mandatory elements must be filled in). The provider can still continue to edit it until they are satisfied with the description and can then submit it for publication; once submitted, the provider is notified by email. While the record is *submitted for publication* the

[17] https://datatracker.ietf.org/doc/html/rfc5646
[18] http://w3id.org/meta-share/omtd-share/

entry is validated at the metadata, technical and legal level. The validation, which is described in more detail in Chapter 3, aims to check the consistency of the description and, where required, its technical compliance with the ELG specifications; it does not include any qualitative evaluation of the resource itself. The validation is currently performed by the ELG team. When validators identify a problem, they contact the provider and recommend changes and additions to the metadata; in such cases, the status is changed to *syntactically valid* again and the provider is notified to make the appropriate amendments. When the validators have approved an item, it is automatically visible via the ELG catalogue. Published metadata records cannot be edited any more, i. e., they are immutable.

Metadata records added by individuals go through the whole publication life cycle. Human validation aims at ensuring a minimum level of quality included in the records, which can be achieved through interactions with the provider. This procedure cannot be adopted for metadata records automatically imported from other catalogues. For these, the responsibility for the quality and extent of information lies with the source catalogue. The same policy, that of accepting records as is, has been adopted for records added through bulk initiatives, such as the collaborative survey of LRTs undertaken in the context of the European Language Equality project[19] and described in Chapter 6.

7 ELG and the FAIR Principles

The publication of the FAIR principles (Wilkinson et al. 2016) marked a landmark for infrastructures that support the sharing and re-use of data resources. The FAIR principles are guidelines set to enhance re-usability of data by improving their findability, accessibility, interoperability and re-usability. They are intended both for humans and machines, and put an emphasis on machine actionability, i. e., the capacity of computational systems to find, access, interoperate, and reuse data with no or minimal human intervention.[20] ELG has implemented mechanisms and policies to ensure that resources (data and software) included in ELG as well as the metadata that describe them are FAIR, i. e., adhere to the FAIR principles.[21]

Findability principles

- *F1 – (Meta)data are assigned a globally unique and persistent identifier* Resources hosted in ELG and ELG-compatible services are assigned a DOI (Digital Object Identifier)[22] provided by DataCite[23]. Metadata for resources will also have their own unique identifier created on the basis of the resource

[19] https://european-language-equality.eu

[20] https://www.go-fair.org/fair-principles/

[21] https://force11.org/info/the-fair-data-principles/

[22] https://www.doi.org

[23] https://datacite.org

DOI. For metadata records that do not have an accompanying file and hence cannot be assigned a DOI, we use their URL as an identifier.

- *F2 – Data are described with rich metadata*
  The ELG metadata schema is rich in information. Providers are encouraged to add not only the mandatory but also recommended information. The validation process for resources and services aims at improving metadata quality.
- *F3 – Metadata clearly and explicitly include the identifier of the data they describe*
  The element "identifier" (with the "identifier scheme" attribute) is included in the metadata record.
- *F4 – (Meta)data are registered or indexed in a searchable resource*
  All metadata records are indexed and searchable in the ELG catalogue and also accessible to search engines. In addition, we expose the metadata records of LRTs to Google's dedicated search engine for research datasets.[24]

Accessibility principles

- *A1 – (Meta)data are retrievable by their identifier using a standardised communications protocol*
  All metadata in ELG are accessible via the ELG catalogue. Resources hosted in ELG and ELG-compatible are accessible via their DOI and directly retrievable via a URL. The HTTPS protocol is used.
- *A1.1 The protocol is open, free, and universally implementable*
  HTTPS is used for providing access to metadata and resources.
- *A1.2 The protocol allows for an authentication and authorisation procedure, where necessary*
  HTTPS is used for providing access to metadata and resources. ELG uses an authentication and authorisation system.
- *A2 – Metadata are accessible, even when the data are no longer available*
  When a resource or a metadata record is deleted, a tombstone page with all the required elements following DataCite recommendations is put in place.

Interoperability principles

- *I1 – (Meta)data use a formal, accessible, shared, and broadly applicable language for knowledge representation*
  All metadata records are exported in XML format, a subset is available in JSON-LD format; work is ongoing for the export into RDF using the MS-OWL ontology.
- *I2 – (Meta)data use vocabularies that follow FAIR principles*
  The metadata elements and values are taken from two RDF/OWL ontologies, MS-OWL and OMTD-SHARE[25].

[24] https://datasetsearch.research.google.com

[25] http://w3id.org/meta-share/omtd-share

- *I3 – (Meta)data include qualified references to other (meta)data*
  Qualified relations are used for linking between versions of the resources and, in cases of imported records, for linking with their source metadata records.

Re-usability principles

- *R1 – (Meta)data are richly described with a plurality of accurate and relevant attributes*
  Alongside the "description" element where providers are advised to add as much information as possible for the benefit of human users, the ELG schema includes elements that can be used to identify potential uses of a resource and properties that make clear where they can be of use, e. g., "intended application", "service function", "domain", etc.
- *R1.1 – (Meta)data are released with a clear and accessible data usage license*
  All resources must have a licence; the licence value and a link to the licence text are included in the metadata. Metadata are also permissively licensed with a Creative Commons licence.
- *R1.2 – (Meta)data are associated with detailed provenance*
  The source for the metadata record is explicitly added in the metadata record ("metadata creator" or "source repository"). Properties about the creation of a resource are included in the metadata.
- *R1.3 – (Meta)data meet domain-relevant community standards*
  With regard to the metadata, the ELG schema is based on META-SHARE, a well-established metadata vocabulary in the LT community. For the tools and services added in the ELG catalogue, the technical specifications follow current best practices (e. g., preparing a Docker image). For data, a set of recommendations, taking into account established file formats, standards, and de facto best practices, is under construction.

8 Related Platforms and Infrastructures

ELG builds upon previous work of the ELG consortium partners and the wider European LT community (Rehm et al. 2020b), especially META-NET[26] and ELRC[27].

The ELG platform shares common features and goals with other platforms, repositories, projects or other initiatives: 1. a collection of LT/NLP tools or datasets, 2. a platform, which harvests metadata records from distributed sources, 3. a platform for the sharing of tools or datasets, 4. a platform for the deployment of services, 5. a repository for storing data files. Comparisons can be made along various dimensions. We include here an overview at the level of the main functionalities provided, while the respective background and technical details are presented in Chapters 3 and 4. An alternative and minimally outdated comparison is provided in Rehm et al. (2020a).

[26] http://www.meta-net.eu

[27] https://www.elrc-share.eu

META-SHARE[28] is a network of repositories (Piperidis 2012; Piperidis et al. 2014). Each repository, or node, hosts various types of resources (datasets, services, etc.) described with the META-SHARE metadata schema (Gavrilidou et al. 2012). Each node is deployed at a different organisation. The nodes periodically harvest metadata records from each other. Architecture and conceptual design of the ELG platform have been inspired by the META-SHARE setup but designed and implemented from scratch. ELG adopts a different approach as it operates as a centralised platform where individuals can directly register, download and run resources and services. Harvesting is also performed but from external catalogues (e. g., ELRC-SHARE[29], LINDAT/CLARIAH-CZ[30], etc.), as described in Chapter 6. From an engineering point of view, ELG is a radically improved version of META-SHARE, e. g., 1. ELG offers REST APIs while META-SHARE does not, 2. the ELG front end and back end are implemented as different layers that can be developed in parallel, 3. the metadata schema has been updated and extended to cover new resource types and description requirements.

The OpenMinTeD platform[31] was designed as an open, service-oriented e-Infrastructure for Text and Data Mining of scientific content (Labropoulou et al. 2018). It includes a catalogue for datasets, NLP and text mining services, worfklows, lexica etc., described with a rich metadata schema, OMTD-SHARE. REST APIs for searching, metadata and resource upload/download are provided, as in the case of ELG. OpenMinTeD was a centralised repository, and harvesting was employed as a one-off procedure for importing metadata records from a few content providers. It supported the creation of workflows from tools contained in the catalogue, and their execution on datasets provided through the same platform; the functionality was based on the Galaxy[32] worfklow management system (Afgan et al. 2018).

ELRC-SHARE[33] (Piperidis et al. 2018a) is an infrastructure developed by the European Language Resource Coordination action[34] with the objective to host, document, manage and distribute LRs pertinent to MT, with a particular focus on the needs of the eTranslation[35] service of the European Commission. It is a centralised repository with a catalogue of datasets, which are added and documented by individuals. Metadata records of tools and services are listed as for information only.

The European AI-on-demand platform, as initiated by the EU project AI4EU seeks to bring together the European AI community while promoting European values.[36] The platform is a facilitator of knowledge transfer from research to multiple

[28] http://www.meta-share.org

[29] https://www.elrc-share.eu

[30] https://lindat.mff.cuni.cz

[31] https://github.com/openminted – the OpenMinTeD platform is not available online any more.

[32] https://galaxyproject.org/learn/advanced-workflow/

[33] https://www.elrc-share.eu

[34] https://lr-coordination.eu

[35] https://cor.europa.eu/en/engage/Pages/e-translation.aspx

[36] https://www.ai4europe.eu

business and industry domains. The AI catalogue[37] is designed for hosting datasets and services in the area of AI; for instance, it includes NLP resources, computer vision services, etc. The capabilities of the metadata schema used are rather limited compared to the ELG schema. It also provides catalogues for organisations involved in AI[38], collaborating projects[39] and educational resources[40], but the catalogues are all separate, without any linking between the entities as offered in the ELG catalogue.

CLARIN[41] (Hinrichs and Krauwer 2014; Eskevich et al. 2020) is a European Research Infrastructure providing access to digital language resources and tools to researchers in the humanities and social sciences. CLARIN does not host a single repository; instead, it is organised in the form of a network of centres that operate their own repositories and catalogues. The individual centres are free in their choice of repository software and metadata schema (Broeder et al. 2008). The CLARIN Virtual Language Observatory[42] is the central catalogue which harvests metadata from all centres as well as other catalogues of interest to scholars in the target disciplines and displays them in a uniform way, although only a subset of the metadata elements are common. Processing services are catalogued centrally in the Language Switchboard [43], while some CLARIN centres make available processing services connected to their catalogues or offered separately (e. g., LINDAT/CLARIAH-CZ[44], PORTULAN-CLARIN[45], CLARIN:EL[46], etc.). Unlike ELG, there is no central compute infrastructure for deploying and running processing services.

The Language Application Grid (LAPPS Grid)[47] (Ide et al. 2014, 2016) is an open, interoperable web service platform for NLP research and development. It provides facilities for selecting and combining NLP tools and services to create workflows, composite services, and applications, and to evaluate, reproduce, and share them. It is based largely on the Galaxy[48] worfklow management system and does not actually include a catalogue. Some limited metadata have to be provided in order to create the files that are required for adding tools used in Galaxy wokflows, e. g., the name of the tool, a description, input parameters etc. For datasets no metadata are required since they are not permanently stored in Galaxy.

Hugging Face[49] is an AI/NLP company, offering repository and deployment functionalities for machine learning (Wolf et al. 2020). It hosts a large set of models and

[37] https://www.ai4europe.eu/research/ai-catalog

[38] https://www.ai4europe.eu/ai-community/organizations

[39] https://www.ai4europe.eu/ai-community/projects

[40] https://www.ai4europe.eu/education/education-catalog

[41] https://www.clarin.eu

[42] https://vlo.clarin.eu

[43] https://switchboard.clarin.eu

[44] https://lindat.mff.cuni.cz

[45] https://portulanclarin.net

[46] https://inventory.clarin.gr

[47] https://www.lappsgrid.org

[48] https://galaxyproject.org/learn/advanced-workflow/

[49] https://HuggingFace.co

datasets that can be used for model training. It offers a catalogue with a limited REST API, e. g., the API does not allow filtering search results, etc. Similar to this, there are other catalogues and repositories, such as Kaggle[50] and Papers With Code[51], which target the machine learning community. These are also community-driven, i. e., resources are registered by individuals and have their own metadata schemas.

Finally, we should mention the long lasting initiative of ELRA and the LREC community in establishing the LREC Map (Calzolari et al. 2010), as well as the growing popularity of initiatives that include general (e. g., European Open Science Cloud[52]) or federated catalogues (e. g., Gaia-X[53]) and also general repositories (e. g., Zenodo[54]), which bring together a large range of resources from and for various disciplines. See Chapter 6 for more details.

9 Conclusions

ELG has been designed as the primary platform for the European LT community, adopting a holistic view of technology development, deployment and use, bringing together language data, resources and processing services as well as the commercial and non-commercial LT actors and initiatives. ELG has established and implemented a standardised resource life cycle catering for all stages, from creation to publication and version evolution. The primary services offered are dedicated to the deposition, discovery, distribution and deployment of language resources and technologies through appropriate interfaces for technical and non-technical providers, developers, consumers and integrators. Such interfaces include web GUIs, REST APIs and a Python Software Development Kit (SDK). Its operations are supported by a metadata model underlying the description, search, discovery and distribution of resources and services, conforming to the FAIR principles. On this basis, ELG has started building bridges to existing initiatives for harvesting and importing information and resources from other infrastructures, platforms and repositories under mutually agreed conditions, business policies, acknowledgement and attribution of the source, and collaborates in joint initiatives and crowdsourcing campaigns.

References

Afgan, Enis, Dannon Baker, Bérénice Batut, Marius van den Beek, Dave Bouvier, Martin Čech, John Chilton, Dave Clements, Nate Coraor, Björn A Grüning, Aysam Guerler, Jennifer Hillman-Jackson, Saskia Hiltemann, Vahid Jalili, Helena Rasche, Nicola Soranzo, Jeremy Goecks, James

[50] https://www.kaggle.com

[51] https://paperswithcode.com

[52] https://eosc-portal.eu

[53] https://www.gaia-x.eu

[54] https://zenodo.org

Taylor, Anton Nekrutenko, and Daniel Blankenberg (2018). “The Galaxy platform for accessible, reproducible and collaborative biomedical analyses: 2018 update”. In: *Nucleic Acids Research* 46.W1, W537–W544. DOI: 10.1093/nar/gky379. URL: https://academic.oup.com/nar/article/46/W1/W537/5001157.

Albertoni, Riccardo, David Browning, Simon Cox, Alejandra Gonzalez-Beltran, Andrea Perego, and Peter Winstanley, eds. (2020). *Data Catalog Vocabulary (DCAT) – Version 2*. W3C Recommendation. URL: https://www.w3.org/TR/vocab-dcat-2/.

Broeder, Daan, Thierry Declerck, Erhard Hinrichs, Stelios Piperidis, Laurent Romary, Nicoletta Calzolari, and Peter Wittenburg (2008). “Foundation of a Component-based Flexible Registry for Language Resources and Technology”. In: *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC 2008)*. Marrakech, Morocco: ELRA. URL: http://www.lrec-conf.org/proceedings/lrec2008/pdf/364_paper.pdf.

Calzolari, Nicoletta, Claudia Soria, Riccardo Del Gratta, Sara Goggi, Valeria Quochi, Irene Russo, Khalid Choukri, Joseph Mariani, and Stelios Piperidis (2010). “The LREC Map of Language Resources and Technologies”. In: *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC 2010)*. Valletta, Malta: ELRA. URL: http://www.lrec-conf.org/proceedings/lrec2010/pdf/370_Paper.pdf.

Eskevich, Maria, Franciska de Jong, Alexander König, Darja Fišer, Dieter Van Uytvanck, Tero Aalto, Lars Borin, Olga Gerassimenko, Jan Hajic, Henk van den Heuvel, Neeme Kahusk, Krista Liin, Martin Matthiesen, Stelios Piperidis, and Kadri Vider (2020). “CLARIN: Distributed Language Resources and Technology in a European Infrastructure”. In: *Proc. of the 1st Int. Workshop on Language Technology Platforms (IWLTP 2020, co-located with LREC 2020)*. Ed. by Georg Rehm, Kalina Bontcheva, Khalid Choukri, Jan Hajic, Stelios Piperidis, and Andrejs Vasiljevs. Marseille, France: ELRA, pp. 28–34. URL: https://aclanthology.org/2020.iwltp-1.5.

Gavrilidou, Maria, Penny Labropoulou, Elina Desipri, Stelios Piperidis, Haris Papageorgiou, Monica Monachini, Francesca Frontini, Thierry Declerck, Gil Francopoulo, Victoria Arranz, and Valerie Mapelli (2012). “The META-SHARE Metadata Schema for the Description of Language Resources”. In: *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC 2012)*. Istanbul, Turkey: ELRA, pp. 1090–1097. URL: http://www.lrec-conf.org/proceedings/lrec2012/pdf/998_Paper.pdf.

Hinrichs, Erhard and Steven Krauwer (2014). “The CLARIN Research Infrastructure: Resources and Tools for eHumanities Scholars”. In: *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC 2014)*. Reykjavik, Iceland: ELRA, pp. 1525–1531. URL: http://www.lrec-conf.org/proceedings/lrec2014/pdf/415_Paper.pdf.

Ide, Nancy, James Pustejovsky, Christopher Cieri, Eric Nyberg, Denise DiPersio, Chunqi Shi, Keith Suderman, Marc Verhagen, Di Wang, and Jonathan Wright (2016). “The Language Application Grid”. In: *Worldwide Language Service Infrastructure*. Ed. by Yohei Murakami and Donghui Lin. Cham: Springer, pp. 51–70. DOI: 10.1007/978-3-319-31468-6_4.

Ide, Nancy, James Pustejovsky, Christopher Cieri, Eric Nyberg, Di Wang, Keith Suderman, Marc Verhagen, and Jonathan Wright (2014). “The Language Application Grid”. In: *Proc. of the Ninth International Conference on Language Resources and Evaluation (LREC 2014)*. Reykjavik, Iceland: ELRA. URL: http://www.lrec-conf.org/proceedings/lrec2014/pdf/926_Paper.pdf.

ISO (2020). *ISO 3166 – Country Codes*. International Organization for Standardization. URL: https://www.iso.org/iso-3166-country-codes.html.

Khan, Anas Fahad, Christian Chiarcos, Thierry Declerck, Daniela Gifu, Elena González-Blanco García, Jorge Gracia, Maxim Ionov, Penny Labropoulou, Francesco Mambrini, and John P. McCrae (2022). “When Linguistics Meets Web Technologies. Recent advances in Modelling Linguistic Linked Open Data”. In: *Semantic Web Journal*. Accepted for publication.

Labropoulou, Penny, Dimitris Galanis, Antonis Lempesis, Mark Greenwood, Petr Knoth, Richard Eckart de Castilho, Stavros Sachtouris, Byron Georgantopoulos, Stefania Martziou, Lucas Anastasiou, Katerina Gkirtzou, Natalia Manola, and Stelios Piperidis (2018). “OpenMinTeD: A Platform Facilitating Text Mining of Scholarly Content”. In: *Proceedings of WOSP 2018 (co-located with LREC 2018)*. Miyazaki, Japan: ELRA, pp. 7–12. URL: http://lrec-conf.org/workshops/lrec2018/W24/pdf/13_W24.pdf.

Labropoulou, Penny, Katerina Gkirtzou, Maria Gavriilidou, Miltos Deligiannis, Dimitris Galanis, Stelios Piperidis, Georg Rehm, Maria Berger, Valérie Mapelli, Michael Rigault, Victoria Arranz, Khalid Choukri, Gerhard Backfried, José Manuel Gómez Pérez, and Andres Garcia-Silva (2020). "Making Metadata Fit for Next Generation Language Technology Platforms: The Metadata Schema of the European Language Grid". In: *Proceedings of the 12th Language Resources and Evaluation Conference (LREC 2020)*. Ed. by Nicoletta Calzolari, Frédéric Béchet, Philippe Blache, Christopher Cieri, Khalid Choukri, Thierry Declerck, Hitoshi Isahara, Bente Maegaard, Joseph Mariani, Asuncion Moreno, Jan Odijk, and Stelios Piperidis. Marseille, France: ELRA, pp. 3421–3430. URL: https://www.aclweb.org/anthology/2020.lrec-1.420/.

Maali, Fadi and John Erickson, eds. (2014). *Data Catalog Vocabulary (DCAT) – Version 1*. W3C Recommendation. URL: https://www.w3.org/TR/2020/SPSD-vocab-dcat-20200204/.

McCrae, John Philip, Penny Labropoulou, Jorge Gracia, Marta Villegas, Víctor Rodríguez-Doncel, and Philipp Cimiano (2015). "One Ontology to Bind Them All: The META-SHARE OWL Ontology for the Interoperability of Linguistic Datasets on the Web". In: *The Semantic Web: ESWC 2015 Satellite Events*. Ed. by Fabien Gandon, Christophe Guéret, Serena Villata, John Breslin, Catherine Faron-Zucker, and Antoine Zimmermann. Lecture Notes in Computer Science. Springer International Publishing, pp. 271–282. URL: https://link.springer.com/chapter/10.1007/978-3-319-25639-9_42.

Phillips, Addison and Mark Davis (2009). *Tags for Identifying Languages*. Tech. rep. RFC 5646. Internet Engineering Task Force. URL: https://datatracker.ietf.org/doc/rfc5646.

Piperidis, Stelios (2012). "The META-SHARE Language Resources Sharing Infrastructure: Principles, Challenges, Solutions". In: *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*. Ed. by Nicoletta Calzolari, Khalid Choukri, Thierry Declerck, Mehmet Uğur Doğan, Bente Maegaard, Joseph Mariani, Asuncion Moreno, Jan Odijk, and Stelios Piperidis. Istanbul, Turkey: ELRA.

Piperidis, Stelios, Penny Labropoulou, Miltos Deligiannis, and Maria Giagkou (2018a). "Managing Public Sector Data for Multilingual Applications Development". In: *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*. Ed. by Nicoletta Calzolari, Khalid Choukri, Christopher Cieri, Thierry Declerck, Sara Goggi, Koiti Hasida, Hitoshi Isahara, Bente Maegaard, Joseph Mariani, Hélène Mazo, Asuncion Moreno, Jan Odijk, Stelios Piperidis, and Takenobu Tokunaga. Miyazaki, Japan: ELRA. URL: http://www.lrec-conf.org/proceedings/lrec2018/pdf/648.pdf.

Piperidis, Stelios, Penny Labropoulou, and Maria Gavriilidou (2018b). "clarin:el An infrastructure for the documentation, sharing and processing of language data (in Greek)". In: *Proceedings of the 12th International Conference on Greek Linguistics (ICGL12)*. Vol. 2. Berlin, Germany: Edition Romiosini/CeMoG, Freie Universität Berlin, pp. 851–869. URL: http://www.cemog.fu-berlin.de/en/icgl12/offprints/piperidis-lampropoulou-gavriilidou/icgl12_Piperidis-et-al.pdf.

Piperidis, Stelios, Harris Papageorgiou, Christian Spurk, Georg Rehm, Khalid Choukri, Olivier Hamon, Nicoletta Calzolari, Riccardo del Gratta, Bernardo Magnini, and Christian Girardi (2014). "META-SHARE: One year after". In: *Proceedings of the 9th Language Resources and Evaluation Conference (LREC 2014)*. Ed. by Nicoletta Calzolari, Khalid Choukri, Thierry Declerck, Hrafn Loftsson, Bente Maegaard, Joseph Mariani, Asuncion Moreno, Jan Odijk, and Stelios Piperidis. Reykjavik, Iceland: ELRA, pp. 1532–1538. URL: http://www.lrec-conf.org/proceedings/lrec2014/pdf/786_Paper.pdf.

Rehm, Georg (2016). "The Language Resource Life Cycle: Towards a Generic Model for Creating, Maintaining, Using and Distributing Language Resources". In: *Proceedings of the 10th Language Resources and Evaluation Conference (LREC 2016)*. Ed. by Nicoletta Calzolari, Khalid Choukri, Thierry Declerck, Marko Grobelnik, Bente Maegaard, Joseph Mariani, Asuncion Moreno, Jan Odijk, and Stelios Piperidis. Portorož, Slovenia: ELRA, pp. 2450–2454. URL: https://aclanthology.org/L16-1388.pdf.

Rehm, Georg, Maria Berger, Ela Elsholz, Stefanie Hegele, Florian Kintzel, Katrin Marheinecke, Stelios Piperidis, Miltos Deligiannis, Dimitris Galanis, Katerina Gkirtzou, Penny Labropoulou, Kalina Bontcheva, David Jones, Ian Roberts, Jan Hajic, Jana Hamrlová, Lukáš Kačena, Khalid Choukri, Victoria Arranz, Andrejs Vasiļjevs, Orians Anvari, Andis Lagzdiņš, Jūlija Meļņika, Gerhard Backfried, Erinç Dikici, Miroslav Janosik, Katja Prinz, Christoph Prinz, Severin Stampler, Dorothea Thomas-Aniola, José Manuel Gómez Pérez, Andres Garcia Silva, Christian Berrío, Ulrich Germann, Steve Renals, and Ondrej Klejch (2020a). "European Language Grid: An Overview". In: *Proceedings of the 12th Language Resources and Evaluation Conference (LREC 2020)*. Ed. by Nicoletta Calzolari, Frédéric Béchet, Philippe Blache, Christopher Cieri, Khalid Choukri, Thierry Declerck, Hitoshi Isahara, Bente Maegaard, Joseph Mariani, Asuncion Moreno, Jan Odijk, and Stelios Piperidis. Marseille, France: ELRA, pp. 3359–3373. URL: https://www.aclweb.org/anthology/2020.lrec-1.413/.

Rehm, Georg, Katrin Marheinecke, Stefanie Hegele, Stelios Piperidis, Kalina Bontcheva, Jan Hajic, Khalid Choukri, Andrejs Vasiļjevs, Gerhard Backfried, Christoph Prinz, José Manuel Gómez Pérez, Luc Meertens, Paul Lukowicz, Josef van Genabith, Andrea Lösch, Philipp Slusallek, Morten Irgens, Patrick Gatellier, Joachim Köhler, Laure Le Bars, Dimitra Anastasiou, Albina Auksoriūtė, Núria Bel, António Branco, Gerhard Budin, Walter Daelemans, Koenraad De Smedt, Radovan Garabík, Maria Gavriilidou, Dagmar Gromann, Svetla Koeva, Simon Krek, Cvetana Krstev, Krister Lindén, Bernardo Magnini, Jan Odijk, Maciej Ogrodniczuk, Eiríkur Rögnvaldsson, Mike Rosner, Bolette Pedersen, Inguna Skadina, Marko Tadić, Dan Tufiş, Tamás Váradi, Kadri Vider, Andy Way, and François Yvon (2020b). "The European Language Technology Landscape in 2020: Language-Centric and Human-Centric AI for Cross-Cultural Communication in Multilingual Europe". In: *Proceedings of the 12th Language Resources and Evaluation Conference (LREC 2020)*. Ed. by Nicoletta Calzolari, Frédéric Béchet, Philippe Blache, Christopher Cieri, Khalid Choukri, Thierry Declerck, Hitoshi Isahara, Bente Maegaard, Joseph Mariani, Asuncion Moreno, Jan Odijk, and Stelios Piperidis. Marseille, France: ELRA, pp. 3315–3325. URL: https://www.aclweb.org/anthology/2020.lrec-1.407/.

Rehm, Georg, Stelios Piperidis, Kalina Bontcheva, Jan Hajic, Victoria Arranz, Andrejs Vasiļjevs, Gerhard Backfried, José Manuel Gómez Pérez, Ulrich Germann, Rémi Calizzano, Nils Feldhus, Stefanie Hegele, Florian Kintzel, Katrin Marheinecke, Julian Moreno-Schneider, Dimitris Galanis, Penny Labropoulou, Miltos Deligiannis, Katerina Gkirtzou, Athanasia Kolovou, Dimitris Gkoumas, Leon Voukoutis, Ian Roberts, Jana Hamrlová, Dusan Varis, Lukáš Kačena, Khalid Choukri, Valérie Mapelli, Mickaël Rigault, Jūlija Meļņika, Miro Janosik, Katja Prinz, Andres Garcia-Silva, Cristian Berrio, Ondrej Klejch, and Steve Renals (2021). "European Language Grid: A Joint Platform for the European Language Technology Community". In: *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations (EACL 2021)*. Kyiv, Ukraine: ACL, pp. 221–230. URL: https://www.aclweb.org/anthology/2021.eacl-demos.26.pdf.

Wilkinson, Mark D., Michel Dumontier, IJsbrand Jan Aalbersberg, Gabrielle Appleton, Myles Axton, Arie Baak, Niklas Blomberg, Jan-Willem Boiten, Luiz Bonino da Silva Santos, Philip E. Bourne, Jildau Bouwman, Anthony J. Brookes, Tim Clark, Mercè Crosas, Ingrid Dillo, Olivier Dumon, Scott Edmunds, Chris T. Evelo, Richard Finkers, Alejandra Gonzalez-Beltran, Alasdair J.G. Gray, Paul Groth, Carole Goble, Jeffrey S. Grethe, Jaap Heringa, Peter A.C 't Hoen, Rob Hooft, Tobias Kuhn, Ruben Kok, Joost Kok, Scott J. Lusher, Maryann E. Martone, Albert Mons, Abel L. Packer, Bengt Persson, Philippe Rocca-Serra, Marco Roos, Rene van Schaik, Susanna-Assunta Sansone, Erik Schultes, Thierry Sengstag, Ted Slater, George Strawn, Morris A. Swertz, Mark Thompson, Johan van der Lei, Erik van Mulligen, Jan Velterop, Andra Waagmeester, Peter Wittenburg, Katherine Wolstencroft, Jun Zhao, and Barend Mons (2016). "The FAIR Guiding Principles for Scientific Data Management and Stewardship". In: *Scientific Data* 3. DOI: 10.1038/sdata.2016.18. URL: http://www.nature.com/articles/sdata201618.

Wolf, Thomas, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush (2020). "Transformers: State-of-the-art Natural Language Processing". In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. ACL, pp. 38–45. DOI: 10.18653/v1/2020.emnlp-demos.6. URL: https://aclanthology.org/2020.emnlp-demos.6.

Appendix

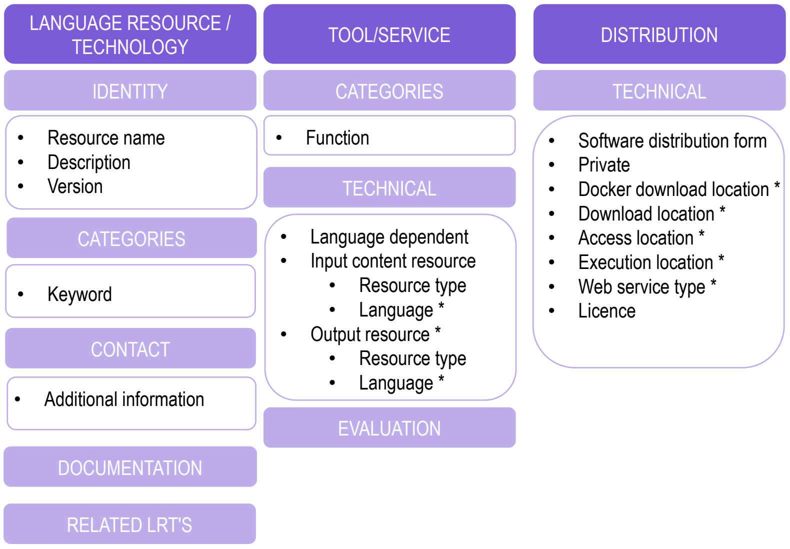


Fig. 6 ELG minimal schema version for a tool/service

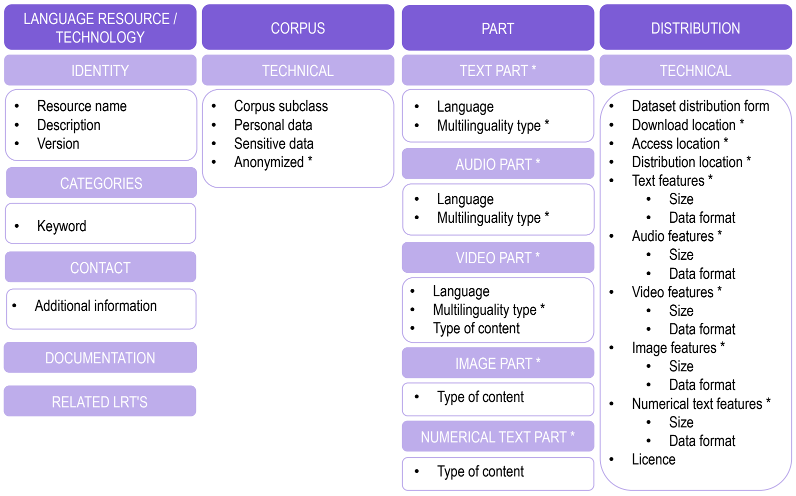


Fig. 7 ELG minimal schema version for a corpus

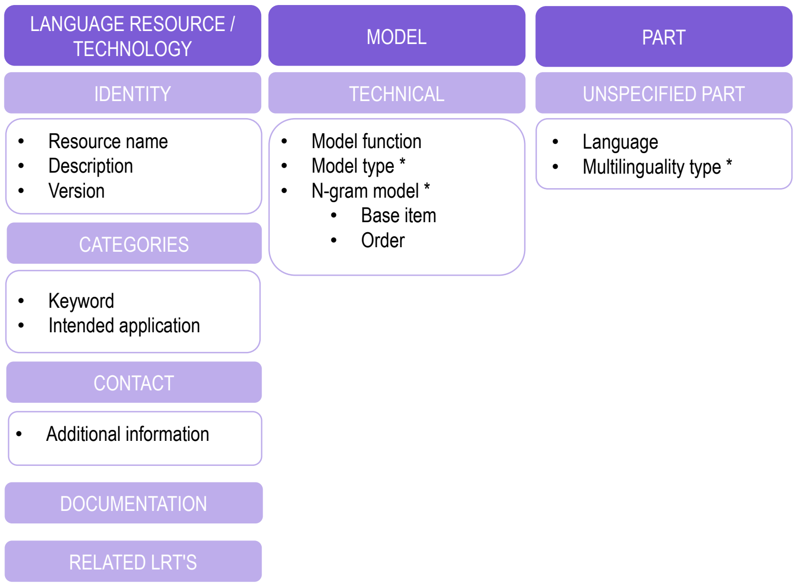


Fig. 8 ELG minimal schema version for a model

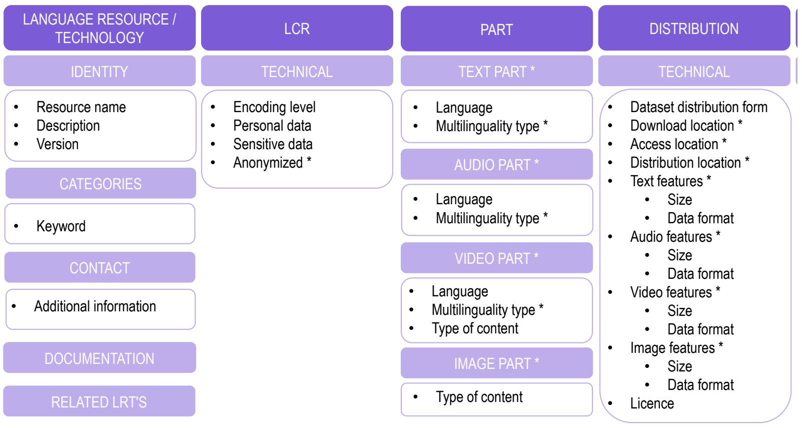


Fig. 9 ELG minimal schema version for a lexical/conceptual resource

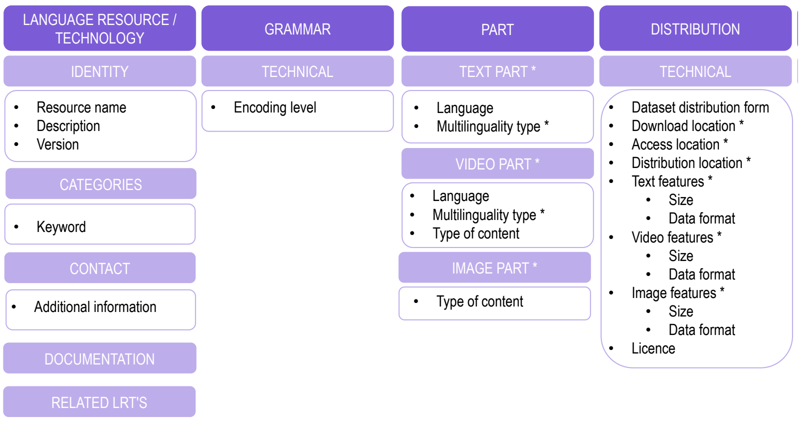


Fig. 10 ELG minimal schema version for a grammar

Chapter 3
Using the European Language Grid as a Consumer

Ian Roberts, Penny Labropoulou, Dimitris Galanis, Rémi Calizzano, Athanasia Kolovou, Dimitris Gkoumas, Andis Lagzdiņš, and Stelios Piperidis

Abstract This chapter describes the European Language Grid cloud platform from the point of view of a *consumer* who wishes to access language resources or make use of language technology tools and services. Three aspects are discussed: 1. the web-based user interface (UI) for casual and non-technical users, 2. the underlying REST APIs that drive the UI but can also be called directly by third parties to integrate ELG functionality in their own tools, and 3. the Python Software Development Kit (SDK) that we have developed to simplify access to these APIs from Python code. The chapter concludes with a preview of the upcoming payment module that will enable the sale of commercial LT services and resources through ELG, and a discussion of how ELG compares and relates to other similar platforms and initiatives.

1 Introduction

The European Language Grid (ELG) platform (Rehm et al. 2021) provides access to Language Technology (LT) tools and services, both basic Natural Language Processing (NLP) tools and end-to-end applications, as well as data resources, such as structured and unstructured datasets and corpora, Machine Learning models, lexica, ontologies, terminologies, etc. Chapters 7 (p. 131 ff.) and 8 (p. 151 ff.) present the current state of LT services as well as datasets and language resources included in the ELG platform respectively.

Ian Roberts
University of Sheffield, UK, i.roberts@sheffield.ac.uk

Penny Labropoulou · Dimitris Galanis · Athanasia Kolovou · Dimitris Gkoumas · Stelios Piperidis
Institute for Language and Speech Processing, R. C. “Athena”, Greece, penny@athenarc.gr, galanisd@athenarc.gr, akolovou@athenarc.gr, dgkoumas@athenarc.gr, spip@athenarc.gr

Rémi Calizzano
Deutsches Forschungszentrum für Künstliche Intelligenz GmbH, Germany, remi.calizzano@dfki.de

Andis Lagzdiņš
Tilde, Latvia, andis.lagzdins@tilde.lv

G. Rehm (ed.), *European Language Grid*, Cognitive Technologies,
https://doi.org/10.1007/978-3-031-17258-8_3

ELG enables consumers of Language Technology to browse through the ELG catalogue and have an overview of its contents, search for specific resources and select as well as view the features of a resource through its formal description (metadata record). Users can download resources hosted in the ELG cloud infrastructure in accordance with their licensing conditions, or, in the case of external resources, be re-directed to the location where they can be downloaded from or accessed. They can also try out services in order to assess whether they comply with their needs; for this to happen, the services must comply with the ELG technical interoperability specifications, which are outlined in Chapter 4. Furthermore, ELG includes a catalogue of commercial companies and academic and research organisations that are active in the LT domain and of EU and national projects that have funded the development and maintenance of LRTs (see Chapter 9); LRTs, actors and projects are interlinked offering a comprehensive image of the LT landscape in Europe.

Different types of users have different requirements and different levels of technical expertise, and the ELG platform provides a variety of access methods to address these; all the principal functionality of the ELG is offered through both web-based user interfaces (UIs, see Section 2) for interactive use and Application Programming Interfaces (APIs, see Section 3) for programmatic access. In addition, the ELG team supports the advanced needs of LT integrators with dedicated tools and helpers; most notably a Software Development Kit (SDK) for Python (see Section 4), which is currently the most widely used programming language in the LT community.

Supporting consumers to easily discover resources is of utmost importance, especially when a catalogue contains many entries, as in the case of ELG (over 13,000 metadata records for LRTs and 1,800 related entities at the time of writing and constantly increasing). Best practices and recommendations (Wu et al. 2019; Wilkinson et al. 2016) have been taken into account in the design and implementation of the ELG catalogue pages and interaction mechanisms with the consumers.

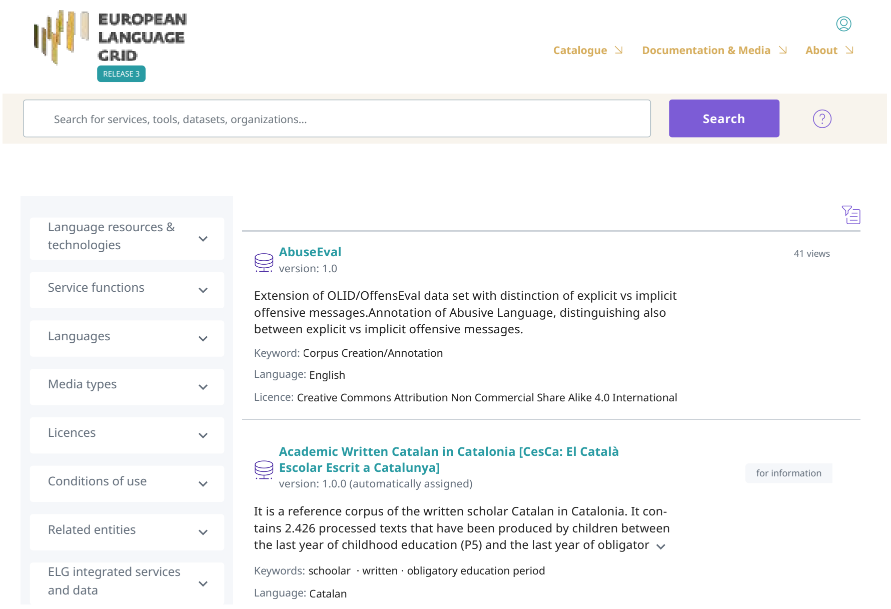
At present all functionality of the ELG platform is offered free of charge. All users can view the catalogue and metadata descriptions as well as download open access resources. In order to download resources with restrictive licences and try out ELG-compatible services, users must register in the platform, as described in Section 5. It should be noted that while the ELG platform does not currently charge fees for access to any resources or services, restrictions may apply with regard to the intended use(s) of the resource (e. g., available only for non-commercial use), request for explicit consent to licensing conditions, etc. Resources available with commercial licences are described in the ELG catalogue but for now re-directed to the providers for further information. A prototype billing module, described in Section 6, has been implemented and will be fully launched following the setup of the ELG legal entity (see Chapter 13). Finally, in Section 7 we compare the ELG platform to other similar services and initiatives, from the point of view of the service or resource consumer. A similar comparison from the point of view of the provider can be found in Chapter 4.

2 Web-based Interface

The ELG platform targets a diverse set of user types with different needs and levels of technical expertise. The primary access route for non-technical users is via the web user interface (UI), which prioritises user-friendliness and ease of use alongside raw performance considerations. The catalogue UI includes two main pages: the *catalogue page*, which offers access to the catalogue contents, and the *view pages* for each metadata record or resource (LT, LR, organisation, project).

2.1 Viewing the Catalogue

After ELG's homepage, the dedicated catalogue page (Figure 1) is the primary entry point through which users have access to the ELG platform contents and functions. Users can browse through the entire catalogue to find entries that might interest them. They can also look for specific entries, using the free text search bar, filtering the catalogue with one or more facets, or combining these two modes.



AbuseEval
version: 1.0
41 views

Extension of OLID/OffensEval data set with distinction of explicit vs implicit offensive messages.Annotation of Abusive Language, distinguishing also between explicit vs implicit offensive messages.

Keyword: Corpus Creation/Annotation
Language: English
Licence: Creative Commons Attribution Non Commercial Share Alike 4.0 International

Academic Written Catalan in Catalonia [CesCa: El Català Escolar Escrit a Catalunya]
version: 1.0.0 (automatically assigned)
for information

It is a reference corpus of the written scholar Catalan in Catalonia. It contains 2.426 processed texts that have been produced by children between the last year of childhood education (P5) and the last year of obligator

Keywords: schoolar · written · obligatory education period
Language: Catalan

Fig. 1 Browse/Search page of the ELG catalogue

The main section of the catalogue page shows all published entries sorted by name in alphabetical order. Users can also sort the entries according to the update date of the metadata record, so that they can view the most recently added entries first. The catalogue shows only the most recent version of each entry if multiple versions are registered. The snippet informs the users of additional older entries, which can be viewed and accessed through the view page of the newest version (see Section 2.3). This allows users to always keep up to date with the most recent version of a service, but also access older versions when needed, for instance, when reproducing previously published experiments.

Each entry is shown with an informative snippet, designed to serve as a preview of the full metadata record and to help users decide whether they want to explore the entry further. Following well-established practices in catalogues, each entry is represented by its name, an excerpt of its description, a set of metadata tags, and popularity indicators. The set of metadata tags has been carefully selected to accommodate consumer requirements, as identified in a user survey conducted during the ELG design and specification phase (Melnika et al. 2019) and subsequently enriched based on user feedback. All types of entries include their free-text keywords. Entries representing LRTs additionally include the resource type (represented with an icon), language(s), and licence(s). The popularity indicators, displayed at the right hand side of the snippet, consist of counts of visits of the view page of all versions of an entry, counts of downloads (for ELG-hosted resources only) and number of calls (for ELG-compatible services only; again for all versions of the entry). Finally, dedicated badges are shown for resources hosted in ELG and ELG-compatible services, as well as for a subset of the metadata records that have been imported from other catalogues with minimal metadata (see Chapter 6).

2.2 Searching the Catalogue

Search of the catalogue is supported in two different modes, which can be combined in order to refine search queries and support users in easily finding entries of interest: free text search (Section 2.2.1) and faceted search (Section 2.2.2).

2.2.1 Free Text Search

Users enter a word or phrase in the search box at the top of the catalogue page (see Figure 1) and click the "Search" button to submit the query. By default, the search functionality matches whole words using the OR operator. Advanced queries, utilising the Lucene query syntax[1], are supported, allowing users to search for partial or exact matches, words or phrases, etc. Only certain metadata elements have been indexed to make them searchable; these include a resource's name(s), short name(s), keywords and a subset of technical elements appropriate for each entry type and deemed important as a search criterion. For example, for all LRTs, additional in-

[1] https://www.lucenetutorial.com/lucene-query-syntax.html

dexed elements are the “resource type”, “language” and “licence”; for LT tools/services, “service function” is also added to the search elements.

In addition, to improve recall of search results, for those metadata elements that take values from controlled vocabularies, i. e., “service function”,“intended LT application”, and “language”, the query is expanded with the use of synonyms. Synonyms for the first two elements are derived from a taxonomy of LT activities[2], which provides the values. For alternative names of languages, besides the official ones included in the ISO 639-3 standard for language codes[3] (International Organization for Standardization 2007), we exploit open access vocabularies published as linked data, i. e., the Glottolog list of languoids (families, languages, dialects)[4], the lexvo ontology of languages[5], and the WALS list of languages[6]; all these vocabularies are offered through Glottolog.

2.2.2 Faceted Search

Users can filter the catalogue or previous search results by selecting values from the list of facets (Figure 2) on the left side of the catalogue page (Figure 1). For facets with a long list of values, such as languages and licences, the facet values are broken down into subsections or a search bar is included to refine the list.

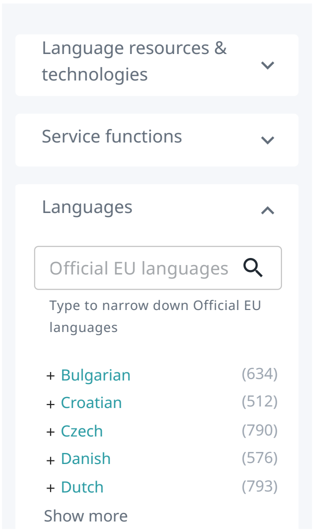


Fig. 2 Faceted search in the ELG catalogue

[2] Part of the OMTD-SHARE ontology, see http://w3id.org/meta-share/omtd-share.

[3] https://iso639-3.sil.org/code_tables/639/data

[4] https://glottolog.org

[5] http://lexvo.org/ontology

[6] https://wals.info/languoid

The facets were selected in the initial phase of the ELG development based on user preferences collected through a survey conducted for the technical platform specifications (Melnika et al. 2019). Important criteria for users searching for are language coverage (62%), licence and access conditions (59%) and availability of open source code (56%). Later on, more facets have been added to reflect updates in the metadata schema and improve search capabilities (Wu et al. 2019).

There are two facets, based on the *resource type* and *entity type* elements, that create dedicated subsets of the catalogue contents. The values are taken from the respective elements of the ELG metadata schema, but are tuned to current LT approaches. Thus, with regard to LRTs, users can view specific catalogues of tools and services, corpora, lexical/conceptual resources, models, grammars and other language descriptions. In the ELG schema the last three are subclasses of the *language description* type, but we opted to treat them as separate resource types primarily to improve the visibility of models; these are what define the state of the art for many NLP tasks and are likely to be particularly popular, so need to be easily discoverable. The two catalogues of organisations and projects are a valuable asset for boosting and activating interactions within and across the LT community (including match-making in the ELG marketplace) and eventually also for monitoring funding outcomes.

LRTs can be further filtered using the facet *ELG integrated services and data* to restrict the catalogue view to the ELG-compatible services and resources hosted in ELG, for users who wish to take advantage of the "try out" functionality offered by ELG for services or of the direct download of resources uploaded in ELG.

The facet *languages* shows the language coverage of the LRTs in the ELG catalogue, i. e., the languages of the contents of data resources and the ones that tools/services cater for. Given the scope of ELG, the official EU languages are presented in a separate group shown at the top of the facet. The encoding of language values in the catalogue follows the BCP 47 recommendations (Phillips and Davis 2009), i. e., it allows for users adding a tag consisting of subtags for language, region, script and language variants, but for simplicity of the UI the facet browser includes only the values of the *language* subtag. Moreover, it includes only one of the known names of a language; e. g., for "Catalan; Valencian", only the first name is shown. For languages and language varieties without an ISO 639 code, we show the name associated with the respective Glottocode[7] if it has one.

The facets *intended LT application* and *service function* are used for classifying LRTs and related entities with concepts specific to the LT community; consumers can search for services that perform specific functions (e. g., dependency parsers, Machine Translation tools), but also for corpora or models that have been created or can be used for a a specific application (e. g., bilingual or multilingual corpora to be used for building machine translation models), as well as for organisations and projects active in an LT area; the values of these two elements are both taken from the taxonomy of LT areas[8], and free text values that have been added by users.

[7] https://glottolog.org/meta/glossary

[8] http://w3id.org/meta-share/omtd-share/

Licensing and access conditions are among the search criteria most requested by users: *licences* gives the detailed list of licences used for LRTs in the catalogue.[9] The more coarse-grained facet *Conditions of use* groups licences by the general types of conditions they impose (e. g., “no commercial use”, “share-alike”), intended for users with little knowledge of legal terms. Users are still advised to carefully read the licence specified on the view page of each LRT for all terms and conditions.

The *media types* facet was introduced at a later stage when the number of multimodal resources included in the catalogue increased. As for languages, this refers to the media type of the contents of resources or the media type of the input/output of tools, and can be used to quickly search not only for text-related applications and resources, but also for audio, video and image ones.

The ELG catalogue includes both entries added by individuals and entries aggregated from other catalogues.[10] Thus, the facet *source* refers to the source of the metadata record. It includes the name of the catalogue from which the record has been imported or the value “ELG/ELE” for records originating in ELG or added by the collaborating project European Language Equality (ELE)[11] through processes described in Chapter 6.

2.3 Viewing Metadata Records and Resources

By clicking the title of an entry on the catalogue page, users can view its full description. Figures 3 and 4 show the view page of a tool/service and a corpus respectively. Specific view pages have been implemented for all LRT types published in ELG. Their design takes into account user preferences and requirements, design and accessibility considerations and the ELG metadata schema. They allow users to access detailed information about an item, test it, if it is a service integrated in the platform, and, finally, obtain and use it for their purposes.

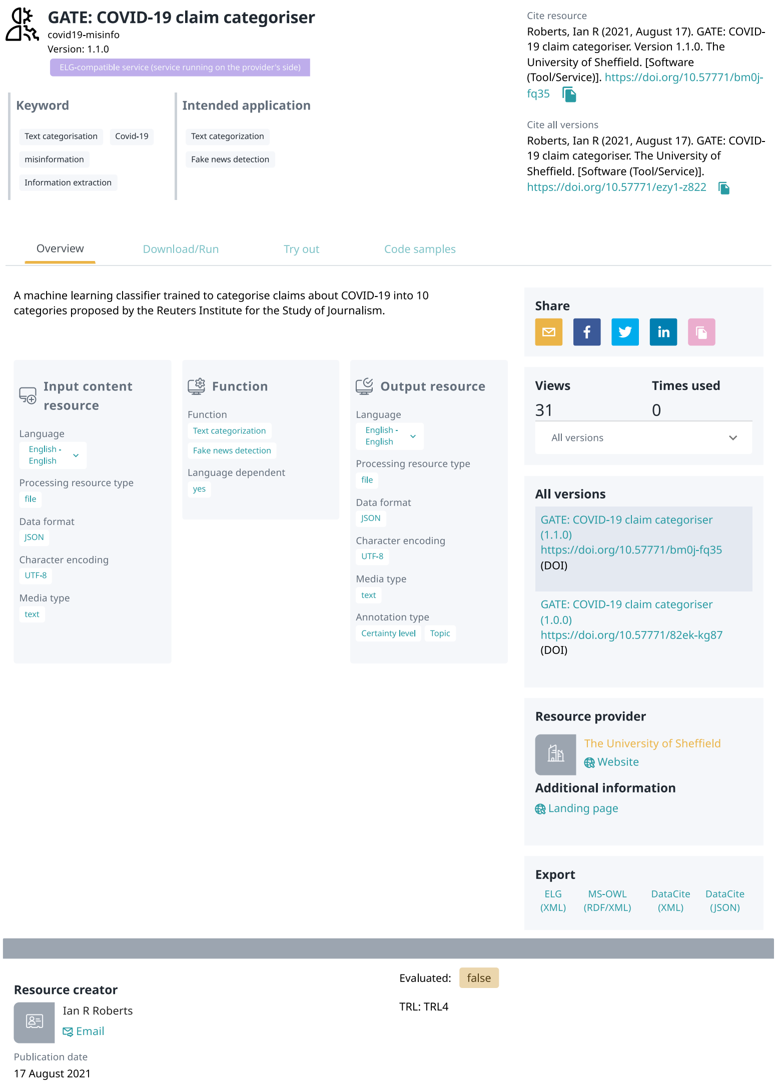
Even though the types of information shown on the view pages differ for each category, we apply a consistent visual look and feel for all of them. The information on the view page of each item comes from the respective metadata record. Taking into consideration the specificities and richness of the metadata schema, but also user-friendliness, the information is layered along specific sections of the page. Thus, view pages share a common layout that consists of a header, a right-hand sidebar, a main content area and a bottom content region; the positioning of the elements on the page and the formatting of the text is carefully thought through to draw users’ attention to the most important information.

The header shows the name and version of the resource, its resource type and optionally important flags (e. g., to indicate that a certain service is deployed in ELG).

[9] Chapter 6 discusses why this element was made mandatory.

[10] See Chapters 4 and 6 for more information on the respective modes of population.

[11] https://european-language-equality.eu



GATE: COVID-19 claim categoriser
covid19-misinfo
Version: 1.1.0
ELG-compatible service (service running on the provider's side)

Keyword
Text categorisation | Covid-19 | misinformation | Information extraction

Intended application
Text categorization | Fake news detection

Cite resource
Roberts, Ian R (2021, August 17). GATE: COVID-19 claim categoriser. Version 1.1.0. The University of Sheffield. [Software (Tool/Service)]. https://doi.org/10.57771/bm0j-fq35

Cite all versions
Roberts, Ian R (2021, August 17). GATE: COVID-19 claim categoriser. The University of Sheffield. [Software (Tool/Service)]. https://doi.org/10.57771/ezy1-z822

Overview | Download/Run | Try out | Code samples

A machine learning classifier trained to categorise claims about COVID-19 into 10 categories proposed by the Reuters Institute for the Study of Journalism.

Input content resource
Language: English - English
Processing resource type: file
Data format: JSON
Character encoding: UTF-8
Media type: text

Function
Function: Text categorization, Fake news detection
Language dependent: yes

Output resource
Language: English - English
Processing resource type: file
Data format: JSON
Character encoding: UTF-8
Media type: text
Annotation type: Certainty level, Topic

Share

Views	Times used
31	0

All versions

All versions
GATE: COVID-19 claim categoriser (1.1.0) https://doi.org/10.57771/bm0j-fq35 (DOI)
GATE: COVID-19 claim categoriser (1.0.0) https://doi.org/10.57771/82ek-kg87 (DOI)

Resource provider
The University of Sheffield
Website

Additional information
Landing page

Export
ELG (XML) | MS-OWL (RDF/XML) | DataCite (XML) | DataCite (JSON)

Resource creator
Ian R Roberts
Email

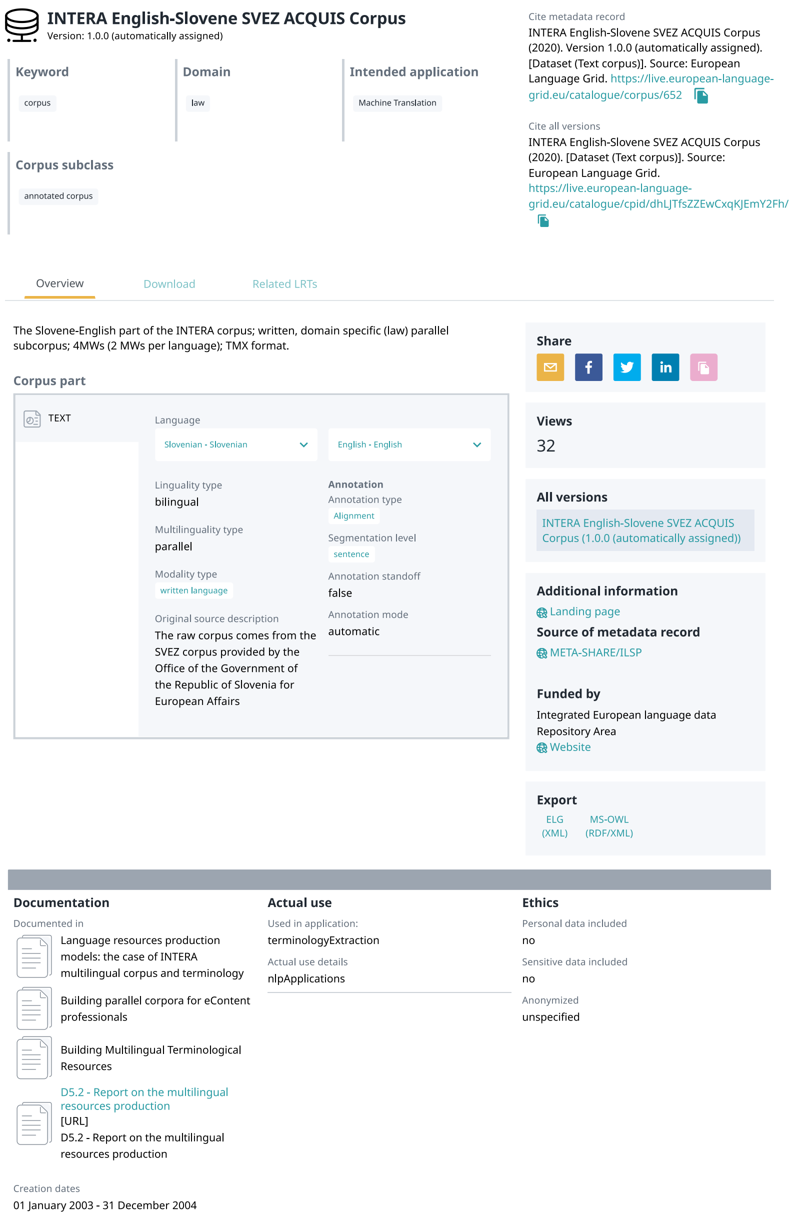
Publication date
17 August 2021

Evaluated: false

TRL: TRL4

Fig. 3 View page of an ELG-compatible service

At the top of the right-hand sidebar, the button "Claim" may appear for some of the metadata records; these are records with minimal metadata that have been imported through automatic harvesting and bulk collection procedures (see Chapter 6). The claiming process enables interested users, i. e., the rightful owners of these LRTs, to ask to curate and enrich them. The same area provides for all records information on how they can be cited, according to data and software citation principles (Smith



INTERA English-Slovene SVEZ ACQUIS Corpus
Version: 1.0.0 (automatically assigned)

Keyword
corpus

Domain
law

Intended application
Machine Translation

Corpus subclass
annotated corpus

Cite metadata record
INTERA English-Slovene SVEZ ACQUIS Corpus (2020). Version 1.0.0 (automatically assigned). [Dataset (Text corpus)]. Source: European Language Grid. https://live.european-language-grid.eu/catalogue/corpus/652

Cite all versions
INTERA English-Slovene SVEZ ACQUIS Corpus (2020). [Dataset (Text corpus)]. Source: European Language Grid. https://live.european-language-grid.eu/catalogue/cpid/dhLJTfsZZEwCxqKJEmY2Fh/

Overview Download Related LRTs

The Slovene-English part of the INTERA corpus; written, domain specific (law) parallel subcorpus; 4MWs (2 MWs per language); TMX format.

Corpus part

TEXT

Language
Slovenian - Slovenian
English - English

Linguality type
bilingual

Multilinguality type
parallel

Modality type
written language

Original source description
The raw corpus comes from the SVEZ corpus provided by the Office of the Government of the Republic of Slovenia for European Affairs

Annotation
Annotation type
Alignment

Segmentation level
sentence

Annotation standoff
false

Annotation mode
automatic

Share

Views
32

All versions
INTERA English-Slovene SVEZ ACQUIS Corpus (1.0.0 (automatically assigned))

Additional information
Landing page

Source of metadata record
META-SHARE/ILSP

Funded by
Integrated European language data Repository Area
Website

Export
ELG (XML) MS-OWL (RDF/XML)

Documentation
Documented in
Language resources production models: the case of INTERA multilingual corpus and terminology
Building parallel corpora for eContent professionals
Building Multilingual Terminological Resources
D5.2 - Report on the multilingual resources production
[URL]
D5.2 - Report on the multilingual resources production

Creation dates
01 January 2003 - 31 December 2004

Actual use
Used in application:
terminologyExtraction
Actual use details
nlpApplications

Ethics
Personal data included
no
Sensitive data included
no
Anonymized
unspecified

Fig. 4 View page of a corpus

et al. 2016; Data Citation Synthesis Group 2014) and DataCite guidelines[12]. They also have the option to share the URL link of the page by email or through social media and export the metadata record as an XML file in the ELG-compliant schema. Statistics of resource usage are shown both for the particular resource version and for all versions (if there are multiple versions). Links to other versions of the same resource are also displayed here.

In the content area, tabs split information into smaller views and enable users to navigate to offered functionalities of the platform. The first tab provides an overview of the main features of the entry that help users decide if the resource fits their needs. In terms of layout it is similar across resource types, but the information types (metadata elements) differ. Compare, for instance, Figures 3 and 4 that show the overview tab for a service and a corpus. The top shows a free text description for all record types, followed by a section for classification information (keywords, domain, service function, etc.) and an area for technical metadata, e. g., the media type(s) and language(s) of a corpus, the input and output data formats for a service, etc. The bottom section contains hyperlinks to useful documents, creation details, etc. and is again specific to resource types.

Depending on the resource type, the "Download" or "Download/Run" tab presents information related to the distribution of the resource, such as the licence under which it can be accessed, a technical description of its content files (e. g., size and format for data resources), and access to the resource itself – a direct download link if the resource is uploaded into ELG (see Section 3.2), otherwise a redirect to the resource on its provider's site. Figure 5 shows the tab for a corpus hosted in ELG.

A third tab appears if the item is related to other items, e. g., a project with the LRTs this project has funded, an organisation with the LRTs it has created and the projects it is involved in.

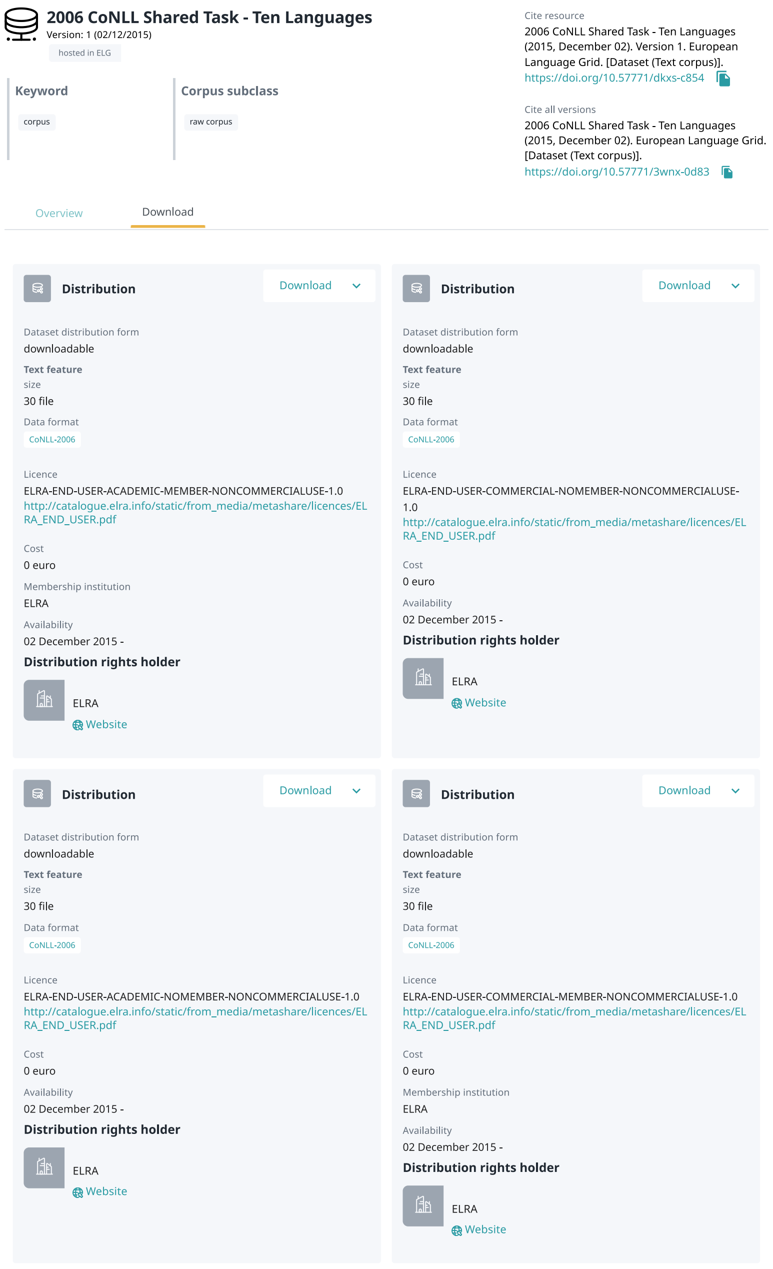
Finally, ELG-compatible services have two more tabs that enable users to try out the service (see Section 2.5) and inform them how to use it via the command line or Python SDK (see Section 4).

2.4 Consumer's Grid

Individuals can browse the catalogue, view detailed metadata cards and download open access resources without any registration. To access restricted resources and run ELG-compatible services, they must be registered with an ELG account and also logged in. For registered users, ELG offers a dashboard ("grid") for managing and performing actions on catalogue items depending on their rights (see Chapter 2 for more information on user roles and rights). As for view pages, the grid follows a similar layout which is customised for each user type.

The consumer's grid (Figure 6) allows registered users to monitor their usage of daily quotas, view details on downloads of LRTs they performed and of the services

[12] https://datacite.org/cite-your-data.html



2006 CoNLL Shared Task - Ten Languages
Version: 1 (02/12/2015)
hosted in ELG

Keyword
corpus

Corpus subclass
raw corpus

Cite resource
2006 CoNLL Shared Task - Ten Languages (2015, December 02). Version 1. European Language Grid. [Dataset (Text corpus)].
https://doi.org/10.57771/dkxs-c854

Cite all versions
2006 CoNLL Shared Task - Ten Languages (2015, December 02). European Language Grid. [Dataset (Text corpus)].
https://doi.org/10.57771/3wnx-0d83

Distribution Download

Dataset distribution form
downloadable
Text feature
size
30 file
Data format
CoNLL-2006

Licence
ELRA-END-USER-ACADEMIC-MEMBER-NONCOMMERCIALUSE-1.0
http://catalogue.elra.info/static/from_media/metashare/licences/ELRA_END_USER.pdf

Cost
0 euro
Membership institution
ELRA
Availability
02 December 2015 -
Distribution rights holder
ELRA
Website

Distribution Download

Dataset distribution form
downloadable
Text feature
size
30 file
Data format
CoNLL-2006

Licence
ELRA-END-USER-COMMERCIAL-NOMEMBER-NONCOMMERCIALUSE-1.0
http://catalogue.elra.info/static/from_media/metashare/licences/ELRA_END_USER.pdf

Cost
0 euro
Availability
02 December 2015 -
Distribution rights holder
ELRA
Website

Distribution Download

Dataset distribution form
downloadable
Text feature
size
30 file
Data format
CoNLL-2006

Licence
ELRA-END-USER-ACADEMIC-NOMEMBER-NONCOMMERCIALUSE-1.0
http://catalogue.elra.info/static/from_media/metashare/licences/ELRA_END_USER.pdf

Cost
0 euro
Availability
02 December 2015 -
Distribution rights holder
ELRA
Website

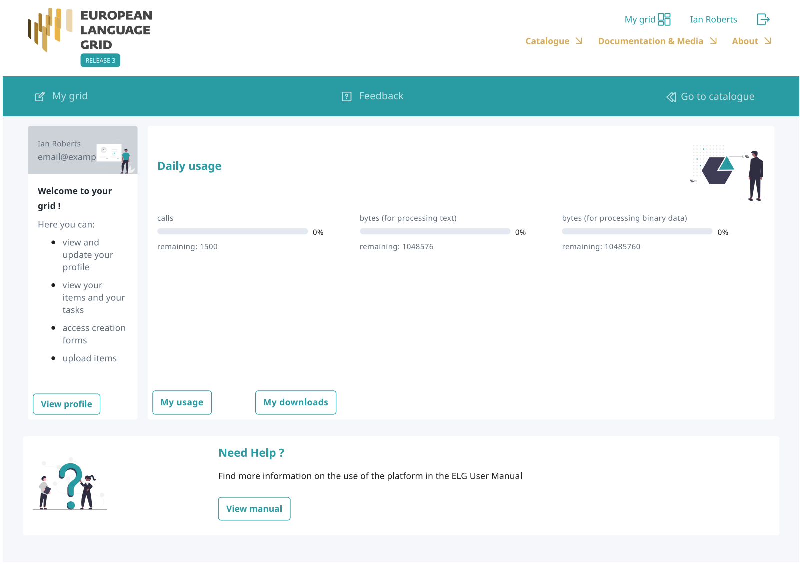
Distribution Download

Dataset distribution form
downloadable
Text feature
size
30 file
Data format
CoNLL-2006

Licence
ELRA-END-USER-COMMERCIAL-MEMBER-NONCOMMERCIALUSE-1.0
http://catalogue.elra.info/static/from_media/metashare/licences/ELRA_END_USER.pdf

Cost
0 euro
Membership institution
ELRA
Availability
02 December 2015 -
Distribution rights holder
ELRA
Website

Fig. 5 Download tab for a corpus



Ian Roberts
email@examp

Welcome to your grid !

Here you can:

- view and update your profile
- view your items and your tasks
- access creation forms
- upload items

Daily usage

calls	bytes (for processing text)	bytes (for processing binary data)
0%	0%	0%
remaining: 1500	remaining: 1048576	remaining: 10485760

Need Help ?

Find more information on the use of the platform in the ELG User Manual

Fig. 6 Consumer's grid (see Figure 4 in Chapter 4, p. 73, for the Provider's grid)

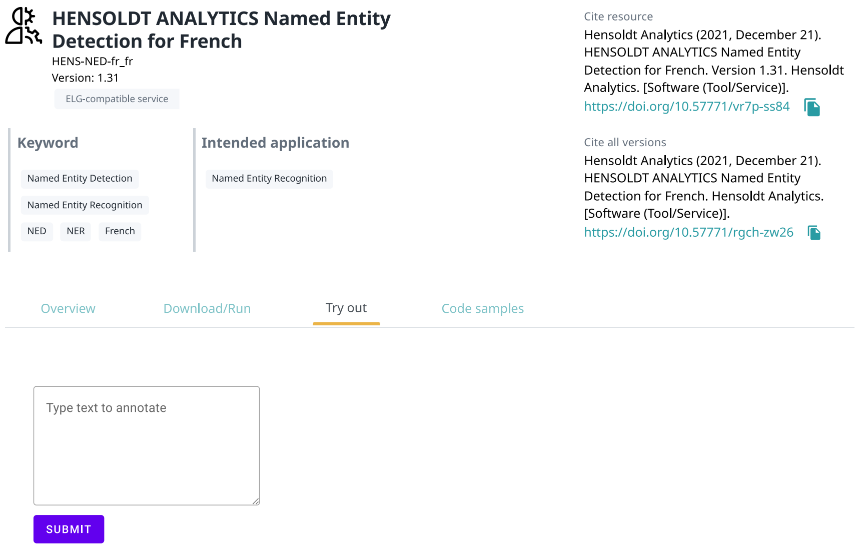
they have deployed. Additional elements of the "My grid" section that are relevant only to *provider* users are discussed in Chapter 4.

2.5 Try out UIs for Language Technology Services

One of the key benefits of having an LT service fully integrated in ELG is that users have access to a "try out" UI from which they can test the service directly using their web browser. ELG provides standard trial UIs[13] covering all principal service types:

- *Information Extraction (IE) & text analysis* services take text input and produce standoff *annotations* over that text.
  In addition to this generic text analysis UI there is also a specific one for dependency parsers that renders CoNLL-U style annotations as a tree structure.[14]
- *Text-to-text* services (most notably Machine Translation, but also summarisation, anonymisation, etc.) take text and return new text that is derived from the input.
- *Text classification* services take text input and classify it somehow (e. g., language identification, "fake news" detection, etc.)
- *Speech recognition* services accept audio and return a text transcription.

[13] Service providers whose tools do not fit one of the above UIs are free to provide their own.
[14] https://universaldependencies.org/format.html


HENSOLDT ANALYTICS Named Entity Detection for French
HENS-NED-fr_fr
Version: 1.31
ELG-compatible service

Keyword: Named Entity Detection, Named Entity Recognition, NED, NER, French

Intended application: Named Entity Recognition

Cite resource
Hensoldt Analytics (2021, December 21). HENSOLDT ANALYTICS Named Entity Detection for French. Version 1.31. Hensoldt Analytics. [Software (Tool/Service)]. https://doi.org/10.57771/vr7p-ss84

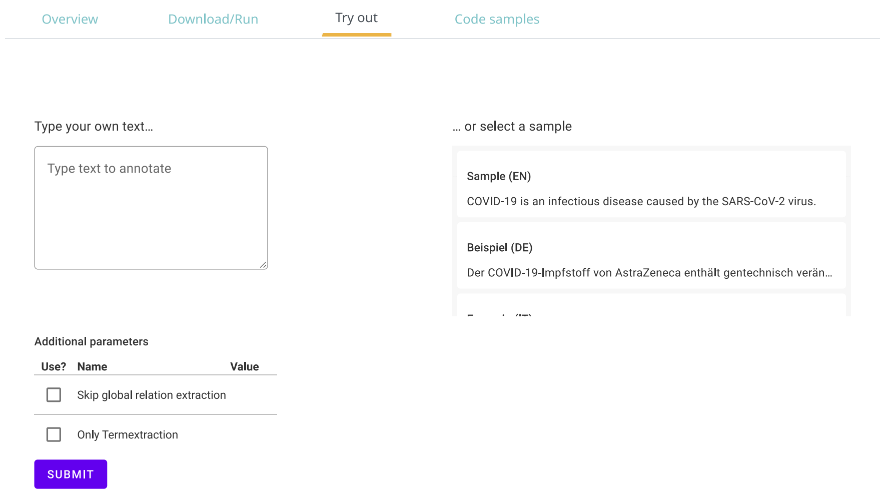
Cite all versions
Hensoldt Analytics (2021, December 21). HENSOLDT ANALYTICS Named Entity Detection for French. Hensoldt Analytics. [Software (Tool/Service)]. https://doi.org/10.57771/rgch-zw26

Fig. 7 An example “try out” UI for a named entity service

- *Audio annotation* services take audio and return standoff annotations over particular time segments of the audio stream.
- *Text-to-speech* services take text and return audio.
- *Image OCR* (optical character recognition) services take image data and return text extracted from the image.

The trial UIs for services are available to any user who has logged in to the ELG portal. The UI appears in the “Try out” tab when viewing a service in the catalogue; Figure 7 shows an example for a simple service that only requires plain text. However, some services can be much more complex, requiring additional parameters or providing snippets of sample data that users can test the service with – if a service declares these kinds of items in its metadata record, then the try out UI will automatically adapt, as shown in Figure 8. This service – also see Chapter 18 – declares two optional parameters and offers a selection of samples in different languages.

The UIs have been designed to render all of the main service response types in a user-friendly way, for example, annotations over text are shown as colour highlights (Figure 9), translated text is displayed alongside the original, audio can be played directly in the browser, etc.



Type your own text…

Type text to annotate

… or select a sample

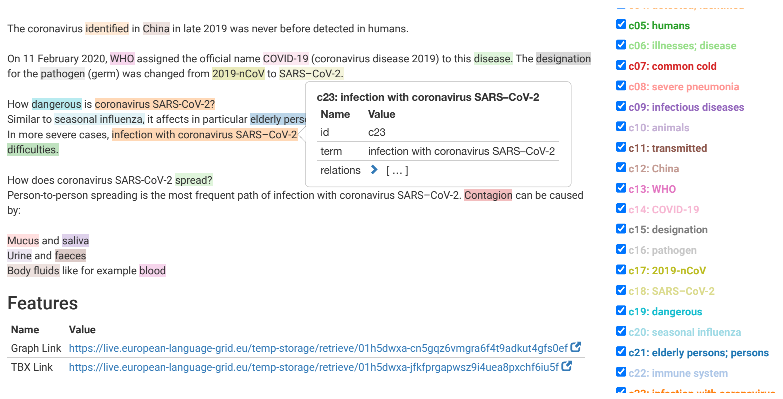
Sample (EN)
COVID-19 is an infectious disease caused by the SARS-CoV-2 virus.

Beispiel (DE)
Der COVID-19-Impfstoff von AstraZeneca enthält gentechnisch verän…

Additional parameters

Use?	Name	Value
☐	Skip global relation extraction	
☐	Only Termextraction	

Fig. 8 A more complex “try out” UI for the Text2TCS service


The coronavirus identified in China in late 2019 was never before detected in humans.

On 11 February 2020, WHO assigned the official name COVID-19 (coronavirus disease 2019) to this disease. The designation for the pathogen (germ) was changed from 2019-nCoV to SARS–CoV-2.

How dangerous is coronavirus SARS-CoV-2?
Similar to seasonal influenza, it affects in particular elderly pers
In more severe cases, infection with coronavirus SARS–CoV-2
difficulties.

How does coronavirus SARS-CoV-2 spread?
Person-to-person spreading is the most frequent path of infection with coronavirus SARS–CoV-2. Contagion can be caused by:

Mucus and saliva
Urine and faeces
Body fluids like for example blood

c23: infection with coronavirus SARS–CoV-2

Name	Value
id	c23
term	infection with coronavirus SARS–CoV-2
relations	› […]

Features

Name	Value
Graph Link	https://live.european-language-grid.eu/temp-storage/retrieve/01h5dwxa-cn5gqz6vmgra6f4t9adkut4gfs0ef
TBX Link	https://live.european-language-grid.eu/temp-storage/retrieve/01h5dwxa-jfkfprgapwsz9i4uea8pxchf6iu5f

✓ c05: humans
✓ c06: illnesses; disease
✓ c07: common cold
✓ c08: severe pneumonia
✓ c09: infectious diseases
✓ c10: animals
✓ c11: transmitted
✓ c12: China
✓ c13: WHO
✓ c14: COVID-19
✓ c15: designation
✓ c16: pathogen
✓ c17: 2019-nCoV
✓ c18: SARS–CoV-2
✓ c19: dangerous
✓ c20: seasonal influenza
✓ c21: elderly persons; persons
✓ c22: immune system

Fig. 9 Example result for the Text2TCS service showing rendered text annotations

3 Public REST APIs

The web user interfaces described above are built on top of a set of REST APIs, and the same APIs can also be called directly by third parties, allowing ELG functionality to be accessed programmatically and embedded into other tools. The current public APIs break down into three principal groups: 1. accessing/using the catalogue (Section 3.1), 2. accessing and downloading ELG-hosted data resources (Section 3.2), 3. calling ELG-hosted LT services (Section 3.3).

All APIs are HTTPS-based and use JSON as the primary data representation format. Where authentication is required, this is performed using OAuth2 access tokens issued by the ELG user management layer (see Section 5).

3.1 Accessing and Using the Catalogue

The ELG catalogue is a Python web application based on the Django REST Framework.[15] It offers a number of services as REST APIs, including the following ones which are useful for consumers: 1. searching the catalogue, 2. authorising the download of a resource or access of any resource or page, 3. retrieving the metadata description of a resource.

3.2 Downloading a Resource

ELG allows providers to upload and store the actual contents of their LRTs within the platform (data files for corpora, source code for software, etc.), and the catalogue offers an API to allow consumers to download this data subject to licensing terms.

LRT data is stored in a storage service compatible with the API of Amazon S3. Access by consumers is mediated by a Storage Proxy.[16] The proxy defers to a data management module within the catalogue application (see Section 6) to determine, based on authentication information provided by the user who attempts the download, whether that user has the permission to download the requested resource. Factors considered in making a decision include whether the resource is open access to all requesters (authenticated or not), if it requires authentication, or if the user must explicitly accept the terms of the licence prior to download.

3.3 Language Technology Service Public API

One of the great strengths of ELG is its use of a single harmonised set of APIs for all ELG-compatible LT services regardless of provider. This differs from other API aggregator platforms such as RapidAPI[17], where each service provider defines their own API and the caller must adapt their code for each different service.

For each LT service the platform provides two endpoints at which the service can be called, which implement synchronous and asynchronous modes of operation. These endpoints are implemented in the LT Service Execution Server. The endpoint URLs can be found in the `service_info` section of the metadata record JSON structure returned by the catalogue API.

The synchronous mode simply consists of a single API call in which the caller will `POST` the data to be processed and receive the results via the response to the same request. The asynchronous mode accepts the same type of request but instead of blocking the caller until the results are ready it returns a polling URL, which the

[15] https://www.django-rest-framework.org
[16] https://gitlab.com/european-language-grid/platform/s3proxy
[17] https://rapidapi.com

caller must repeatedly poll for status updates. This requires more HTTP requests but for long-running services (or those that take some time to scale up from idle) the asynchronous mode is more resilient to connection failures or intermediary proxy timeouts between the client and the ELG platform.

Any query parameters appended to the URL will be passed through to the service and may affect its behaviour – each service declares the parameters that it supports in its metadata. All available versions of a given service are exposed at the same endpoint, the `?version=...` parameter is used to select between them, with the latest version used by default if no parameter is given.

The `POST` data must have an appropriate `Content-Type` header for the service in question; services that take text (such as text analysis or MT services) expect "text/plain"[18], services that take audio (such as speech recognition) expect "audio/x-wav" or "audio/mpeg", and services that take images expect the "image/png", "image/jpeg", etc. A few services expect their input to be "structured text" that has been pre-segmented by the caller, for these the request must be presented in an ELG-defined JSON format. The response will be in JSON, in one of a variety of formats depending on the data type:

- *Standoff annotations* are represented in a style inspired by the format used by Twitter, each *type* of annotation mapping to a JSON array of objects referencing the start and end locations of the annotation (characters for text, fractional seconds for audio), and an optional set of *features*.
- *Classifications* of the whole input have their own format giving an ordered list of classes, each with an optional score.
- *New texts* such as translations of text or transcriptions of audio are returned in a structured format referred to as a "texts" response (note texts is plural). This is described in more detail below.
- *Audio* responses such as text-to-speech are still represented in JSON. Short snippets of audio can be returned inline in base 64 encoding, longer audio will typically be stored at a short-lived temporary URL for the caller to download via a separate HTTPS request.

The full specification of these response types can be found in the ELG documentation.[19] The "texts" response type is the most complex one as it is able to encode a nested tree structure of texts, where each node in the tree can be either a leaf node containing a single string of content, or a branch node containing another level of texts. The vast majority of services currently using this response format produce one of the three basic forms shown in Listing 1: a single text, a flat list of segments or alternatives, or a two-level list where each segment has a set of alternatives.

The property `role` is used to distinguish the cases. Not all services populate this property but it is encouraged; conventionally a role of "sentence", "paragraph" or "segment" denotes segments of text that are all part of the same transcript or translation, and "alternative" denotes different translations or transcriptions of the same

[18] UTF-8 encoding is the default but can be overridden by adding the `charset=...` parameter.

[19] https://european-language-grid.readthedocs.io/en/stable/all/A3_API/LTPublicAPI.html

```
// A single text
{
  "response":{
    "type":"texts",
    "texts":[
      {"content":"This is some text"}
    ]
  }
}

// A flat list of segments or alternatives
{
  "response":{
    "type":"texts",
    "texts":[
      {"content":"First sentence", "role":"sentence"},
      {"content":"Second sentence", "role":"sentence"},
    ]
  }
}

// A two level list of segments that each have a number of alternatives
{
  "response":{
    "type":"texts",
    "texts":[
      {
        "role":"sentence",
        "texts":[
          {"content":"Translation one", "role":"alternative"},
          {"content":"First translation", "role":"alternative"}
        ]
      },
      ...
    ]
  }
}
```

Listing 1 The three most common types of “texts” response

input segment. In the case of alternatives, each entry may also have a “score” representing the relative quality of the different options.

For errors (and also for warning messages), ELG, being a multilingual platform, uses a format designed to be amenable to internationalisation (i18n). Each message is represented as a JSON object with three properties “code”, “text” and “params” (see Listing 2). The property “code” is the primary identifier for the error; there is a list of standard message codes provided in the ELG documentation but providers are free to create their own codes if the standard messages do not adequately cover their needs. The property “text” is a string for the message text in English, and it may include numbered placeholders `{0}`, `{1}`, etc. If the message has placeholders,

```
{
  "code":"elg.request.type.unsupported",
  "text":"Request type {0} not supported by this service",
  "params":["audio"]
}
```

Listing 2 An example “status message” object from the ELG API, designed to be easily translated into many languages.

```
POST https://live.european-language-grid.eu/i18n/resolve?lang=fr
Content-Type: application/json

[
  {
    "code":"elg.request.type.unsupported",
    "text":"Request type {0} not supported by this service",
    "params":["audio"]
  }
]

// response
Content-Type: application/json

["La demande du type audio n'est pas supportée par ce service"]
```

Listing 3 Resolving a status message to a translated string

the corresponding values are given in the “params” array (as a zero-based index, so 0 refers to the first item, 1 to the second, etc.). The error message may also include an optional “detail” object providing more technical details about the error.

The standard ELG message codes have translations into a number of different languages (twelve at the time of writing, with more in the pipeline), and ELG provides a special API endpoint that accepts an array of errors and an ISO 639 language code, and returns an array of message strings in the requested language (if available) with all placeholders filled in. If the requested message code is not available in that language the endpoint falls back to English, and if the message code is not known at all then the “text” fallback from the original error is used instead.

Listing 3 shows an example of calling the “resolver” API; the `?lang=...` parameter specifies the desired language. If it is not provided then the resolver will respect any `Accept-Language` HTTP header on the request.[20] If no language is requested by the parameter or the header then messages will be returned in English by default.

Some long running services will return more meaningful progress updates as they work through their various stages of processing, and these updates will be passed back to the caller if they use the asynchronous API mode – requests to the polling

[20] For browser-based clients this will typically result in the messages being returned in the user’s preferred browsing language.

URL for a given job will return the latest progress update if the process is not yet complete. These updates are represented as i18n message objects in the same way as the errors and warnings described above, and they can be resolved to strings using the same resolver API endpoint.

4 Python SDK for Users

ELG provides many APIs to access the catalogue and search for specific resources, to download corpora hosted in ELG, to call services or many other uses (see Section 3). This provides ELG users with a lot of flexibility in the way they want to interact with the platform, however, the basic APIs are rather low level. For example, the search endpoint is paginated and returns only 20 results per call, which means that multiple API calls are needed to obtain more than 20 results. Similarly, calling a service via the public LT service API in the asynchronous mode requires multiple API calls to be made at the correct times and in the correct sequence to perform what is, from the user's perspective, a single action.

In order to simplify interactions with the platform, we developed a Python SDK that operates on top of the various ELG APIs and provides simple methods to easily interact with ELG and consume the resources in Python. We chose Python as the language for this first ELG SDK as it is probably the most widely-used programming language within the LT community.

The SDK is included in the ELG Pypi package which can be installed using the `pip` command familiar to any Python programmer. The basic SDK for consumer use is installed using `pip install elg`. The SDK provides access to most ELG functions through Python. It provides access to the catalogue with methods that allow users to search the catalogue and look for corpora, services, and organisations. The SDK enables users to call ELG-compatible services, and even to combine them using a simple pipeline mechanism.

4.1 Browsing the Catalogue

The SDK enables access to the ELG catalogue. It uses the same filters as the UI, i. e., we can filter for the type of resource or LT service, languages and licence; free text search can also be used. Listing 4 shows how to search for an English to French machine translation service. The SDK handles issues such as pagination automatically and returns the result as a list of entities, where each entity is a Python object that encapsulates the information about the respective ELG resource.

```python
from elg import Catalog

catalog = Catalog()

# Search and get the result as a list of Python objects
results = catalog.search(
    resource = "Tool/Service", # "Corpus", "Lexical/Conceptual
                               # resource" or "Language
                               # description"
    function = "Machine Translation", # only for "Tool/Service"
    languages = ["en", "fr"],  # string or list if multiple
                               # languages
)
```

Listing 4 Example code to use the ELG catalogue

4.2 Downloading a Resource

The Python SDK has a `Corpus` class that corresponds to a corpus or data set. It can be initialised using the identifier of the resource. If the resource is stored in ELG, it can be downloaded using the `download` method of the `Corpus` class. Listing 5 shows the most simple usage and parameters are available to choose the distribution or specify the download location for example.

```python
from elg import Corpus

corpus = Corpus.from_id(913) # initialise the Corpus using its ID
corpus.download()            # download corpus method
```

Listing 5 Example code to download an ELG corpus

4.3 Obtaining an Access Token

Some functions are restricted to authorised users of ELG (see Section 5). For the restricted APIs, an access token must be retrieved to identify the user behind the API call. It is possible to obtain a short-lived valid access token through the UI but this is not convenient for programmatic use.

To address this limitation, the Python SDK includes the `Authentication` class that interacts directly with the ELG OpenID Connect authentication service to obtain tokens, i. e., the access token to authenticate the API call and the refresh token which is used to refresh the access token when it expires.

```python
from elg import Authentication

auth = Authentication.init()
# here the user is asked to authenticate in the browser

auth = Authentication.init(scope="offline_access")
# here we are requesting an ``offline'' token that remains valid until
# revoked, as opposed to the usual token that requires re-authentication
# after 6 hours

auth.to_json("tokens.json") # export the tokens to a json file

auth = Authentication.from_json("tokens.json")
# creation of an Authentication object from the tokens in the json file
```

Listing 6 Example of code to obtain, store, and retrieve authentication tokens

Listing 6 shows an example usage of the `Authentication` class. During the process, the user has to authenticate using their browser and paste the resulting authorisation code back to the Python program. Once the `Authentication` object is initialised, it is possible to save the tokens in a `json` file and reuse them. Obtained tokens are by default valid for only six hours. It is possible to get tokens that are valid indefinitely by setting the `scope` parameter to `offline_access`.

4.4 Calling Language Technology Services

The `Service` class of the Python SDK corresponds to an ELG LT service, and can be initialised using the identifier of the service. As users need to be authenticated to use ELG services, a login step is necessary. Alternatively, it is possible to provide an `Authentication` object or a `json` file containing the tokens during the initialisation of the service, which allows the login step to be skipped. Various ways of authenticating during the service initialisation of a service are shown in Listing 7.

A service that is initialised in Python can be called easily (see Listing 8). The Python SDK handles the creation of the input message, any necessary refreshing of the access token, the communication with the REST API, etc.

When calling a service, the input request can be provided in various formats: a plain text, a path to a text or an audio file, or a `Request` object.[21] The result is a Python object that corresponds to one of the response messages (see Section 3.3).

[21] https://european-language-grid.readthedocs.io/en/stable/all/A1_PythonSDK/notebooks/Service.html#Usage

```python
from elg import Service

lt = Service.from_id(474) # login step necessary (unless tokens
    are cached) and the tokens will expire after 6 hours
lt = Service.from_id(474, scope="offline_access") # login step
    necessary (unless tokens are cached) and the tokens will
    never expire
lt = Service.from_id(474, auth_object=auth) # 'auth' is an
    Authentication object. No login step and the expiration of
    the tokens depends on the `auth` object
lt = Service.from_id(474, auth_file="tokens.json") # file
    containing existing tokens. No login step and the expiration
    of the tokens depends on the scope used to create them
```

Listing 7 Different ways of providing authentication during Service initialisation

```python
from elg import Service

lt = Service.from_id(474) # initialise LT service using its ID
result = lt("Nikola Tesla did not live in Berlin.") # run service
print(result)
```

Listing 8 Example code for calling an ELG service

5 User Authentication

While general exploration and search in the ELG catalogue is open to all, various other operations in ELG are restricted to certain users. For example, access to the LT service public API (via the Python SDK, curl or the "try out" UIs) requires the caller to be logged in so that the platform can enforce API call quotas to limit how much data can be processed by each user per day, following the ELG licensing strategy (see Section 6). Similarly, the submission of new resources and metadata records is limited to users who are registered as *providers*; administrative tasks are restricted to the technical ELG team.

Registering a regular user account is a simple self-service procedure. The registration form is available through the sign up/sign in icon in the top right corner of the catalogue page. All registered users are assigned the *consumer* role by default. To get *provider* status, users can submit a request through their profile page. All other roles are assigned internally by the ELG administrators.

ELG uses Keycloak[22], a user management, authentication and authorisation server based on the OAuth2 and OpenID Connect[23] standards. Keycloak supports both interactive authentication of users through the web UI, and programmatic access to the REST APIs using JSON Web Tokens. Users sign in to Keycloak, then they (or

[22] https://www.keycloak.org

[23] https://openid.net/connect/

the client tool they are using, such as the ELG Python SDK) can acquire an access token, which is a cryptographically signed "permit" that encodes their identity and permissions. API endpoints can verify the validity of the token by checking its signature, and then make access decisions based on the "claims" encoded in the token without needing to check every request directly with the authentication server.

The adoption of OpenID Connect opens up the possibility for third party applications to allow their own users to authenticate using ELG accounts, in the same way as many existing websites and applications support "sign in with Google" or "sign in with Facebook". The OpenID Connect specification allows this without compromising the protection of users' personal information. When a given user attempts to "log in with ELG" to a particular third party application for the first time, Keycloak requires the user to grant *explicit consent* before any of their data is shared with the provider, and that consent can be revoked at any time. At the time of writing the first proof of this concept is under development with one of the ELG pilot projects.

6 Licensing and Billing

ELG includes mechanisms that support the consumption of services and resources that are available without any restrictions in terms of commercial aspects. It supports the download of resources under the condition that they are offered free of charge with open access licences or with restrictive licences that require only user authentication and, optionally, accepting the licensing terms. Technical safeguards have been implemented to ensure that access to LRTs is granted in accordance with the above terms, for example, access to LRTs distributed with restricted licences is made available only to those users that fulfil the criteria specified in the licences. With regard to LT services, only the "try out" functionality is available and only for registered users. Each user has two independent daily quotas for the quantity of data processed, one for plain text and the other for binary (audio or image) data, to reflect the fact that binary formats generally require much more data than plain text.

In addition, we also designed and implemented the prototype of a billing module that will enable ELG to offer resources and services distributed with commercial licences. The module is based on the commercial platform Chargebee, which was selected because it fulfilled our requirements: it ensures security and includes various services, such as handling subscriptions, payments, pricing, taxes, emails, ensuring customer satisfaction and conformance to all EU and national laws, and offers several functionalities, such as checkout pages, self-service after the payment, cancellation, creating and managing subscription plans, subscription changes, etc. The integration of the external billing module is based on the interaction between the two platforms, ELG and Chargebee. Information about the pricing of a resource or service is formally encoded in the metadata record in ELG; administrative and execution costs may also be added and calculated on the ELG side. In the Chargebee catalogue we maintain a set of all monetised products and plans, and their prices.

The relationship between the ELG catalogue products and the Chargebee catalogue is not necessarily one-to-one; Chargebee can contain paid plans that allow the use of multiple products from the ELG catalogue, or the download of multiple resources. The relation between the two catalogues depends on the ELG business strategy. All transactions, subscription changes, logs, billing information, subscription data and similar information are stored on the Chargebee side, i. e., a database that is external to ELG. Any information needed from Chargebee can be synchronised through a webhook mechanism. For the ELG platform, this information includes the identity of the user who has performed an action through a subscription plan and/or a purchase, the action performed, the billing plan to which the user subscribed, etc. Chargebee sends this information via HTTPS POST to the ELG back end so that it can register changes in the ELG platform. The ELG back end monitors the user's quota usage and, taking into account the user's subscription plans from the Chargebee platform, decides whether to allow or block a request for running a service. A similar procedure is used for the download of a purchased resource.

7 Consumer-Related Functionalities in ELG and other Platforms

In this section we present platforms and catalogue-based systems that share features with ELG, with a special focus on functionalities for consumers.

7.1 Catalogue and Repository Functionalities

With regard to the presentation and organisation of the contents of such a digital catalogue of artefacts, the users of ELG can see all types of entities on the same page or go through quick links from the top menu to the subset that interests them. Offering such resource type-specific filtering functionalities is an approach adopted by many catalogues, for example, Hugging Face[24] has separate pages for models and datasets, Papers with Code[25] for datasets and benchmarks, some CLARIN centres distinguish between data resources and services (e. g., CLARIN-PL[26], etc.), the European AI on demand platform[27] maintains separate catalogues for AI assets, organisations, projects and educational resources. This approach is particularly useful for expert users with clear search objectives. In addition, distinguishing between separate resource types allows for the selection of different metadata elements and subgroupings of entries along the parameters most suitable to each type (e. g., grouping together services based on the tasks they perform or the degree of complexity

[24] https://huggingface.co

[25] https://paperswithcode.com

[26] https://clarin-pl.eu

[27] https://www.ai4europe.eu

of use, and datasets based on modality or language). On the other hand, the one-size-fits-all page has the benefit of allowing users to have an overview of resources and activities using the same set of filters. ELG combines the two approaches by providing quick links in the top menu and filters for the targeted pages.

With regard to search functionalities, free text search is the most popular one. In some cases, an autocomplete function (e. g., Hugging Face) is used while advanced queries are less used. Faceted search is also common, but in most cases with limited facets (e. g., European AI on Demand platform, Hugging Face, etc.). Search with programmatic modes through REST APIs is offered by many platforms on a limited set of metadata elements in the same way that ELG does.

With regard to the functionalities offered for hosted data resources, direct download of open access resources is common. A download link that can be used from outside the platform (e. g., through a command line mode, or as a URL link) is provided in most cases. The deployment of integrated services on hosted resources is a feature offered in only a few platforms (e. g., OpenMinTeD, clarin:el[28]). Machine Learning platforms, like Hugging Face, can feed hosted datasets into applications, but this is not among the objectives of the ELG platform.

7.2 Language Technology Service Execution

ELG's LT service execution functionality has been designed and implemented from scratch. Below, we compare this functionality with similar related infrastructures or frameworks and highlight the similarities and differences in various aspects, e. g., interchange format, trial/visualisation UIs and support of workflows.

The DKPro[29] family of tools and resources (Gurevych et al. 2007) consists of a growing number of projects addressing different NLP tasks and aspects, such as pre-processing, machine learning, and lexical resources. It offers a collection of tools wrapped as UIMA components (Unstructured Information Management Architecture)[30], i. e., the components implement the interfaces and specifications of the UIMA framework. A UIMA reader component should extend the `ResourceCollectionReaderBase` class and also implement the `getNext(CAS aJCas)` method. A processor must extend `JCasAnnotator_ImplBase` and, furthermore, implement `process(JCas aJCas)` and a writer extends `JCasFileWriter_ImplBase` and implements `process(JCas aJCas)`. A UIMA reader loads data from a text file and creates a Common Analysis System (CAS) object. A processor gets a CAS object, runs the wrapped NLP tool and adds the results to the CAS object. A writer gets a CAS object and serialises its content to a file in a specific format. UIMA is Java-based but it can be used to wrap non-Java tools as well. UIMA allows to programmatically define pipelines (workflows), i. e., chain a reader, various processors, a

[28] https://inventory.clarin.gr

[29] https://dkpro.github.io

[30] https://uima.apache.org

writer and run the pipeline locally; it does not run remote services as in the case of ELG. The DKPro components are interoperable because they all follow the DKPro typesystem[31], which defines which annotations can be added to a CAS object, which features an annotation can contain, how these are serialised etc. The typesystem is actually an ontology for annotations, how they are organised etc. The ELG JSON format does not follow a typesystem. Another difference with ELG is that a CAS object is serialised (by default) in XML Metadata Interchange (XMI) format[32], a standard for exchanging metadata information via XML; other formats are also supported. If the results of a DKPro pipeline are exported in an appropriate format (e. g., XMI) they can be loaded, visualised and even edited with the annotation tool INCEpTION[33] (Klie et al. 2018), which is not possible in the ELG trial UIs.

GATE[34] (Cunningham et al. 2013) is an open source toolkit capable of solving numerous text processing problem. The GATE framework is written in Java and similar to DKPro/UIMA. As with UIMA there are additional modules to support integration with non-Java tools. It allows creating, either via a UI builder or programatically, a pipeline of NLP tools for specific tasks. The completed pipeline can be saved in the XML "recipe" format XGAPP, which can, in turn, be loaded into the developer UI to process small numbers of documents and visualise the resulting annotations, run using a batch processing tool for larger scale processing, or packaged as a service on either the ELG or GATE's own GATE Cloud platform (see Chapter 7, Section 4.2, 140 ff.). Each GATE processing component gets as input a `GATE Document` which is enriched with annotations. Again, as in DKPro, GATE readers and writers load the data and write the processing results. A `GATE Document` is by default serialised to GATE XML, however, other formats are also supported. The annotations that are added in `GATE Document` do not follow a specific typesystem but follow some generic rules – each document has one or more sets of annotations, each set can contain annotations of many types, each annotation can have zero or more features, and while there is no *enforced* typesystem, all standard GATE components share a set of informal conventions for the types and features they use. This logic is very similar to the one adopted in ELG's JSON-based format. Contrary to ELG, the DKPro/UIMA and GATE tools are not dockerized (by default) and run as command line tools locally. Furthermore, the ELG services always process raw text while DKPro and UIMA components can also handle other formats such as PDF, and documents that have already been partially annotated.

GATE Cloud[35] (Tablan et al. 2013) is a platform very similar in spirit to ELG, but specifically built around the requirements of GATE-based text analysis tools. It was developed by the same team at the University of Sheffield that was responsible for the initial design of the ELG LT service execution layer and thus shares many of the same API design decisions. GATE Cloud offers a REST API accepting documents

[31] http://dkpro.github.io/dkpro-core/releases/1.8.0/docs/typesystem-reference.html
[32] https://www.omg.org/spec/XMI/2.5.1/About-XMI/
[33] https://inception-project.github.io
[34] https://gate.ac.uk
[35] https://cloud.gate.ac.uk

via HTTP post and returning annotations in the native JSON or XML formats of the GATE framework. GATE Cloud services process only text (not audio or other media types), but can accept formats such as XML, PDF (with machine-readable text) or Word documents as well as plain text. As well as the single document API, GATE Cloud also supports batch processing of larger amounts of data using on-demand processing capacity from Amazon Web Services. GATE Cloud services are defined as XGAPP "recipes" in the native GATE format, which are wrapped as Docker containers for the REST API or executed as-is by the batch processing engine. GATE Cloud has recently added support for other types of APIs such as image OCR (a service which has itself been integrated into the ELG platform).

The LAPPS Grid platform, as DKPro, is based on a typesystem, the LAPPS Web Service Exchange Vocabulary (Ide et al. 2016), "an ontology of terms for a core of linguistic objects and features exchanged among NLP tools that consume and produce linguistically annotated data. It is intended to be used for module description and input/output interchange to support service discovery, composition, and reuse in the natural language processing domain." In LAPPS Grid, as in ELG, tools are wrapped as web services, packaged as Docker images and exchange JSON messages. However, LAPPS Grid also offers workflows by using Galaxy, a workflow management system. Galaxy includes a visual editor for creating and parameterising workflows and an engine for executing these workflows. LAPPS Grid does *not* have a catalogue and each service is described with a limited set of metadata elements that are required for adding it to the Galaxy tool inventory. ELG was not designed to offer workflows, i. e., it does not include a workflow editor or a workflow execution engine. In addition, all services get as input raw text and they were not designed for playing the role of components in a workflow. However, some pipelines can be created by using external tools, e. g., the Python SDK and some code/adapters (Rehm et al. 2020; Moreno-Schneider et al. 2022). For example, using the ELG Python SDK, a Machine Translation service can be called, the result can extracted from the output JSON message and fed to an ELG NER service.

The OpenMinTeD execution service (Labropoulou et al. 2018) is also built on top of Galaxy. A large number of tools from the DKPro and GATE collections were ingested to OpenMinTeD. Several tools from other providers were also added. All tools were dockerized and are executed inside the container as command line tools, i. e., not as web services. An OpenMinTeD workflow is executed by running a series of Docker images (one after the other) in a cluster managed by Mesos[36], a framework similar to Kubernetes[37]. The workflow itself is created using the Galaxy editor. In OpenMinTeD no specific interchange format was enforced, the recommendation was to use the DKPro typesystem and XMI serialization. However, the GATE tools were using GATE XML format and several others were using their own custom format (e. g., based on JSON). In order to create a "mixed" workflow the creator had to combine the respective components with corresponding format adapters. If the results of

[36] https://mesos.apache.org

[37] https://kubernetes.io

the workflow were in XMI format, they could be visualised using WebAnno[38], a predecessor of INCEpTION.

The European AI on Demand platform[39] covers the whole European AI landscape rather than being restricted to LT or NLP. For example, computer vision is also included. The services are gRPC-based (not REST-based as in ELG) and are packaged as Docker images. The messages that they consume and produce are based on the ProtoBuf serialisation format[40] and no specific typesystem is used. The platform does not offer an execution environment. However, the worfklows that are created with the AI4EU Experiments editor[41], an editor similar to the one offered by Galaxy, are exported to a format that allows their execution in a Kubernetes cluster.

Hugging Face offers a large collection of Transformer-based models for computer vision, language processing, audio processing etc. Transformers are a specific type of neural networks (Vaswani et al. 2017) that have revolutionised machine learning since they achieve state of the art results in many tasks. Hugging Face allows training of Transformer-based models via the AutoNLP API[42], which is not free of charge. While we have performed initial experiments, ELG does not offer integrated model training. In Hugging Face, training as well model deployment is based on Amazon SageMaker, which is built on top of Docker. Hugging Face users can call a model via the trial UIs/widgets that are embedded in the respective page (as in ELG). For doing the same in a programmatic way, Hugging Face offers an inference REST API along with a Python client API[43]. Similar inference functionalities are offered through the ELG REST APIs and the Python SDK. Upon request, Hugging Face also offers an inference solution delivered as a container with the Transformer model for on-premise usage.[44] It can be used via a HTTP API (as in ELG). Finally, Hugging Face has developed a Python-based library (called "transformer") that allows to download a model and either fine-tune it in a specific task or use it for inference. Such functionality is not offered by the ELG Python SDK.

8 Conclusions

The ELG platform has fully achieved all objectives it had set for serving consumers. It allows consumers to browse through the whole ELG catalogue, already populated with more than 13,000 metadata records, apply faceted filtering and exploration, search for specific resources and services, download them (if hosted in ELG) and try out more than 800 functional services, both basic processing NLP services and

[38] https://webanno.github.io/webanno/
[39] https://www.ai4europe.eu
[40] https://developers.google.com/protocol-buffers
[41] https://aiexp.ai4europe.eu
[42] https://huggingface.co/autotrain
[43] https://api-inference.HuggingFace.co/docs/python/html/quicktour.html
[44] https://HuggingFace.co/infinity

end-to-end applications. Users can also access the directory of LT-developing companies and academic organisations, find organisations active in a specific LT area, and initiate collaborations with them. The links between LRTs, organisations and projects allows users to navigate between them and have an overview of the overall European LT landscape. Consumers can access all these functionalities through user-friendly web user interfaces, or in programmatic ways, using the public REST APIs and Python SDK.

References

Cunningham, Hamish, Valentin Tablan, Angus Roberts, and Kalina Bontcheva (2013). "Getting More Out of Biomedical Documents with GATE's Full Lifecycle Open Source Text Analytics". In: *PLOS Computational Biology* 9.2, pp. 1–16. DOI: 10.1371/journal.pcbi.1002854.

Data Citation Synthesis Group (2014). *Joint Declaration of Data Citation Principles – FORCE11*. Ed. by M. Martone. DOI: 10.25490/a97f-egyk. URL: https://doi.org/10.25490/a97f-egyk.

Gurevych, Iryna, Max Mühlhäuser, Christof Müller, Jürgen Steimle, Markus Weimer, and Torsten Zesch (2007). "Darmstadt Knowledge Processing Repository based on UIMA". In: *Proc. of the First Workshop on Unstructured Information Management Architecture (co-located with GLDV 2007)*. Tübingen, Germany, p. 89.

Ide, Nancy, Keith Suderman, Marc Verhagen, and James Pustejovsky (2016). "The Language Application Grid Web Service Exchange Vocabulary". In: *Worldwide Language Service Infrastructure*. Lecture Notes in Computer Science. Springer, pp. 18–32.

International Organization for Standardization (2007). *Codes for the representation of names of languages – Part 3: Alpha-3 code for comprehensive coverage of languages*. URL: https://www.iso.org/cms/render/live/en/sites/isoorg/contents/data/standard/03/95/39534.html.

Klie, Jan-Christoph, Michael Bugert, Beto Boullosa, Richard Eckart de Castilho, and Iryna Gurevych (2018). "The INCEpTION Platform: Machine-Assisted and Knowledge-Oriented Interactive Annotation". In: *Proceedings of the 27th International Conference on Computational Linguistics (COLING 2018): System Demonstrations*. Santa Fe, USA: ACL, pp. 5–9. URL: http://tubiblio.ulb.tu-darmstadt.de/106270/.

Labropoulou, Penny, Dimitris Galanis, Antonis Lempesis, Mark Greenwood, Petr Knoth, Richard Eckart de Castilho, Stavros Sachtouris, Byron Georgantopoulos, Stefania Martziou, Lucas Anastasiou, Katerina Gkirtzou, Natalia Manola, and Stelios Piperidis (2018). "OpenMinTeD: A Platform Facilitating Text Mining of Scholarly Content". In: *Proceedings of WOSP 2018 (co-located with LREC 2018)*. Miyazaki, Japan: ELRA, pp. 7–12. URL: http://lrec-conf.org/workshops/lrec2018/W24/pdf/13_W24.pdf.

Melnika, Julija, Andis Lagzdiņš, Uldis Siliņš, Raivis Skadins, and Andrejs Vasiļjevs (2019). *Deliverable D3.1 Requirements and Design Guidelines*. Project deliverable; EU project European Language Grid (ELG); Grant Agreement no. 825627 ELG. URL: https://www.european-language-grid.eu/wp-content/uploads/2021/02/ELG-Deliverable-D3.1-final.pdf.

Moreno-Schneider, Julián, Rémi Calizzano, Florian Kintzel, Georg Rehm, Dimitris Galanis, and Ian Roberts (2022). "Towards Practical Semantic Interoperability in NLP Platforms". In: *Proceedings of the 18th Joint ACL-ISO Workshop on Interoperable Semantic Annotation (ISA 2022; co-located with LREC 2022)*. Ed. by Harry Bunt. Marseille, France, pp. 118–126. URL: http://www.lrec-conf.org/proceedings/lrec2022/workshops/ISA-18/pdf/2022.isa18-1.16.pdf.

Phillips, Addison and Mark Davis (2009). *Tags for Identifying Languages*. Tech. rep. RFC 5646. Internet Engineering Task Force. URL: https://datatracker.ietf.org/doc/rfc5646.

Rehm, Georg, Dimitrios Galanis, Penny Labropoulou, Stelios Piperidis, Martin Welß, Ricardo Usbeck, Joachim Köhler, Miltos Deligiannis, Katerina Gkirtzou, Johannes Fischer, Christian Chiarcos, Nils Feldhus, Julián Moreno-Schneider, Florian Kintzel, Elena Montiel, Víctor Rodríguez Doncel, John P. McCrae, David Laqua, Irina Patricia Theile, Christian Dittmar, Kalina Bontcheva, Ian Roberts, Andrejs Vasiljevs, and Andis Lagzdiņš (2020). "Towards an Interoperable Ecosystem of AI and LT Platforms: A Roadmap for the Implementation of Different Levels of Interoperability". In: *Proc. of the 1st Int. Workshop on Language Technology Platforms (IWLTP 2020, co-located with LREC 2020)*. Ed. by Georg Rehm, Kalina Bontcheva, Khalid Choukri, Jan Hajic, Stelios Piperidis, and Andrejs Vasiljevs. Marseille, France, pp. 96–107. URL: https://www.aclweb.org/anthology/2020.iwltp-1.15.pdf.

Rehm, Georg, Stelios Piperidis, Kalina Bontcheva, Jan Hajic, Victoria Arranz, Andrejs Vasiļjevs, Gerhard Backfried, José Manuel Gómez Pérez, Ulrich Germann, Rémi Calizzano, Nils Feldhus, Stefanie Hegele, Florian Kintzel, Katrin Marheinecke, Julian Moreno-Schneider, Dimitris Galanis, Penny Labropoulou, Miltos Deligiannis, Katerina Gkirtzou, Athanasia Kolovou, Dimitris Gkoumas, Leon Voukoutis, Ian Roberts, Jana Hamrlová, Dusan Varis, Lukáš Kačena, Khalid Choukri, Valérie Mapelli, Mickaël Rigault, Jūlija Meļņika, Miro Janosik, Katja Prinz, Andres Garcia-Silva, Cristian Berrio, Ondrej Klejch, and Steve Renals (2021). "European Language Grid: A Joint Platform for the European Language Technology Community". In: *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations (EACL 2021)*. Kyiv, Ukraine: ACL, pp. 221–230. URL: https://www.aclweb.org/anthology/2021.eacl-demos.26.pdf.

Smith, Arfon M., Daniel S. Katz, and Kyle E. Niemeyer (2016). "Software citation principles". In: *PeerJ Computer Science* 2. URL: https://peerj.com/articles/cs-86.

Tablan, Valentin, Ian Roberts, Hamish Cunningham, and Kalina Bontcheva (2013). "GATECloud-.net: A Platform for large-scale, Open-Source Text Processing on the Cloud". In: *Philosophical Transactions of the Royal Society A: Math., Phys. and Eng. Sciences* 371.20120071.

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin (2017). "Attention is all you need". In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pp. 6000–6010.

Wilkinson, Mark D., Michel Dumontier, IJsbrand Jan Aalbersberg, Gabrielle Appleton, Myles Axton, Arie Baak, Niklas Blomberg, Jan-Willem Boiten, Luiz Bonino da Silva Santos, Philip E. Bourne, Jildau Bouwman, Anthony J. Brookes, Tim Clark, Mercè Crosas, Ingrid Dillo, Olivier Dumon, Scott Edmunds, Chris T. Evelo, Richard Finkers, Alejandra Gonzalez-Beltran, Alasdair J.G. Gray, Paul Groth, Carole Goble, Jeffrey S. Grethe, Jaap Heringa, Peter A.C 't Hoen, Rob Hooft, Tobias Kuhn, Ruben Kok, Joost Kok, Scott J. Lusher, Maryann E. Martone, Albert Mons, Abel L. Packer, Bengt Persson, Philippe Rocca-Serra, Marco Roos, Rene van Schaik, Susanna-Assunta Sansone, Erik Schultes, Thierry Sengstag, Ted Slater, George Strawn, Morris A. Swertz, Mark Thompson, Johan van der Lei, Erik van Mulligen, Jan Velterop, Andra Waagmeester, Peter Wittenburg, Katherine Wolstencroft, Jun Zhao, and Barend Mons (2016). "The FAIR Guiding Principles for Scientific Data Management and Stewardship". In: *Scientific Data* 3. DOI: 10.1038/sdata.2016.18. URL: http://www.nature.com/articles/sdata201618.

Wu, Mingfang, Fotis Psomopoulos, Siri Jodha Khalsa, and Anita de Waard (2019). "Data Discovery Paradigms: User Requirements and Recommendations for Data Repositories". In: *Data Science Journal* 18.1. URL: http://datascience.codata.org/articles/10.5334/dsj-2019-003/.

Chapter 4
Contributing to the European Language Grid as a Provider

Dimitris Galanis, Penny Labropoulou, Ian Roberts, Miltos Deligiannis, Leon Voukoutis, Katerina Gkirtzou, Rémi Calizzano, Athanasia Kolovou, Dimitris Gkoumas, and Stelios Piperidis

Abstract The ELG platform enables producers of language resources and language technology tools and services to upload, describe, share, and distribute their services and products as well as to describe their companies, academic organisations and projects. This chapter presents the functionalities offered through web-based user interfaces for describing LT resources or related entities with metadata and for managing their publication. It gives a detailed description of the options that providers of LT tools can exploit to integrate them into ELG as ready-to-deploy services and the tools that ELG offers in their support during the preparation, upload and integration phases. The tools and packaging recommendations for resources to be uploaded in ELG are also presented. The chapter concludes with a discussion of functionalities offered to providers by ELG and other related platforms.

1 Introduction

The European Language Grid platform (Rehm et al. 2021) offers various functionalities for providers of Language Resources and Technologies (LRTs) through which they can share their assets with the Language Technology (LT) community and interested clients, customers or users of these technologies. The minimum requirement is that they make them accessible (by uploading them to ELG or through another website) and describe them with a metadata record that complies with the ELG specifications (see Chapter 2), where they specify the access location and licensing con-

Dimitris Galanis · Penny Labropoulou · Miltos Deligiannis · Leon Voukoutis · Katerina Gkirtzou · Athanasia Kolovou · Dimitris Gkoumas · Stelios Piperidis
Institute for Language and Speech Processing, R. C. "Athena", Greece,
galanisd@athenarc.gr, penny@athenarc.gr, mdel@athenarc.gr, leon.voukoutis@athenarc.gr, katerina.gkirtzou@athenarc.gr, akolovou@athenarc.gr, dgkoumas@athenarc.gr, spip@athenarc.gr

Ian Roberts
University of Sheffield, UK, i.roberts@sheffield.ac.uk

Rémi Calizzano
Deutsches Forschungszentrum für Künstliche Intelligenz GmbH, Germany,
remi.calizzano@dfki.de

G. Rehm (ed.), *European Language Grid*, Cognitive Technologies,
https://doi.org/10.1007/978-3-031-17258-8_4

ditions under which they can be used. To take advantage of the advanced features of ELG, providers can also integrate LT tools as ready-to-deploy services, following the ELG specifications, or upload the resource itself, in which case it will be stored and preserved according to the Data Management Plan (see Chapter 8) and made readily available to LRT consumers. Furthermore, descriptions of organisations that are active in the LT area can be added in order to promote their activities and products. Descriptions of projects that have been funded in the broader LT area can also be included in the ELG catalogue. LRTs, organisations that have provided or created them and projects that have contributed to their funding are linked together.

Detailed documentation is provided and a suite of helper tools have been developed aiming to make the contribution and integration of all entities briefly sketched above as simple as possible, taking into account the technical expertise and preferences of users. In ELG, the provision and management of catalogue entries is supported through web user interfaces (UIs) and REST application programming interfaces (APIs). Section 2 describes the steps a provider must take to contribute entries to the catalogue, and the tools provided by ELG to support this process. The ELG catalogue intends to be a reliable source for resources that can be accessed and (re-)used by commercial and non-commercial, research and public organisations as well as individuals. For this purpose, management and curation policies and processes for the metadata, data and services included in ELG have been set up, albeit with variations depending on the source and type of contribution. Only authorised and authenticated individuals can add LRTs in ELG; the registration and assignment of the "provider" user role is a simple process for all interested users (see Chapter 3). In addition, all entries go through a formal publication life cycle (see Chapter 2). Before being published in the catalogue, added metadata records are validated by the ELG core team (Section 3). Section 4 looks into the requirements for the different types of resources and entities in ELG, either integrated in ELG or available remotely and added to ELG as metadata records only. Further technical specifications are set for LT services that are intended to be deployed through the ELG cloud infrastructure, and for data resources hosted in ELG. Before being published in ELG, these resources go through a process that aims to ensure their technical validity and, for services, to set up the required environment for their deployment. Section 5 presents similar platforms and infrastructures and discusses the approach and tools they offer for providers of LRTs, in analogy to the comparison made for the platform functionalities from the point of view of consumers in Chapter 3.

2 Adding Resources to the ELG Platform

LRT providers come from a variety of backgrounds, some within Language Technology fields such as NLP or Computational Linguistics, and others from neighbouring fields such as Digital Humanities. Different providers have different levels of technical knowledge and familiarity with formal metadata descriptions, so ELG attempts to offer an integrated environment suitable for both expert and non-expert users. The functions exposed for registering and managing catalogue entries and their accompanying data files are designed to be user-friendly while still offering advanced features to users with the relevant skills.

All metadata records must comply with the ELG metadata schema (Labropoulou et al. 2020). The schema offers a rich set of metadata elements for each type of LRT or entity (organisation, project) to be added. Individual elements are either *mandatory*, *recommended* or *optional*, depending on the record type. Providers can add entries with only the mandatory elements, although they are also encouraged to add the recommended ones. See Chapter 2 for more details.

2.1 Creating Metadata Records

Providers can add records in one of two ways: either by creating and uploading XML files compliant to the ELG schema (Section 2.1.1), or by using the interactive editor offered by ELG (Section 2.1.2). In practice many users will adopt a combination of the two approaches, for example, a provider who wishes to submit many similar records (such as MT services based on the same underlying engine but with models for different language pairs) may create their first record using the editor, export it as XML, and use this file as a template to generate the remaining records.

2.1.1 Creation and Upload of Metadata Files

This first option is probably more appealing to expert and technical users, especially those that wish to register multiple related records or produce frequently updated versions of LRTs registered in ELG. To facilitate the process of adding records, pre-filled metadata templates and examples (with the mandatory and recommended elements) are available in the ELG GitLab repository[1]. As mentioned above, any existing metadata record can be exported from ELG as XML to be used as a template.

A REST endpoint for metadata validation of single files or zipped archives of XML files is publicly available and offered for providers that want to validate their metadata files and ensure they comply with the ELG schema before uploading them to the platform.[2] The XSD validator checks that all mandatory elements are filled in and that filled-in values are consistent with the data type declared for the elements – for example, if elements take values from controlled vocabularies or should follow a specific pattern – and returns the results in JSON form.

Users can upload their metadata records through the provider's grid (see Section 2.3) as single files or in batch mode. The import step includes additional validation rules, which check the syntactic and, to a certain extent, semantic integrity of the record. For example, checks are performed for metadata elements that depend

[1] https://gitlab.com/european-language-grid/platform/ELG-SHARE-schema

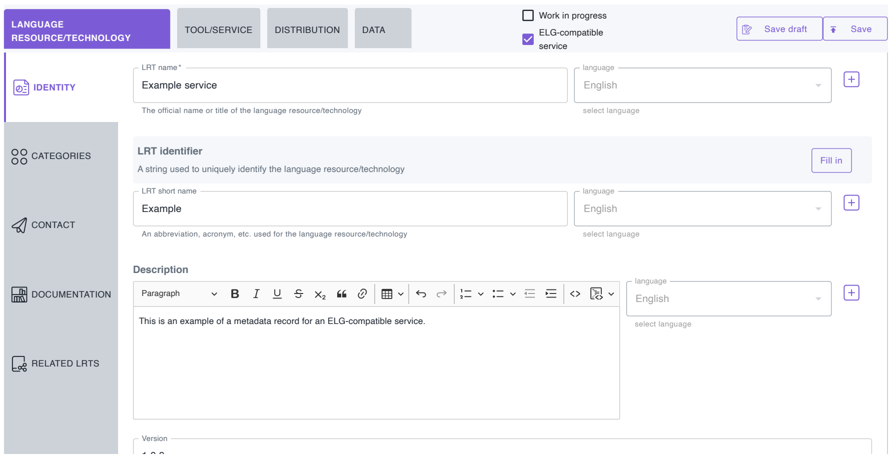
[2] https://live.european-language-grid.eu/catalogue/#/validate-xml

on the presence or value of other elements (e. g., the element “multilinguality type” which is mandatory for bilingual and multilingual resources), or for duplicate values (e. g., the same “language” value used twice). Validation errors are reported to the user for correction. If the file is valid, it is imported to the platform and the provider can perform further edits with the editor or submit it for publication in accordance with the publication life cycle (see Chapter 2).

2.1.2 Metadata Editor

The editor can be accessed through the provider’s grid (see Section 2.3). It supports users in creating new metadata records, as well as editing and updating existing ones. The editor includes the mandatory and recommended fields of the ELG schema. Chapter 2 provides a summary of all mandatory metadata elements.

The editor has been designed with non-expert users in mind, and intends to hide the richness of the ELG schema. For this reason, we offer a full-fledged UI with metadata elements grouped into semantically coherent sets and layered along horizontal and vertical tabs, following the ELG conceptual structure. Different editor forms with the same look and feel have been implemented for each resource or entity type. Figure 1 shows the editor for tools/services; the horizontal tabs correspond to the main classes of the schema – in this case, LRT, tool/service and distribution – and the vertical tabs to categories of elements within that main section. The figure shows the LRT horizontal tab, whose options include “identity” (identification metadata such as the resource name, long description, and name of the creator responsible for the record), “categories” (classification elements such as keywords and subject domain), and “documentation” (links to publications, user manuals, or other documents describing the resource).



☐ Work in progress
✓ ELG-compatible service

LRT name * : Example service
The official name or title of the language resource/technology
language: English
select language

LRT identifier
A string used to uniquely identify the language resource/technology

LRT short name: Example
An abbreviation, acronym, etc. used for the language resource/technology
language: English
select language

Description
This is an example of a metadata record for an ELG-compatible service.
language: English
select language

Version

Fig. 1 ELG metadata editor

The editor guides the user to fill in at least all of the mandatory elements with appropriate values. Help tips and examples are available for metadata elements, and different editing controls are used for elements depending on their data type. For instance, the elements of controlled vocabularies are shown using dropdown lists. For vocabularies with many values (e. g., languages, service functions, etc.), we use a combination of dropdown lists with suggested values as the user types in the text.

The combination of dropdown lists and dynamically suggesting values is also applied to improve normalisation. For example, some elements such as keywords allow free text entry, however as the user types, a popup suggests matching values that have previously been used for the same element in other records, "nudging" the user to choose identical values instead of slight variations. The same lookup mechanism, of suggesting values from those already imported in the catalogue, is used for reducing the chance for duplicates of related entities such as agents, projects, documents, licences, and other resources.[3] For such entities, the ELG schema requires a set of minimal information, a name/title, and, optionally, an identifier and metadata elements that could uniquely distinguish it from similar entities (e. g., email for persons, website for organisations, a URL with the text for licences, etc.). Thus, when adding related entities through the editor, users type in a name/title, and are shown matching entries (if any) to select from; if not, they are prompted to fill in the required elements mentioned above. The same set of metadata elements is also used at the import of metadata records to uniquely identify the related entities.

Through the editor, providers have the option of saving incomplete metadata records ("draft"), for which only the data type of the metadata elements is validated (e. g., that they have entered a valid URL). When they decide to properly save the metadata record, we validate the entry using the yup library[4], implementing at least the same rules used at the import of metadata files. In case of errors, messages describe the error and location where it occurred (see Figure 2); clicking on the error, users are forwarded to its location.

2.2 Uploading and Managing Data Files

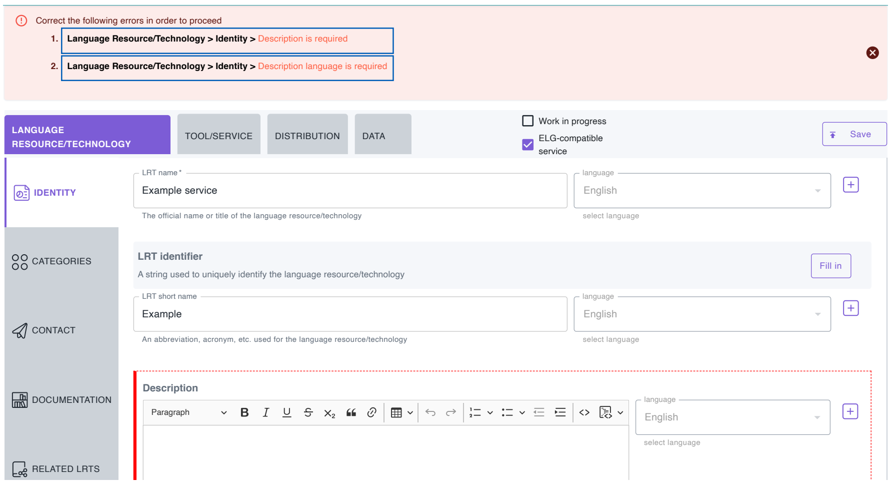
Data files, i. e., the physical files that contain the contents of a resource, must be uploaded as a ZIP file. Section 4.2.2 presents recommendations for the packaging of data resources, especially for those that can be split into subsets.

Providers can upload data files as a first step when they upload an XML file[5], or during the editing process with the editor. The editor includes a tab entitled "Data" (Figure 3) through which users can manage the files (upload, replace and delete).

[3] This is a well-known issue across catalogues; the adoption of unique persistent identifiers is recommended to resolve it, but not all entities are assigned such a unique identifier or it may not be known to the provider that submits the metadata record.

[4] https://github.com/jquense/yup

[5] At the time of writing, the upload of data files during the batch import of XML metadata records is not supported.



Correct the following errors in order to proceed

1. Language Resource/Technology > Identity > Description is required
2. Language Resource/Technology > Identity > Description language is required

Work in progress: ☐
ELG-compatible service: ✓

LRT name *: Example service — language: English
The official name or title of the language resource/technology — select language

LRT identifier
A string used to uniquely identify the language resource/technology

LRT short name: Example — language: English
An abbreviation, acronym, etc. used for the language resource/technology — select language

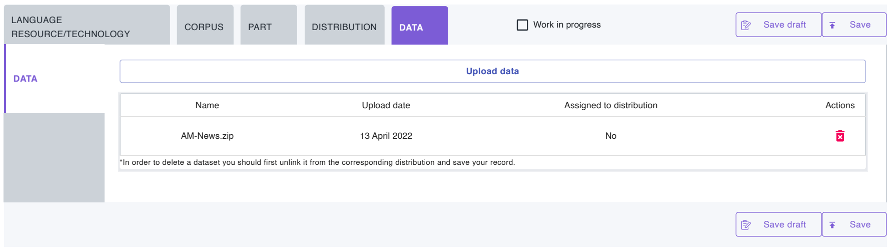
Description — language: English
select language

Fig. 2 ELG metadata editor with error messages

A resource may be available in a range of distributable forms (“distributions”), for example, in different file formats (e. g., as PDF, XML or TXT files). ELG supports the upload of multiple data files for the same resource. For this reason, when users upload more than one package of data files, they are prompted to associate each package with the respective distribution (i. e., the one that includes the metadata that describe the size and format of the particular set of files). This action is performed by selecting the specific package on the “distribution” tab.

2.3 Managing Catalogue Entries

The ELG platform presents users that have the “provider” role set with a “grid” (dashboard), through which they can access and manage the catalogue items they have created, as well as create new items (Figure 4). Since every provider is by definition

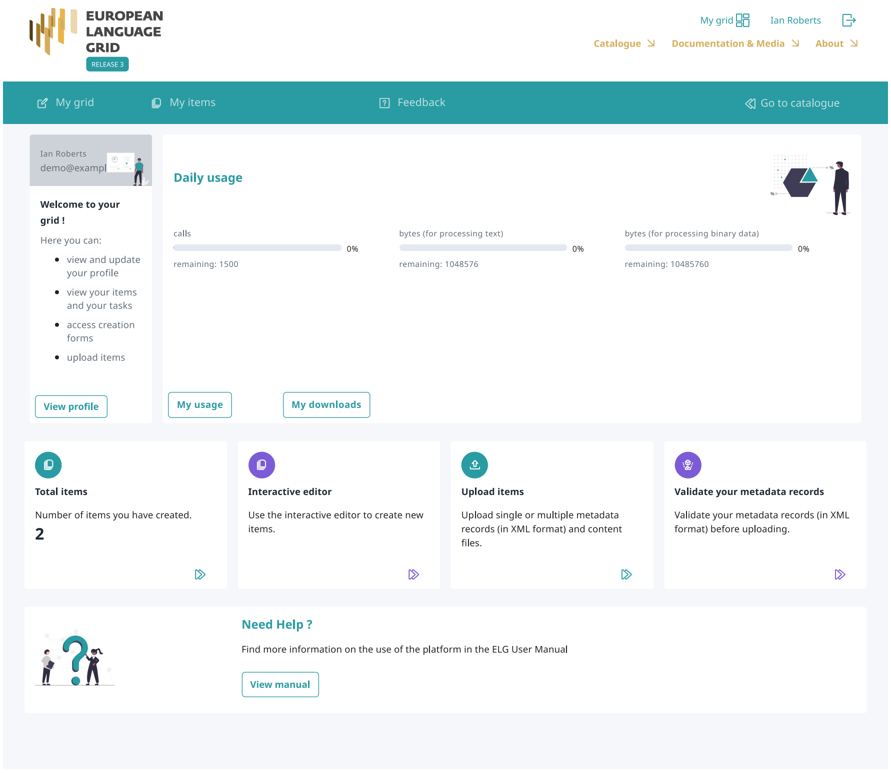


Work in progress: ☐

Name	Upload date	Assigned to distribution	Actions
AM-News.zip	13 April 2022	No	🗑

*In order to delete a dataset you should first unlink it from the corresponding distribution and save your record.

Fig. 3 ELG metadata editor – “data” tab for uploading data files


EUROPEAN LANGUAGE GRID
RELEASE 3

My grid Ian Roberts
Catalogue Documentation & Media About

My grid My items Feedback Go to catalogue

Ian Roberts
demo@exampl

Welcome to your grid !

Here you can:

- view and update your profile
- view your items and your tasks
- access creation forms
- upload items

View profile

Daily usage

calls	bytes (for processing text)	bytes (for processing binary data)
0%	0%	0%
remaining: 1500	remaining: 1048576	remaining: 10485760

My usage My downloads

Total items
Number of items you have created.
2

Interactive editor
Use the interactive editor to create new items.

Upload items
Upload single or multiple metadata records (in XML format) and content files.

Validate your metadata records
Validate your metadata records (in XML format) before uploading.

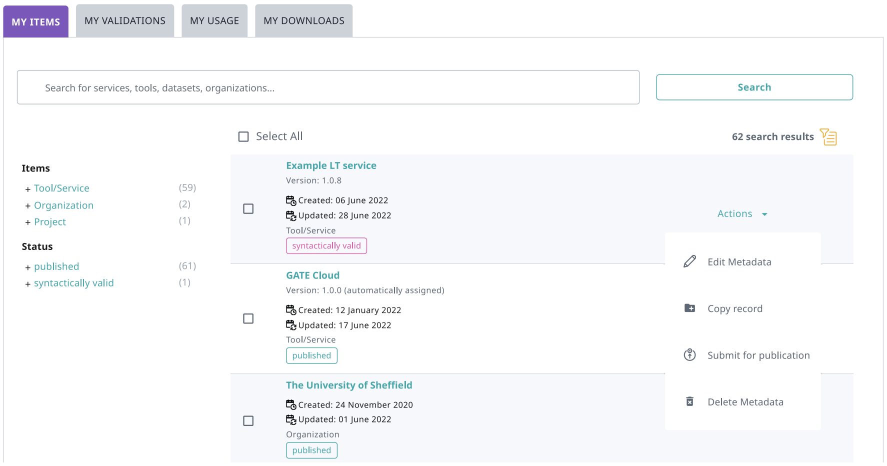
Need Help ?
Find more information on the use of the platform in the ELG User Manual
View manual

Fig. 4 Provider's grid (see Figure 6 in Chapter 3, p. 48, for the Consumer's grid)

also a consumer, the provider's dashboard is an extension of the consumer's dashboard shown in Chapter 3, adding a counter of the number of records this user has created and links to the editor, XML upload, and XML validator tools.

Users can manage the metadata records they have created through a dedicated page ("My items", Figure 5), and, in accordance with their user rights and the publication status of the record, perform the following actions: edit a metadata record, submit it for publication, create a new version of a published record, copy a metadata record (in order to use it as a model and create a similar record), delete a metadata record that has not yet been published, and request the unpublication of one of their records.[6] The "My items" page is a focused version of the catalogue, this time filtering records according to each user's role. This page also implements browse and search functionalities like the main catalogue page.

[6] Records cannot be completely deleted after publication except in exceptional circumstances, and then only by request to the ELG administrators.



Example LT service
Version: 1.0.8
Created: 06 June 2022
Updated: 28 June 2022
Tool/Service
syntactically valid

GATE Cloud
Version: 1.0.0 (automatically assigned)
Created: 12 January 2022
Updated: 17 June 2022
Tool/Service
published

The University of Sheffield
Created: 24 November 2020
Updated: 01 June 2022
Organization
published

Fig. 5 "My items" page

3 Validating and Publishing Metadata Records

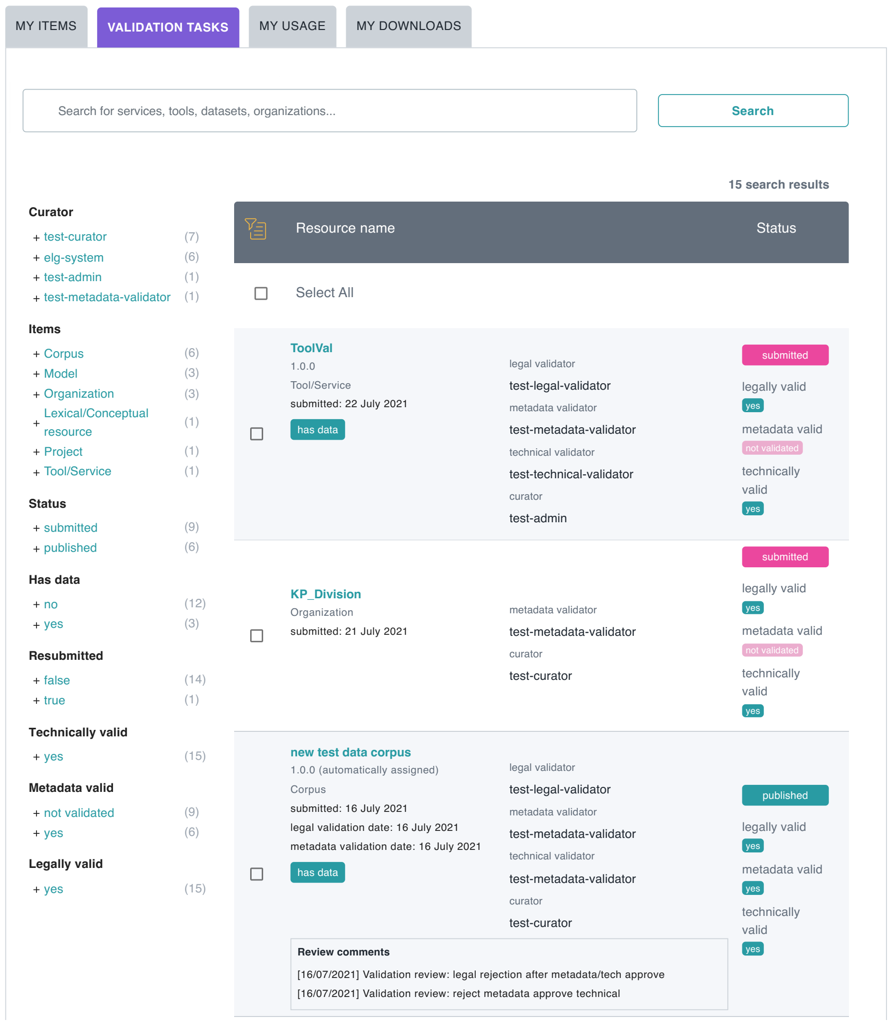
Metadata records added by individuals[7] enter a validation process, as specified in the ELG publication life cycle (see Chapter 2), before they are published in the catalogue: we perform technical/metadata and legal validation for ELG-compatible services and resources with uploaded data files, and validation at the metadata level only for all other metadata records. ELG-compatible services also go through a set of actions required for the registration of the service in the ELG platform (see Section 4.1.8).

Validators have access to the metadata records that have been assigned to them through the "validator's grid", and more specifically the "My validations" page (Figure 6). The validation form includes fields in which the validator can add internal comments (visible only to the other validators), and in the case of rejected records, a field for noting the reasons and suggested changes that are communicated to the provider for corrections. Providers can go through the changes and resubmit the record, which initiates a new round of validation, until final approval. When the metadata record has been approved by the responsible validator or validators, it is automatically made visible in the public catalogue.

4 Entity-Type Specific Requirements

There are several technical requirements that need to be met for LT services (Section 4.1) or resources (Section 4.2) to be deployed through or hosted in ELG successfully. We also present the requirements for metadata-only resources (Section 4.3).

[7] For harvesting and batch import functionalities from other catalogues, see Chapter 6.


MY ITEMS | VALIDATION TASKS | MY USAGE | MY DOWNLOADS

Search for services, tools, datasets, organizations... | Search

15 search results

Curator
+ test-curator (7)
+ elg-system (6)
+ test-admin (1)
+ test-metadata-validator (1)

Items
+ Corpus (6)
+ Model (3)
+ Organization (3)
+ Lexical/Conceptual resource (1)
+ Project (1)
+ Tool/Service (1)

Status
+ submitted (9)
+ published (6)

Has data
+ no (12)
+ yes (3)

Resubmitted
+ false (14)
+ true (1)

Technically valid
+ yes (15)

Metadata valid
+ not validated (9)
+ yes (6)

Legally valid
+ yes (15)

Resource name | Status

Select All

ToolVal
1.0.0
Tool/Service
submitted: 22 July 2021
has data

legal validator
test-legal-validator
metadata validator
test-metadata-validator
technical validator
test-technical-validator
curator
test-admin

submitted
legally valid yes
metadata valid not validated
technically valid yes

KP_Division
Organization
submitted: 21 July 2021

metadata validator
test-metadata-validator
curator
test-curator

submitted
legally valid yes
metadata valid not validated
technically valid yes

new test data corpus
1.0.0 (automatically assigned)
Corpus
submitted: 16 July 2021
legal validation date: 16 July 2021
metadata validation date: 16 July 2021
has data

legal validator
test-legal-validator
metadata validator
test-metadata-validator
technical validator
test-metadata-validator
curator
test-curator

published
legally valid yes
metadata valid yes
technically valid yes

Review comments
[16/07/2021] Validation review: legal rejection after metadata/tech approve
[16/07/2021] Validation review: reject metadata approve technical

Fig. 6 "My validations" page

4.1 ELG-compatible Services

A service is ELG-compatible if it is packaged in a Docker image and follows the ELG LT internal API, i. e., the service consumes and produces messages in the ELG-specified format, as defined in Section 4.1.1 below. When a provider adds a tool or service to ELG either using XML metadata upload or through the metadata editor, they are asked if the service will actually be integrated in ELG, so that conformance to our specifications can be monitored.

4.1.1 Internal LT API Specification

The ELG internal LT API is closely related to the public API described in Chapter 3. The public API is a simplified derivative of the internal API. While both the internal and public APIs make use of the same JSON messages for input and output, the internal API is designed strictly around a single HTTP request-response transaction for each processing task, rather than the multi-step asynchronous mode supported by the public API.

For the internal API, services that accept text receive their requests as JSON, while services that process binary audio or image data receive a MIME "multipart/-form-data" request with the metadata in JSON and the binary data as the relevant audio or image MIME type. The endpoint must return the appropriate JSON response message depending on its function (standoff "annotations", classifications, audio, or new "texts" – which could be a single text, a series of sentences, a list of alternative translations, etc.). Examples include:

- *Information extraction (IE) services for text* accept a "text" request and return an "annotations" response; i. e., annotations whose position is described in terms of zero-based character offsets. Such services include tokenisers, sentence splitters, sentiment analysers, named entity recognisers, dependency parsers, etc.
- *Text classification services* accept a "text" request and return a "classification" response with the classes that have been assigned to the whole input text by the service. Examples are language identifiers, text-level sentiment classifiers etc.
- *Machine translation services* receive a "text" request and generate a new text or list of alternatives returned in a "texts" message. Services such as summarisation would use a similar format.
- *Information extraction services from speech* take "audio" requests and return the same standoff annotations as IE-from-text, but in this case the annotations are time segments in the audio stream, e. g., keyword spotting for audio files.
- *Speech recognition services* take "audio" requests and return a text transcription or a choice of n-best transcriptions, encoded as a "texts" message.
- *Text-to-speech services* take "text" messages and return "audio" messages, which can either include the returned audio inline as base64-encoded data, or as a URL reference to audio which has been uploaded to the temporary storage helper service (see Section 4.1.2).
- *Optical character recognition services* take "image" requests and return the extracted text as a "texts" response.
- *Image classification services* take "image" requests and return "classification" responses.

The formats of the input and output messages are generic and can be easily reused for integrating new types or classes of services. For example, Speech-to-Text services, such as a speech summariser that would consume an "audio" request and return a "texts" response in the same way as a pure speech recogniser, can easily be added. Other examples can be found in Chapter 7.

Detailed, up-to-date guidance on the process of integrating an LT service and selecting the most appropriate integration option can be found in the ELG documentation[8]; more information is provided in Section 4.1.3.

As described in Chapter 3, error, warning and progress report messages are represented as structured objects with a message *code*, representing a message that can be localised into many languages. The ELG team provides a set of standard message codes for common messages, and maintains their translations, but service providers who use their own custom messages are welcome to contribute their own localisations for integration into the public message resolver by contacting the ELG team.

Services that take a long time to process data have the option of returning a series of "progress" messages prior to generating the final response using the standard HTTP "server-sent events" format.[9]

4.1.2 Helper Services

ELG provides certain helper services that can be called at fixed URLs by LT service containers if they run within the platform. Notably, ELG provides a temporary storage helper which LT services can use in order to return data that does not naturally map on to the standard JSON-based response formats. This helper allows an LT service to store arbitrary blobs of binary data on a short-term basis (for any time from ten seconds up to 24 hours), and receive a randomly generated URL that can be included in the response JSON, and which the caller can retrieve up until its expiry time. Typical uses for this service include text-to-speech services that need to return larger chunks of audio data, or services that visualise structures such as parse trees in a binary image format. This is discussed further in the context of the Text2TCS service in Chapter 7, Section 5.1, p. 144 ff.

4.1.3 Integration Requirements and Options

The requirements for integrating an LT tool or service into ELG are as follows.

Expose an ELG-compatible endpoint: The provider needs to make sure that the LT tool or service to be integrated into ELG exposes an HTTP endpoint, i. e., either such an endpoint already exists or it needs to be implemented. The corresponding endpoint application must consume HTTP requests that follow the ELG JSON format, call the included or underlying LT tool and produce responses again in the ELG JSON format as specified in the the ELG LT internal API (Section 4.1.1). Developers working in Python or Java, Groovy, Kotlin, or other JVM-based languages, can make use of helper libraries provided by the ELG team to handle much of the boilerplate code for creating the HTTP listener, parsing and

[8] https://european-language-grid.readthedocs.io/en/stable/all/3_Contributing/Service.html

[9] https://html.spec.whatwg.org/multipage/server-sent-events.html#server-sent-events

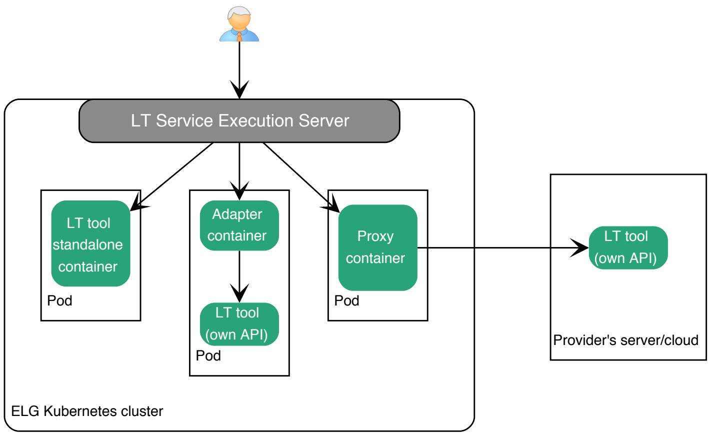


Fig. 7 Integration options

producing the JSON messages, etc., so that the provider can concentrate on their own business logic (see Sections 4.1.6 and 4.1.5 for more details).

Provide the application in the form of a Docker image: The whole application must be packaged as a container image using Docker or similar tools, and uploaded to a Docker registry, such as GitLab[10], DockerHub[11] or Azure Container Registry[12]. More than one image might be needed for one service, depending on how the service is made available. From the three options described in Fig. 7, providers can pick the one that best fits their needs.

- *LT tool packaged in one standalone image:* One image is created that contains the application that exposes the ELG-compatible endpoint and the actual LT tool. This is the most common approach when wrapping tools that are callable as libraries from custom code, such as Python machine learning models.
- *LT tool running remotely outside the ELG infrastructure:* In this case, one *proxy* image is created that exposes one (or more) ELG-compatible endpoints; the proxy container communicates with the actual LT service that runs outside the ELG infrastructure.
- *LT tool requiring an adapter:* This is a compromise between the standalone and remote approaches. A tool that is available as a Docker image but whose API is not natively ELG-compatible can be run alongside a separate ELG-compatible *adapter* image as a single pod in the ELG infrastructure. The adapter receives ELG API requests, communicates with the tool's native API in the pod, and translates the responses back to ELG format.

[10] https://gitlab.com
[11] https://hub.docker.com
[12] https://azure.microsoft.com/en-us/services/container-registry/

```
 1 # Base image.
 2 FROM openjdk:8-jdk-alpine
 3
 4 # SET TARGET DIRECTORY
 5 ENV TARGETDIR /elg/
 6 # This is required for wait.sh
 7 RUN apk update && apk add bash
 8
 9 # Install tini and create unprivileged user
10 RUN apk add --no-cache tini && \
11       addgroup --gid 1001 "elg" && \
12       adduser --disabled-password --gecos "ELG User,,," \
13       --home /elg --ingroup elg --no-create-home --uid 1001 elg
14
15 # Create target directory
16 RUN install -d -o elg -g elg $TARGETDIR
17 # Copy everything to target directory
18 COPY --chown=elg:elg dockerCmd ${TARGETDIR}dockerCmd
19 # Copy/Rename server app jar.
20 ADD --chown=elg:elg  /elg-ilsp-lt-services-rest-simple-0.0.1-
      SNAPSHOT-exec.jar ${TARGETDIR}dockerCmd/app.jar
21
22 # Set working directory
23 USER elg:elg
24 WORKDIR ${TARGETDIR}dockerCmd
25
26 # Make sure script can be executed
27 RUN chmod +rx ./wait.sh
28
29 # The command that is run when the container starts
30 ENTRYPOINT ["sh", "runInContainer.sh"]
```

Listing 1 Example of a dockerfile for an integrated ELG LT service

4.1.4 Creation of Docker Images

The Docker image of an application contains the code of the tool and all dependencies required to run it, e. g., the operating system, frameworks, settings, configuration files and libraries etc. Containers are instantiations of images and can be thought of as lightweight virtual machines.

The process of packaging a service as a Docker image involves creating a dockerfile that describes the build process, running that build, and pushing, i. e., copying the resulting image to a Docker registry that is accessible to the ELG infrastructure. An example dockerfile is shown in Listing 1. The most important parts are:

- Line 2 states that an image containing a lightweight Linux-based operating system that includes Java programming language will be used as the base.
- Line 20 adds the Java-based application (.jar file) that exposes an ELG-compliant LT service to the image (see Section 4.1.5 for more details).

```
# Login to Gitlab container registry
$ docker login registry.gitlab.com

# Build the image and tag it with the name registry.gitlab.com/
    ilsp-nlpli-elg/elg-ilsp-lt-services and a version number
$ docker build -t registry.gitlab.com/ilsp-nlpli-elg/elg-ilsp-lt-
    services:1.0.0 .

# Push the image to the container registry
$ docker push registry.gitlab.com/ilsp-nlpli-elg/elg-ilsp-lt-
    services:1.0.0
```

Listing 2 Example sequence of commands to build and push a Docker image to a registry

- Line 30 specifies the script (.sh) that is run when a container is created from this image; this script starts the Java application.

A simple and robust way to build and store the image of a service in a registry is to put the service code into a source code repository such as GitHub[13] or GitLab, and then to use the repository's continuous integration (CI) mechanism. There are various examples of services built like this, i. e., using GitLab CI, in the ELG GitLab space.[14] Gitlab CI is triggered immediately after a commit on the repository or on demand and runs the build process specified in `.gitlab-ci.yml`.

An image can also be built and stored by running a set of commands locally. This option is helpful because CI services are often restricted, e. g., Gitlab has monthly quotas. In this case, users must first download the source code to a local folder (including the dockerfile), and then run a sequence of commands similar to Listing 2.

Some languages and build systems provide alternatives for building Docker images that do not require developers to write their own dockerfile, or to use Docker at all. For example, Java services based on the Micronaut[15] helper described below can use the Micronaut built-in `dockerPush` or `dockerPushNative` gradle tasks to build and push an image in one step using an automatically generated dockerfile, or Google Jib[16], which is designed specifically around the needs of Java applications and produces intelligently layered images that make more efficient use of space in the container registry. Additional files such as models can also be included.

To be deployed in ELG, a Docker image must meet the following requirements:

- It must be built for the `amd64` architecture (also known as `x86_64`); multi-architecture images may be appreciated by users who want to run the service on their own hardware, but ELG itself runs on `amd64`.
- It must be compatible with the Broadwell micro-architecture, which supports SSE4.2, AVX and AVX2 but *not* AVX512 instructions.

[13] https://github.com

[14] https://gitlab.com/european-language-grid

[15] https://micronaut.io

[16] https://github.com/GoogleContainerTools/jib

- The container must run in *at most* 6GB of RAM, but the smaller its footprint the better. By default, containers are limited to 512MB RAM; if the container requires more memory, this must be specified in the metadata record (using `additionalHWRequirements`). Services requiring more than 6GB are approved only in exceptional cases.
- It must be tagged with an explicit version number such as `:1.0.0`, not the implicit `:latest` tag which typically changes over time.
- The network socket on which the container listens for HTTP requests must bind to all the container's IP addresses (typically by using `0.0.0.0`). Some HTTP libraries only listen on the local loopback `127.0.0.1` by default, which will not be sufficient in ELG.
- Ideally the container should run without needing outgoing network connections to locations outside the hosting cluster. In particular, any model files must be cached within the image at build time, not downloaded at runtime from a repository such as Hugging Face. If outgoing network access is *required*, the target IP address ranges must be specified.

It is recommended for the service to only start listening once it is fully initialised and ready to start handling requests. If this is not possible (e. g., if the code requires some asynchronous initialisation process and the library used opens its sockets before that process is complete), then a separate "readiness" endpoint should also be provided at a separate URL path from the main service endpoint (typically `/elg-ready`) that returns the response code 503 ("service unavailable") if the service is not yet initialised, and 200 or 204 once it is ready to handle requests.

Sections 4.1.5 and 4.1.6 present Java- and Python-based libraries for easily creating an application that offers an ELG-compatible service. Some of these include utilities for creating the Docker image in which the service will be packaged.

4.1.5 Helper Libraries for Java

For LT service developers working in Java or other Java Virtual Machine (JVM) languages such as Groovy[17] or Kotlin[18], ELG provides helper libraries for two popular frameworks, Spring Boot[19] and Micronaut[20]. The programming style is similar in both cases, though Micronaut is better optimised towards creating smaller, lighter images with faster startup times, so if the service implementation does not already have a dependency on Spring, Micronaut is the recommended option. Both libraries depend on a common bindings library[21] of Java model classes that represent the various JSON message structures in a more Java-native way.

[17] https://groovy-lang.org

[18] https://kotlinlang.org

[19] https://spring.io/projects/spring-boot

[20] https://micronaut.io

[21] https://javadoc.io/doc/eu.european-language-grid/elg-java-bindings

An ELG-compatible LT service can be built in three steps[22] using Micronaut:

1. Create a blank Micronaut application using the Micronaut Launch tool.[23]
2. Add the ELG helper as a dependency, which is published to the central repository – for Gradle this means

```
implementation("eu.european-language-grid:lt-service-
    micronaut:1.0.0")
```

3. Create a controller that extends `LTService` (for services that process text-based requests) or `BinaryLTService` (for services that process requests with binary content) and implement the relevant handle or handleSync method.

The process[24] is similar for Spring Boot:

1. Create a blank Spring Boot application using the "Spring Initializr"[25] – additional dependencies are not needed, unless the specific code requires them.
2. Add the ELG helper as a dependency, which is published to the central repository – for Gradle this means

```
implementation("eu.european-language-grid:elg-spring-boot-
    starter:1.0.0")
```

3. Create one or more beans annotated `@ElgHandler`, with one or more public methods annotated `@ElgMessageHandler`. Each method should take an ELG request type such as `TextRequest` as a parameter (and for binary requests a second parameter of type `Flux<DataBuffer>` for the actual data) and return an ELG response type such as `AnnotationsResponse` or a reactive streams Publisher producing that type.

In both cases, Micronaut and Spring Boot, developers must add their code in the appropriate places to call the actual LT tool and build a response based on the tool's results, using the model classes, e. g., an `AnnotationsResponse` object in the case that the results are standoff annotations. Once the objects are created, the frameworks and libraries are able to automatically serialise them into ELG-compliant JSON response messages. Similarly, the frameworks automatically translate the received input JSON messages to objects that can be easily handled by the developer, e. g., in the Spring Boot case a "text" JSON request is deserialised to a `TextRequest` object.

4.1.6 Helper Tools for Python

Similar to Java, the ELG team provides helper tools to create an ELG-compatible service from a Python-based LT service. The helper tools are included in the ELG Pypi package presented in Chapter 3. The package provides two Python classes that

[22] https://gitlab.com/european-language-grid/platform/lt-service-micronaut

[23] https://micronaut.io/launch

[24] https://gitlab.com/european-language-grid/platform/elg-spring-boot-starter

[25] https://start.spring.io

```python
from elg import FlaskService
from elg.model import TextRequest, AnnotationsResponse
import langdetect

class ELGService(FlaskService):
    def process_text(self, request: TextRequest):
        langs = langdetect.detect_langs(request.content)
        ld = {}
        for l in langs:
            ld[l.lang] = l.prob
        return AnnotationsResponse(features=ld)

service = ELGService("LangDetection")
app = service.app
```

Listing 3 Example ELG service created using the FlaskService class of the ELG Python package

can be extended to create a simple HTTP server that exposes an ELG-compatible endpoint of the LT tool. The ELG Python package also comes with a command-line interface (CLI) that helps with the creation of the Docker image.

For the ELG-compatible endpoint, the developer creates a Python class extending either `FlaskService` or `QuartService` as a base class, and must implement one of the four following handler methods: `process_text`, `process_structured_text`, `process_audio` or `process_image`, depending on the required input type for the LT service. This method will contain the code of the LT tool, it takes as input an ELG request object of the relevant type and should return a valid ELG response object. As a simple example, Listing 3 shows an LT tool that detects the language of the input text. The `ELGService` class inherits from the `FlaskService` class, which already contains all the code needed to create the server. This allows the developer to focus on the LT tool by only having to define the handler method. The `FlaskService` and `QuartService` classes work the same way; the first is based on Flask[26], which is more suited to CPU-bound synchronous code, the second uses the asyncio-based Quart framework[27], which is better for I/O bound code – `QuartService` is the only supported option if the handler method uses `async/await`[28]. Both base classes support the progress reporting mechanism and correctly handle exceptions raised by the tool, mapping them to ELG-compliant failure responses.

After having defined the HTTP server compatible with the ELG LT internal API using the `FlaskService` or `QuartService` class, the next step is to create the Docker image. The ELG CLI that comes with the Python package contains the `elg docker create` command to help during this step. The command automatically generates the dockerfile based on the arguments. Listing 4 shows an example for the language detection service presented in Listing 3. All the available options of the

[26] https://flask.palletsprojects.com/en/2.0.x/

[27] https://pgjones.gitlab.io/quart/

[28] https://www.european-language-grid.eu/2021/10/04/choose-the-right-tool-to-create-your-elg-service-in-python/

```
elg docker create -n ELGService -p elg_service.py -r langdetect
```

Listing 4 CLI command to generate the dockerfile automatically

command are accessible with `elg docker create --help`. Once the dockerfile is generated, the creation and the publication of the Docker image follows the same process as described in Section 4.1.4.

The ELG documentation includes a complete tutorial on how to create an ELG-compatible service using the Python package.[29] With these helper tools, we seek to facilitate as much as possible the creation of an ELG-compatible service from an LT tool implemented in Python. Using the Python helper ensures that the resulting service follows best practice in terms of error handling, request parsing, etc. and the construction of the dockerfile. This makes the services deployed in the ELG infrastructure efficient and secure.

4.1.7 Metadata Requirements

In addition to the metadata requirements for tools and services (see Chapter 2), the metadata records of ELG-compatible services must also include a set of technical metadata that are necessary for their deployment in the platform:

- `dockerDownloadLocation`: location of the image with the LT service;
- `serviceAdapterDownloadLocation`: location of the adapter image (if any);
- `executionLocation`: REST endpoint at which the LT tool is exposed within the Docker image (`http://localhost:{port}{/path}`);
- `additionalHWRequirements`: can be used to specify hardware requirements for this tool beyond the default limits of 512MB RAM and one CPU core;
- We also recommend providing *sample data* on which the service produces sensible results. Sample data help speed up the validation process, and can be used through the trial UIs and the "Code samples" tab by consumers who want to test the service. Providers can upload a file with samples, add a URL where the samples are located, or simply add the data in a dedicated free text element.

Figure 8 shows the mandatory elements replicating the editor (with sections horizontally and tabs vertically); elements marked with an asterisk are mandatory, given certain conditions, or required depending on the presence of another value or element.

[29] https://european-language-grid.readthedocs.io/en/stable/all/A1_PythonSDK/TutoServiceIntegration.html

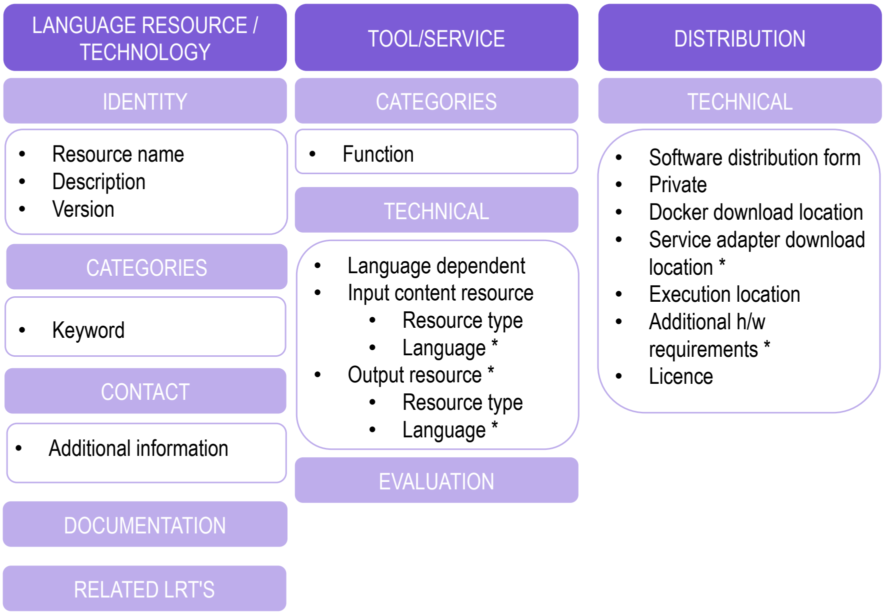


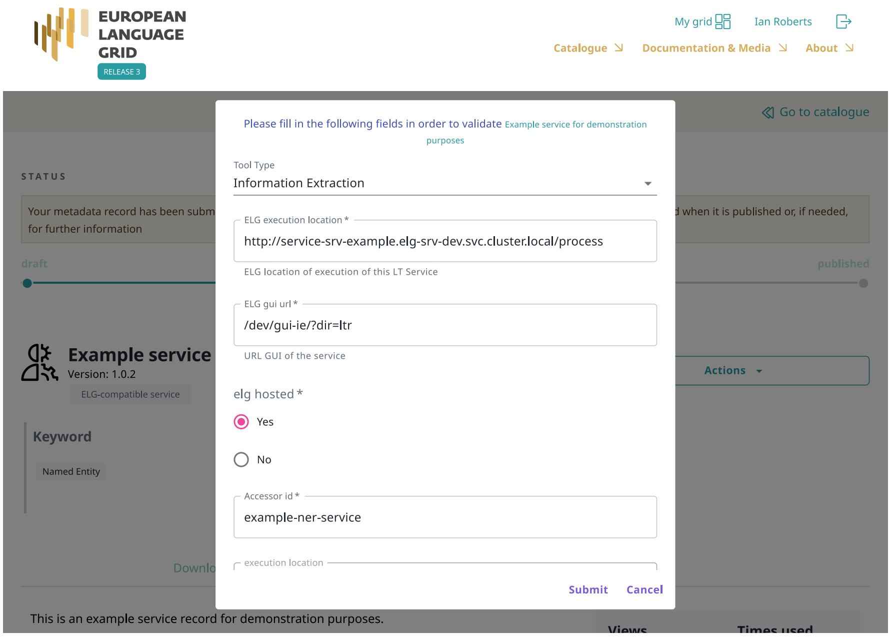
Fig. 8 Mandatory metadata for an ELG-compatible service

4.1.8 Technical Validation and Registration of ELG-Compatible Services

When LT providers have completed the packaging of their service, they can add it to ELG by supplying a metadata record via either the XML upload or editor mechanisms described in Section 2.1, specifying that it is an "ELG-compatible service" when prompted. Submitting the record initiates the validation process, which is performed internally by the ELG team.

The validation starts with the service registration process: The metadata or technical validator inspects the metadata record (accessed through the validator's grid) and deploys the service in the ELG Kubernetes cluster by creating the respective entries in the Helm charts that control the cluster. After that, the validator registers the service using a registration form (Figure 9), which specifies:

- Kubernetes-specific endpoint to be used by the LT execution server when calling the service, derived from the `executionLocation` metadata element value.
- ID of the trial UI to be used for rendering the processing results.
- Type of service (e. g., Speech Recognition, Text-to-Speech, Text Classification, etc.), which determines the appearance of the "Code samples" tab.
- Accessor ID that is used to form the public API endpoint URLs at which the service can be called. If the service was created as a new version of an existing service then it will share the same accessor ID as the service it replaces, but other than this, two distinct services must have different accessor IDs.



Please fill in the following fields in order to validate Example service for demonstration purposes

Tool Type: Information Extraction

ELG execution location *: http://service-srv-example.elg-srv-dev.svc.cluster.local/process
ELG location of execution of this LT Service

ELG gui url *: /dev/gui-ie/?dir=ltr
URL GUI of the service

elg hosted *: [✓ Yes] No

Accessor id *: example-ner-service

execution location

Your metadata record has been subm… for further information … when it is published or, if needed,

Example service
Version: 1.0.2
ELG-compatible service

This is an example service record for demonstration purposes.

Fig. 9 Registration form for ELG-compatible LT services

When the registration is completed, the service is visible only to the validator and the provider. The technical validator and the provider check that the service behaves as expected using test input, and that the results it returns can be rendered adequately by the assigned trial UI – this is where good sample data is particularly useful. When required, the validator may communicate with the provider to recommend changes in the technical implementation of the service or metadata. When the service is finally running as it should the technical validator approves it; it will be published once it also receives approval from the legal validator (see Chapter 2 for more information on the ELG publication life cycle).

4.1.9 Custom Try Out Interface

The ELG-provided trial UIs[30] have been designed to support common service types in a generic way, but there may be specific services for which the standard UIs either do not work or do not represent the results in a particularly intuitive way. If this is the case, it is possible to supply an alternative trial UI that better suits the service to be

[30] https://gitlab.com/european-language-grid/usfd/gui-ie

```javascript
// set up message listener
window.addEventListener('message', (e) => {
  if(e.origin ===
      'https://live.european-language-grid.eu') {
    const serviceInfo = JSON.parse(e.data);
    // configure UI here - store ServiceUrl and Authorization, fetch
    // parameter metadata from ApiRecordUrl, etc.
  }
});

// request configuration from the parent frame
setTimeout(() => {
  // the content of the message is unimportant, any message will trigger
  // the configuration reply.
  window.parent.postMessage("GUI:Ready for config",
          "https://live.european-language-grid.eu");
}, 500);
```

Listing 5 Typical JavaScript setup code for a trial UI

added. The standard UIs are open source under the Apache Licence[31], and providers are free to use this code as a basis for their own UI.

A trial UI is a single-page HTML/JavaScript application which is loaded into an `<iframe>` by the catalogue page when the user views an ELG-compatible service. Trial UIs run entirely in the browser and must not send user data to anywhere other than the ELG service endpoint and the i18n message resolver service. The JavaScript inter-frame messaging mechanism is used to supply the UI with the data it needs to configure itself for use with this particular service – when the UI `<iframe>` loads it must register a message listener that expects to receive message data that can be parsed as JSON, then dispatch a message to the parent frame to trigger the configuration message in return.[32] An example of this mechanism is shown in Listing 5.

The message event `data` sent by the parent frame will be JSON containing the following properties:

`ServiceUrl` The public LT service API URL at which the service can be called. The URL may include query string parameters if the service has more than one deployed version.

`ApiRecordUrl` The catalogue API URL from which the metadata record for this service may be retrieved with a `GET` request. This provides access to service parameter declarations, sample data, etc.

`Authorization` An HTTP `Authorization` header value that will authenticate calls to the `ServiceUrl` and `ApiRecordUrl` as the user who is logged in.

[31] https://www.apache.org/licenses/LICENSE-2.0

[32] To avoid the parent frame sending the configuration data before the UI frame is ready to receive it.

`Language` (optional) ISO code for the preferred language of the user. If present, this should be used as the `lang` parameter when resolving status messages to strings using the i18n resolver (see Section 4.1.1)

The custom UI can be hosted at any HTTPS URL – the `ServiceUrl` and `ApiRecordUrl` return the appropriate CORS headers to support cross-origin requests. Trial UIs run as Docker images in the ELG Kubernetes cluster. UIs can be created either by the ELG team or by a provider that needs a custom visualisation interface for the tools they contribute. Custom UIs can be integrated into ELG together with the ELG technical team.

4.2 ELG-hosted Resources

Together with metadata descriptions, providers are encouraged to upload the corresponding data files of their language resources so that they are readily available for download through ELG. To register their resources, they can select their preferred option from the ones presented in Section 2.1 and upload the accompanying files following the instructions in Section 2.2.

4.2.1 Requirements for ELG-hosted Resources

ELG requires data files to be uploaded as compressed ZIP files. There are no other specific metadata requirements apart from those defined for records of the resource type to which they belong (i. e., corpora, models, etc.). Chapter 2, Section 5, (p. 19 ff.) describes the metadata schema in more detail.

4.2.2 Packaging Data and Splitting Metadata Records: Recommendations

Datasets are composed of files that can be organised according to different criteria. For example, a multilingual corpus of texts from various domains can be described as a whole (one metadata record) or split into subsets (with corresponding metadata records) using the language or domain criteria. Depending on their intended use, different ways of packaging datasets and making them available can be suggested.[33]

We prepared a set of recommendations for the packaging of data files to enable users, especially those accessing ELG through programmatic APIs, to automatically identify, download and use corpora as is, without having to download them and manually search among them the subsets that interest them.[34]

[33] https://www.w3.org/TR/vocab-dcat-3 provides a similar argumentation for data distributions.

[34] These recommendations can be applied in different contexts, depending on whether the resource will be uploaded in ELG: when providers upload their corpora into ELG, they can use them to package the files and register the resource as one or multiple metadata records; if they decide to

The following cases are foreseen:

Multilingual resources are recommended to be split into bilingual pairs, so that users can easily find and use them, for example, in the case of bilingual corpora, to train bilingual models.

Resources from shared tasks are usually already split into training, development, gold, and test datasets, with a direct link to each of these. This is an established practice, and adopted in ELG as is. We recommend to register them as separate metadata records.

In both cases, a parent metadata record, to which the metadata records of all subsets can point is recommended using the "isPartOf" relation.

4.3 Metadata Records for External LRTs, Organisations and Projects

When external LRTs, organisations or projects are added to ELG, the only requirement for such metadata records is that they conform to the minimal version of the ELG metadata schema, i. e., they include the mandatory metadata elements described in Chapter 2, Section 5 (p. 19 ff.). Providers can use one of the options described in Section 2.1 (p. 69 ff.). For these records, the validation process aims to ensure that the metadata description is consistent and informative for users.

5 Provider-Related Functionalities in ELG and other Platforms

In this final section of the chapter we discuss some aspects of the functionalities offered to LT providers in ELG in relation to those available in other similar platforms. This discussion cannot be exhaustive. It rather attempts to give an overview of their design and implementation, highlight the main options utilised by the platforms, and offer explanations of the adopted approaches.

5.1 Metadata Requirements

Although the use of certain metadata schemas (e. g., DC[35], DCAT[36], schema.org[37], etc.) is growing, these schemas are usually restricted to the documentation of gen-

grant access to external corpora through hyperlinks, they can follow them for splitting the resource into one or multiple records and marking the availability through a direct link (element "downloadLocation").

[35] https://www.dublincore.org/specifications/dublin-core/dcmi-terms/

[36] https://www.w3.org/TR/vocab-dcat-3/

[37] https://schema.org

eral properties and do not satisfy domain- or community-specific requirements, especially with regard to discovery. Thus, most platforms use their own metadata schemas or ask for a minimum set of elements which are community-, domain-, or resource type specific (see Chapter 6 for a discussion of metadata schemas). Technical metadata are typically mandatory when resources are deployed in a platform. ELG has a detailed schema with a minimum set of required metadata to allow for flexibility and strictness when this is mandated for operational reasons (i. e., resources deployed in ELG, added by individuals, harvested from other sources).

CLARIN has initiated the Component MetaData Infrastructure[38], which provides a framework to describe and reuse different "metadata profiles" for resource types and communities. Specific metadata profiles, e. g., those of web services, are "recommended" with an aim to ensure interoperability and operational requirements. However, these profiles may promote different mandatory elements, depending on the use of the profile by each CLARIN Centre. Hugging Face[39] uses a dataset and model card, in which part of the required information is specified via YAML[40] tags.

5.2 Provider User Interface and Metadata User Interface

User-friendly editors that can cover *multiple* metadata schemas are difficult to implement, especially when the schemas have a complex structure. Nevertheless, most platforms include such an option. ELG, like META-SHARE[41] (Piperidis 2012; Piperidis et al. 2014), OpenMinTeD[42] (Labropoulou et al. 2018) and the European AI-on-demand platform[43], offer provider-specific UIs and a metadata editor supporting their respective schemas for describing resources. Hugging Face offers a rather simple UI with limited functionality. LAPPS Grid[44] (Ide et al. 2016) does not provide such UIs, a provider must communicate with the technical team in order to add services to the Galaxy[45] toolbox. Various CLARIN teams have created editors that support CMDI metadata (e. g., COMEDI[46], ARBIL[47], etc.). For more technical users, platforms offer APIs through which they can upload metadata records with JSON being the most widely used format for the records.

[38] https://www.clarin.eu/content/component-metadata
[39] https://huggingface.co
[40] https://huggingface.co/docs/datasets/v1.12.0/dataset_card.html
[41] http://www.meta-share.org
[42] https://openminted.github.io
[43] https://www.ai4europe.eu
[44] https://www.lappsgrid.org
[45] http://galaxy.lappsgrid.org
[46] https://clarino.uib.no/comedi/page
[47] https://portal.clarin.nl/node/14320

5.3 Try Out User Interface

Hugging Face offers embedded trial UIs to access their public “inference API”. These are similar in spirit to the ELG “try out” UI mechanism, with a publicly documented API being called by a generic user interface. In addition, Hugging Face provides “Spaces”[48] which enable users to create and deploy their own UIs for demonstrating a model. The approach followed by Hugging Face Spaces is different from ELG; it is based on developers coding their own back-end server code and front-end UI as a single unit using the Streamlit[49] or Gradio[50] Python libraries. The developer adds this source code to a Git repository and Hugging Face then deploys the code to their infrastructure directly from the source code rather than from a developer-supplied Docker image. The UI is tightly coupled to the server-side code and the “API” is an implementation detail that varies from “space” to “space”. ELG does not offer this kind of option by default, but the documented APIs mean that third parties could create a similar service on top of the LT services offered by ELG.

5.4 Helper Tools for Packaging Resources

As described in the previous sections, ELG offers command line utilities and SDKs for creating and submitting metadata for resources, preparing ELG-compatible services, etc. OpenMinTeD offered only a metadata validation service, without a corresponding command line tool. The European AI-on-demand platform, however, provides such utilities through Acumos[51] an open source framework, that makes it easy to build, share, and deploy AI applications.

5.5 Packaging Data Resources

ELG has adopted a lightweight policy for the packaging of uploaded datasets, given that direct deployment is currently not foreseen. In the CLARIN infrastructure, each centre has its own processes and recommended formats for uploaded resources, taking into account preservation or deployment purposes (e. g., submitting the resources to processing). Hugging Face maintains a detailed set of instructions for the upload of datasets and models, which is crucial for ensuring that they can be deployed.

[48] https://huggingface.co/spaces

[49] https://streamlit.io

[50] https://gradio.app

[51] https://www.acumos.org

6 Conclusions

ELG enables producers of language resources and language technology tools and services to upload, describe, share, and distribute their services and products as well as to describe their companies, academic organisations and projects. ELG offers to providers web-based user interfaces for describing LT resources or related entities with metadata records and provides them with functionalities for managing the life cycle of their assets; a billing component for commercial services and resources has been implemented (see Chapter 3, Section 6, p. 59 f.) and will be activated as soon as the ELG legal entity is in place (see Chapter 13). Providers of LT tools can exploit such functionalities to integrate LT tools in the ELG platform as ready-to-deploy services. LT data and tool providers are requested to follow the specifications and recommendations for packaging tools and resources to be uploaded in ELG. In the wider language technology ecosystem, provider-related functionalities are offered by other platforms, too, respecting their own target groups, objectives and policies. ELG has built bridges to some of these platforms, see Chapter 6 for more details.

References

Ide, Nancy, James Pustejovsky, Christopher Cieri, Eric Nyberg, Denise DiPersio, Chunqi Shi, Keith Suderman, Marc Verhagen, Di Wang, and Jonathan Wright (2016). "The Language Application Grid". In: *Worldwide Language Service Infrastructure*. Ed. by Yohei Murakami and Donghui Lin. Cham: Springer, pp. 51–70. DOI: 10.1007/978-3-319-31468-6_4.

Labropoulou, Penny, Dimitris Galanis, Antonis Lempesis, Mark Greenwood, Petr Knoth, Richard Eckart de Castilho, Stavros Sachtouris, Byron Georgantopoulos, Stefania Martziou, Lucas Anastasiou, Katerina Gkirtzou, Natalia Manola, and Stelios Piperidis (2018). "OpenMinTeD: A Platform Facilitating Text Mining of Scholarly Content". In: *Proceedings of WOSP 2018 (co-located with LREC 2018)*. Miyazaki, Japan: ELRA, pp. 7–12. URL: http://lrec-conf.org/workshops/lrec2018/W24/pdf/13_W24.pdf.

Labropoulou, Penny, Katerina Gkirtzou, Maria Gavriilidou, Miltos Deligiannis, Dimitris Galanis, Stelios Piperidis, Georg Rehm, Maria Berger, Valérie Mapelli, Michael Rigault, Victoria Arranz, Khalid Choukri, Gerhard Backfried, José Manuel Gómez Pérez, and Andres Garcia-Silva (2020). "Making Metadata Fit for Next Generation Language Technology Platforms: The Metadata Schema of the European Language Grid". In: *Proceedings of the 12th Language Resources and Evaluation Conference (LREC 2020)*. Ed. by Nicoletta Calzolari, Frédéric Béchet, Philippe Blache, Christopher Cieri, Khalid Choukri, Thierry Declerck, Hitoshi Isahara, Bente Maegaard, Joseph Mariani, Asuncion Moreno, Jan Odijk, and Stelios Piperidis. Marseille, France: ELRA, pp. 3421–3430. URL: https://www.aclweb.org/anthology/2020.lrec-1.420/.

Piperidis, Stelios (2012). "The META-SHARE Language Resources Sharing Infrastructure: Principles, Challenges, Solutions". In: *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*. Ed. by Nicoletta Calzolari, Khalid Choukri, Thierry Declerck, Mehmet Uğur Doğan, Bente Maegaard, Joseph Mariani, Asuncion Moreno, Jan Odijk, and Stelios Piperidis. Istanbul, Turkey: ELRA.

Piperidis, Stelios, Harris Papageorgiou, Christian Spurk, Georg Rehm, Khalid Choukri, Olivier Hamon, Nicoletta Calzolari, Riccardo del Gratta, Bernardo Magnini, and Christian Girardi (2014). "META-SHARE: One year after". In: *Proceedings of the 9th Language Resources and Evaluation Conference (LREC 2014)*. Ed. by Nicoletta Calzolari, Khalid Choukri, Thierry Declerck, Hrafn Loftsson, Bente Maegaard, Joseph Mariani, Asuncion Moreno, Jan Odijk, and Stelios Piperidis. Reykjavik, Iceland: ELRA, pp. 1532–1538. URL: http://www.lrec-conf.org/proceedings/lrec2014/pdf/786_Paper.pdf.

Rehm, Georg, Stelios Piperidis, Kalina Bontcheva, Jan Hajic, Victoria Arranz, Andrejs Vasiļjevs, Gerhard Backfried, José Manuel Gómez Pérez, Ulrich Germann, Rémi Calizzano, Nils Feldhus, Stefanie Hegele, Florian Kintzel, Katrin Marheinecke, Julian Moreno-Schneider, Dimitris Galanis, Penny Labropoulou, Miltos Deligiannis, Katerina Gkirtzou, Athanasia Kolovou, Dimitris Gkoumas, Leon Voukoutis, Ian Roberts, Jana Hamrlová, Dusan Varis, Lukáš Kačena, Khalid Choukri, Valérie Mapelli, Mickaël Rigault, Jūlija Meļņika, Miro Janosik, Katja Prinz, Andres Garcia-Silva, Cristian Berrio, Ondrej Klejch, and Steve Renals (2021). "European Language Grid: A Joint Platform for the European Language Technology Community". In: *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations (EACL 2021)*. Kyiv, Ukraine: ACL, pp. 221–230. URL: https://www.aclweb.org/anthology/2021.eacl-demos.26.pdf.

Chapter 5
Cloud Infrastructure of the European Language Grid

Florian Kintzel, Rémi Calizzano, and Georg Rehm

Abstract The European Language Grid (ELG) is a cloud-based platform, utilising a variety of software packages as well as infrastructure components and virtual hardware. The additional software components developed by the ELG project are usually provided as open source to facilitate re-use by third parties. This chapter provides an overview of the infrastructural setup used by the ELG cloud platform. The selected architecture also has implications for providers as well as users of the platform, e. g., in terms of the scaling behaviour of individual Language Technology (LT) services.

1 Introduction

One of the key technical goals of the ELG cloud platform is the ability to integrate functional Language Technology (LT) services from a variety of sources, i. e., to build a large platform and a corresponding community of providers and users of these services. The LT tools and services to be continuously integrated into the ELG platform are, thus, heterogeneous and vary in their technical setup, which is why a set of common approaches needs to be established to make the integration of the tools and services possible. One of the most basic joint technical approaches is the requirement for all functional services to be containerised so that they can run on the ELG cloud infrastructure. Providers can optionally benefit from utilising additional support functionality, e. g., source code repositories, container registries and deployment pipelines offered by the ELG platform.

Conceptually, the ELG platform consists of three layers, the user interface (UI) layer, the back end layer and the base infrastructure (see Figure 1). While the UI and back end are described in more detail in Chapters 2, 3 and 4, the present chapter focuses on the base infrastructure setup along with supporting functionality. Among others, this chapter is helpful for providers of functional LT tools and services or users interested in running parts of the ELG platform on their own hardware.

Florian Kintzel · Rémi Calizzano · Georg Rehm
Deutsches Forschungszentrum für Künstliche Intelligenz GmbH, Germany,
florian.kintzel@dfki.de, remi.calizzano@dfki.de, georg.rehm@dfki.de

The rest of this chapter is structured as follows. First, Section 2 gives an overview of the building blocks of the ELG infrastructure. Section 3 provides information about the deployment side of the ELG platform, while Section 4 describes how the platform's scaling profile lends itself to usage in different real-world scenarios. Finally, Section 5 concludes the chapter with an overview of future work on the ELG platform infrastructure.

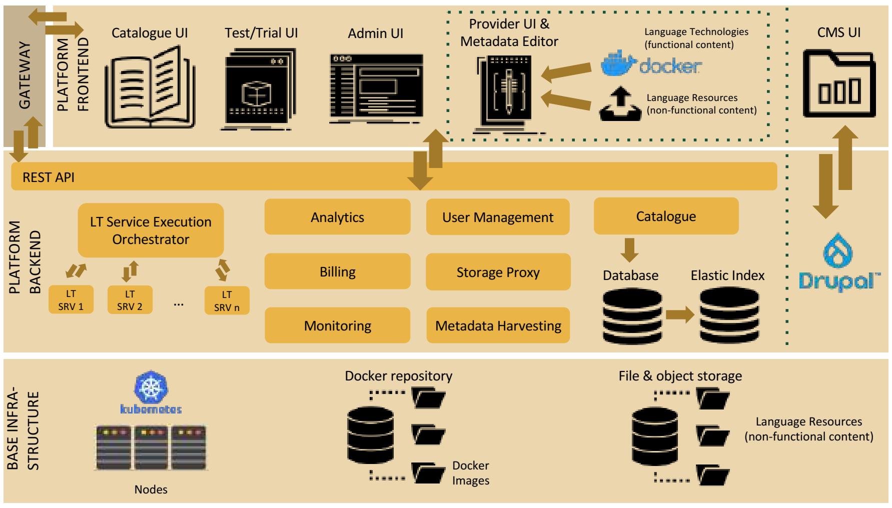


Fig. 1 ELG platform architecture

2 Cloud Infrastructure

The base infrastructure consists, first and foremost, of the compute nodes on which the European Language Grid runs, alongside their respective volume storage and networking facilities. On these, the Kubernetes[1] core components are installed (Section 2.1) including S3-compatible object storage (Section 2.2). We use a *managed* approach to Kubernetes, i. e., the installation, update and operation of the Kubernetes system itself is taken care of by a cloud provider. Together, this forms the hardware basis of the European Language Grid.

Conceptually, the base infrastructure also consists of a larger set of Git[2] repositories and container registries which are described in Sections 2.3 and 2.4.

[1] https://kubernetes.io

[2] https://git-scm.com

2.1 Kubernetes and Cloud Native

Kubernetes is an open source system for automating deployment, scaling, and management of containerised applications. It has seen widespread usage in recent years as *the* container orchestration tool of choice. Adoption of Kubernetes in a *managed* setup was still in a relatively early stage at the time the ELG project was exploring different cloud providers in early 2019. While various products by the typical hyperscalers already existed, European providers had only very recently started offering comparable solutions.

Our selection of Kubernetes as the framework of choice for ELG was primarily based on the following criteria:

- Kubernetes provides self-healing capabilities that can detect common failure situations and restart affected containers automatically.
- Through the use of a managed approach to Kubernetes, failures of the core Kubernetes system itself are the responsibility of the cloud provider.
  These first two criteria together allowed the ELG project to have a relatively small footprint in terms of operational complexity as failures are either self-healed or taken care of by the cloud provider, at least in theory. While exceptions *do* exist, this still has reduced the operational effort considerably.
- Kubernetes facilitates the usage of OCI-compatible containers.[3] As ELG aims to integrate different technologies used for the implementation of LT services and tools, OCI-compatible containers form a common approach for integration.
- Kubernetes provides off-the-shelf functionality for scaling up resources based on dynamic load. As ELG integrates hundreds of different LT tools and services, this functionality was deemed essential.
- Kubernetes namespaces[4] are useful to separate the different platform components from one another.
- Continuous adoption of Kubernetes within the industry assures continued support and development of this technology.

An ecosystem of compatible technologies has been established around Kubernetes with the Cloud Native Computing Foundation (CNCF).[5] CNCF promotes the use of a large set of base technologies for solving, e. g., authentication, monitoring, deployment and other common challenges. Most supporting technologies used in ELG (Section 3.2) are part of CNCF. Alongside this, a set of architecture patterns has emerged that aim to support properties such as Gannon et al. (2017):

- Cloud-native applications often operate at the global level.
- Cloud-native applications must scale well with thousands of concurrent users.
- Built on the assumption that infrastructure is fluid and failure is constant.
- Designed so that upgrade and test occur without disrupting production.
- Security must be part of the underlying application architecture.

[3] https://opencontainers.org – Open Container Initiative

[4] https://kubernetes.io/docs/concepts/overview/working-with-objects/namespaces

[5] https://www.cncf.io

2.2 Storage

The various components of the European Language Grid platform utilise persistent storage differently, as follows:

- Static Language Resources, i. e., corpora, models etc. available for direct download on the European Language Grid platform are persisted on S3-compatible object storage and can be fetched from there.
- The major infrastructural part of the ELG platform – the hundreds of LT tools services – do not utilise persistent storage at all, as they are designed stateless. All application code is shipped within an OCI-compatible container. This includes additional resources needed to run the service, e. g., language models and additional configuration files.
- The core ELG platform components (catalogue, authentication, CMS etc.) utilise network block storage attached to their running containers for persistence. This block storage is in turn backed up to the object storage on a regular basis.

Therefore, static resources *can* potentially be available for direct download and be included in the respective service container image as well. We decided for this approach to simplify deployment and management of images and resources, e. g., for a local installation of a set of LT services, it is only necessary to pull and run the respective images, i. e., no additional language resources need to be handled. Though this potentially results in duplication of resource files (within an image and as an additional separate file for download) it was deemed a necessary trade-off to keep the deployment model easier.

2.3 Software Repositories

ELG is comprised of various independent software packages for, e. g., platform components and individual LT services. The main ELG GitLab project repository[6] is set up as a GitLab group, consisting of various sub-groups and repositories. The different repositories in this group can be categorised as follows.

- The ELG Infrastructure Repository consists of a set of configuration files, mostly in the form of Helm[7] charts (see Section 3.1). These define which packages, i. e., containers, the ELG system consists of, as well as numerous additional configuration parameters such as the number of replicas and package-specific configurations. It can be used to set up multiple clusters. We maintain different branches within the repository, usually at least one for the development and one for the production cluster. The branches are not only used to distinguish between specific configurations for each cluster, but present different versions

[6] https://gitlab.com/european-language-grid

[7] https://helm.sh

of the ELG system as it matures during development. This is used to facilitate a staged roll-out to the production cluster. The actual source code for these components is not part of this repository. It only includes references to the container registries with the specific components. When installing the ELG cluster, these images are then downloaded ("pulled") from these registries.
- The ELG Cluster Admin repository holds cluster-specific configurations for each ELG instance that are applied separately from the settings of the ELG Infrastructure Repository. These mostly consist of the list of active administrative users for accessing the ELG infrastructure (those needing access to the infrastructure the ELG is running on, not users of the ELG platform), their roles and access rights as well as the configuration for build-bot, our continuous integration utility of choice. Included are also various utilities to manage the cluster. This repository is not needed for local deployment of the ELG, as such a deployment is usually only meant for a single user, typically a developer, and does not participate in continuous deployment.
- The main ELG GitLab platform project repository.[8] This repository hosts the individual components that make up the the core ELG platform and ELG website. These are mainly the platform (catalogue back end and front end components and the website content management system, along with a larger set of internal supporting and utility components.
- Individual sub-groups with repositories for individual LT services, grouped by provider. These consist solely of the LT services provided by members or associates of the ELG project consortium.

Implementation code for LT services not provided by ELG project consortium members is not usually held in the ELG GitLab group but rather managed via provider-specific repositories.

2.4 Container Registries

The images for instantiating containers in the ELG cluster are stored in various container registries. The Kubernetes installation powering ELG pulls the images from these registries on demand. These can be categorised as follows.

- The ELG GitLab project registry[9] is the registry that corresponds to the main ELG GitLab group, it hosts all images for all ELG core platform components (e. g., UI, back end, utilities) and for several ELG LT services developed by ELG project consortium partners. This registry allows public access to facilitate download and re-use of ELG components.
- Public registries for various externally implemented third-party components such as database system, identity and access management.

[8] https://gitlab.com/european-language-grid/platform

[9] registry.gitlab.com/european-language-grid

- Private registries of partners who do not publish their LT services under an open source license (proprietary LT services) or need to use their own registries for technical reasons.
- Various other public registries for open source LT services.
- The dedicated ELG registry.[10] As LT service images are partly pulled from registries external to the ELG project, this registry was set up to serve as a point to collect LT service images when they are ingested into ELG in order to perform versioning. Using this approach, ELG can ensure the availability of older versions of certain tools even if their original site is no longer serving them.

3 Installation

ELG utilises a GitOps approach (see, e. g., Beetz and Harrer 2021) to deployment, i. e., the configuration necessary to set up the compute cluster is managed by version control. The base artefact for deployment is the Helm chart.[11] Helm charts are used to manage the installation and update the ELG platform. Each chart bundles a set of components along with their configuration. All custom charts are defined in the ELG platform repository GitLab group (Section 3.1). Alongside the custom charts, a larger set of third-party charts is utilised to set up the respective components (Section 3.2).

We apply the charts to the cluster using a Continuous Integration (CI) approach, i. e., automatic deployment happens whenever changes to the configuration are detected by the CI (Figure 2).

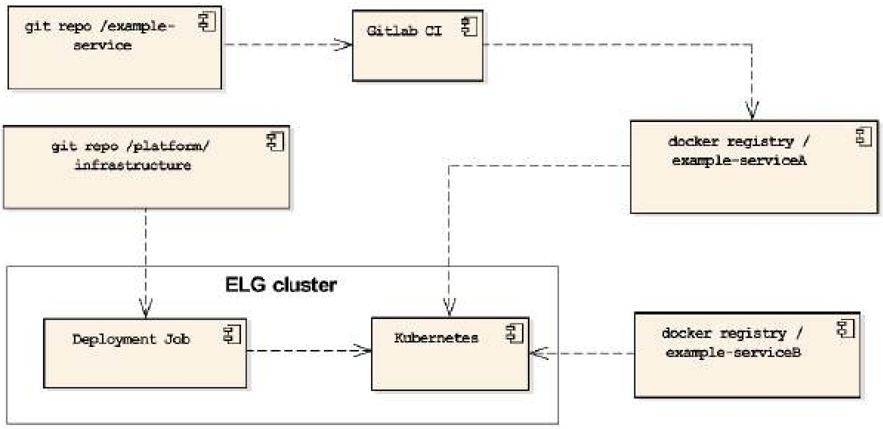

git repo /example-service
Gitlab CI
docker registry / example-serviceA
git repo /platform/ infrastructure
ELG cluster
Deployment Job
Kubernetes
docker registry / example-serviceB

Fig. 2 ELG continuous integration

[10] registry.european-language-grid.eu

[11] https://helm.sh

If a new version of the infrastructure setup is detected, the CI checks out the respective changes and applies them to the cluster state. Any new container versions are then pulled from their distributed container registries. The Kubernetes cluster is updated with the latest configuration and takes care of gracefully shutting down and instantiating new containers.

Continuous integration regarding the ELG infrastructure only deals with updating the ELG cluster with the latest set of images (as specified by their version number) and configuration. It does not deal with building the respective images themselves.

3.1 ELG Charts

These charts were specifically developed for ELG and control its setup and installation. The packages are meant to be installed together, though it is possible to install only a subset for specific use cases (e. g., custom local installations). The architecture of the ELG is described in Chapter 2 as well as, e. g., Rehm et al. (2021), which is why we focus only on the software packages themselves.

- The ELG core package consists of definitions for various supporting functionalities of ELG. These are the Ingress[12] definitions for routing incoming traffic into the ELG cluster, the configuration for the rest server component as well as the configuration for the temporary storage component (used for large file operations). Various smaller configurations can also be found here, e. g., priority classes for pod scheduling, support for maintenance operations and others.
- The ELG back end chart consists of the definitions for the main back end components, the Django[13] and React[14] powered applications that form the ELG catalogue and the ELG back end and administrative applications. Included in this chart are also a set of utility functions that deal with housekeeping.
- The ELG LT services chart bundles the whole set of individual LT services installed in ELG. It is actually a collection of charts that follow a common structure, each sub-chart consisting of the definitions for the LT services of a specific LT services provider as well as a common chart for open source LT services by providers who only offer a small set of services. A definition for each individual LT service consists at the minimum of the reference to its image location, but can consist of numerous additional configurations, e. g., specific hardware requirements, helper images, parameters for scaling the service up and down and various other parameters.

[12] https://kubernetes.io/docs/concepts/services-networking/ingress

[13] https://www.djangoproject.com

[14] https://reactjs.org

3.2 Third-Party Charts

Apart from the core components, we use a set of third-party components, which provide their functionality to the ELG cluster. In the following, we briefly describe the main third-party components.

- Cert-manager[15] is a tool to manage issuing and updating of TLS certificates. It is used to install and refresh TLS certificates to allow for the encryption of all HTTPS traffic that reaches the cluster via one of the configured ingress-rules.
- The Horizontal Pod Autoscaler (HPA)[16] is a standard Kubernetes component used to scale pods based on their load and runtime behaviour. For scalability and load monitoring, Kubernetes collects certain metrics, e. g., CPU and memory load, from each pod. Therefore, it is necessary to have at least one instance of each type of pod to be up and running at all times. Otherwise, no metrics can be collected. This setup is useful to scale ELG core components, e. g., the portal website and back end. It cannot be utilised as is to scale the hundreds of LT services offered by the platform, as these need to be scaled down to zero replicas if they are not needed to not exceed the cluster capacity. Therefore, we introduced KNative (see below), which is feeding the standard autoscaler with a new metric "concurrency", based on the number of active requests to that LT service. Scaling those services still makes use of cluster-autoscaler functionality, but with the new metric also being available if no active replica of an LT service is instantiated.
- KNative[17] and Kourier[18] give ELG the possibility to scale down LT services based on the current number of parallel requests to them (concurrency). The concurrency metric is available even if there is no active replica of an LT service. KNative buffers HTTP requests to one of the ELG APIs until the specific LT service's container has started and keeps track of the concurrency metric to terminate the replica if it is no longer needed. We cannot overstate the importance of this functionality for ELG as the platform consists of hundreds of individual LT service components, not all of which need to run all the time, i. e., it would not be efficient to have all these services consume resources while in idle state. Starting up a container takes a certain amount of time though, while the service initialises. Using a service after it has not been used in a while therefore requires a certain spin-up time. KNative does not natively provide facilities to reduce the spin-up time further, but additional methods might be helpful in the ELG context, e. g., predictive auto scaling (Nanayakkara 2021). If frequent traffic is expected for a particular service, it can easily be configured to have one or more instances running at any given time, depending on hardware availability.

[15] https://cert-manager.io/docs

[16] https://kubernetes.io/de/docs/tasks/run-application/horizontal-pod-autoscale

[17] https://knative.dev/docs

[18] https://github.com/3scale-archive/kourier

- Ingress-Ningx[19] is installed to act as ingress-controller, i. e., handling HTTP traffic received and forwarding them to their respective endpoint within the cluster.
- Keycloak[20] is an open source solution for authentication and authorisation. It interfaces with front end, back end and LT services to provide single-sign on.
- Elasticsearch[21] is used to index the catalogue database for fast faceted search.
- Prometheus[22], Grafana, Loki and AlertManager form the ELG monitoring solution. They collect and analyse logs and metrics from all running components in the cluster (including the hardware) and provide visualisations in the form of dashboards and diagrams (Figure 3).

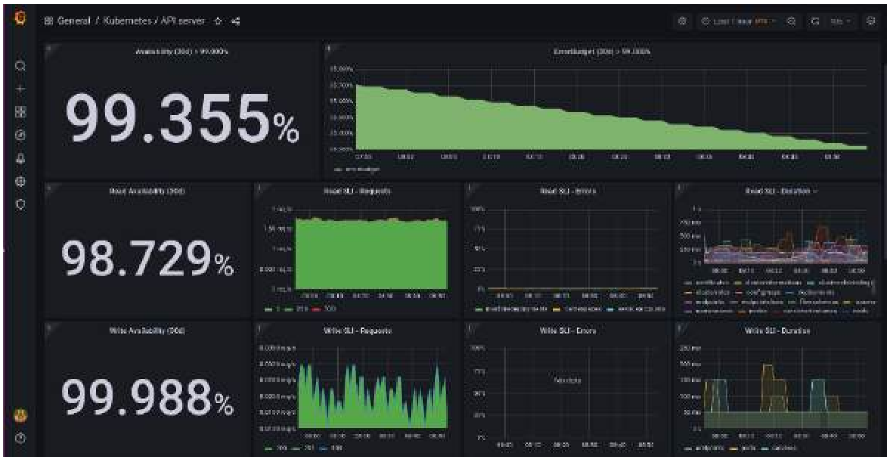


Fig. 3 Monitoring ELG using Prometheus and Grafana

- The ELG back end database uses PostgreSQL[23], a well-supported open source database engine. It holds all relevant data concerning the ELG catalogue, e. g., projects, organisations, LT resources, LT service as well as user information.
- MariaDB[24] is used for persistence of the Drupal CMS that powers the ELG portal. We plan to move this over to PostgreSQL for ease of maintenance.
- Not an off-the-shelf component, but rather specifically adapted for ELG, the s3proxy[25] facilitates the upload of LT resources (models, corpora, but also project and organisation logos etc.) to ELG. It acts as a proxy to the S3-compatible object storage that takes care of validating upload authorisation with the ELG back end and streams data to the object storage.

19 https://nginx.org

20 https://github.com/keycloak/keycloak

21 https://github.com/elastic/elasticsearch

22 https://prometheus.io

23 https://www.postgresql.org

24 https://mariadb.org

25 https://gitlab.com/european-language-grid/platform/s3proxy

4 Scalability of LT Tools and Services

ELG is optimised for stateless LT tools and services. Its database systems are exclusively used by the platform back end for the metadata catalogue, user data etc. LT services do not have persistence enabled for them, with the exception of temporary files used for large file uploads. In the following, we describe our approach for scaling up individual LT services and describe its impact for service usability.

4.1 Implementation

With the goal of hosting thousands of individual LT tools and services with very different hardware needs, it is neither feasible nor practical to have all of them instantiated at the same time as this would require hundreds of Gigabytes of RAM even in idle mode, i. e., even if none of them are actually used. Therefore, ELG leverages the capabilities of KNative[26] which make is possible to automatically scale down services not currently in use to zero replicas. In this state, an LT service does not consume any hardware resources.

Scaling up an LT service happens automatically to an initial number of replicas once a request has been received for that individual service. Requests are buffered while new containers are starting up. This setup is especially suitable for services seeing little or irregular traffic. Further scale-up happens when a configurable threshold of concurrent requests for a given service is exceeded.

LT services deployed on ELG need to be aware that their life-cycle is exclusively controlled by Kubernetes and they need to expect to be started, stopped and horizontally scaled regularly, e. g., when the scheduler detects low resource situations on one of the nodes, if a container fails to respond, if high traffic is received to an LT service and other situations. LT services, therefore, highly benefit from quick start-up times and this is one of the reasons, why we opted for LT services to include necessary resources like models into their OCI images directly.

4.2 Use Cases

Given its scalability (Section 4.1), a number of use cases can be solved with ELG.

- Demonstration of service functionality: providers of LT tools and services can freely deploy their services to the platform and can expect to be discoverable via the platform's catalogue. For the try out functionality of services, a certain spin-up time from idle mode will not impact its usefulness. More performant installations of a given service could, e. g., be offered by the providers themselves.

[26] https://knative.dev/docs

- Batch processing of multiple documents: as the containers of an individual LT service will stay instantiated for some time after usage before scale-down happens, ELG is a good fit for batch processing as the initial scale-up time will not be a major contributing factor to processing time.
- For services intended to power applications where quick response times are required (e. g., mobile apps), however, the time it takes to spin up a container is likely too long (some seconds, depending on a service's implementation). This is why services on ELG can be configured to stay instantiated all the time and still benefit from dynamic scaling in high load situations. To be feasible, dedicated hardware is necessary, which service providers will be able to reserve on the ELG platform for a fee in the future so their services will show the responsiveness and performance they require.
- Remote processing is a second alternative for LT service providers who want to offer their services to the public. In this setup, the ELG platform uses a proxy to forward user requests to an external installation of a service, managed by the service providers themselves. This offers a flexible approach for providers to tune the hardware setup according to their own requirements.
- Management of non-functional LT resources, where only bandwidth limits scalability instead of compute capacity.

5 Conclusions

The ELG platform is growing continuously and the capacity, availability, operational readiness and tooling support of the base infrastructure need to evolve accordingly. We foresee a need to evolve in the following areas in particular.

- Hardware capacity and cost distribution: through the use of cloud technology, ELG has the technical capability to grow horizontally as required by the encountered load. In practice, though, the available hardware is restricted by budget considerations. Batches of utilised compute resources would need to be individually matched to the user requesting them or the provider offering them, to allow the ELG to calculate operational costs on a per request basis. With this and the emerging payment functionality, individual resource usage can be reimbursed.
- Hardware acceleration: ELG currently runs on CPUs exclusively. Already now, a larger number of LT services in ELG would benefit from GPU support. Apart from higher costs, GPU support will pose a number of technical challenges, among them a need to map LT services to specific compute nodes (with or without GPU support).
- Integration and deployment support: the initial integration of a functional LT service will need further automation and tooling support to be able to cope with increased demand and an increased number of running services.
- Workflow support: ELG would benefit from a possibility for easy workflow composition, spanning multiple LT services. Initial efforts have been started towards this goal (Moreno-Schneider et al. 2020).

- Gaia-X: in the Gaia-X[27] project OpenGPT-X[28] the ELG platform is currently being integrated into the wider Gaia-X ecosystem, i. e., ELG is further extended so that it complies to the technical Gaia-X specifications. This will enable all ELG LT services and resources to be discoverable and usable within Gaia-X.

This list only includes a selection of likely areas of improvement. Many additional use cases and requirements for ELG can be imagined – the platform infrastructure will need to grow and evolve as required.

References

Beetz, Florian and Simon Harrer (2021). "GitOps: The Evolution of DevOps?" In: *IEEE Software* 39.4, pp. 70–75. DOI: 10.1109/MS.2021.3119106.

Gannon, Dennis, Roger Barga, and Neel Sundaresan (2017). "Cloud-Native Applications". In: *IEEE Cloud Computing* 4.5, pp. 16–21. DOI: 10.1109/MCC.2017.4250939.

Moreno-Schneider, Julián, Peter Bourgonje, Florian Kintzel, and Georg Rehm (2020). "A Workflow Manager for Complex NLP and Content Curation Pipelines". In: *Proc. of the 1st Int. Workshop on Language Technology Platforms (IWLTP 2020, co-located with LREC 2020)*. Ed. by Georg Rehm, Kalina Bontcheva, Khalid Choukri, Jan Hajic, Stelios Piperidis, and Andrejs Vasiljevs. Marseille, France, pp. 73–80. URL: https://www.aclweb.org/anthology/2020.iwltp-1.12.pdf.

Nanayakkara, Pallage Kamindu (2021). "Serverless Performance Improvement for Knative using Predictive Auto Scaling". PhD thesis. Sri Lanka: Informatics Institute of Technology. URL: http://dlib.iit.ac.lk/xmlui/handle/123456789/702.

Rehm, Georg, Stelios Piperidis, Kalina Bontcheva, Jan Hajic, Victoria Arranz, Andrejs Vasiļjevs, Gerhard Backfried, José Manuel Gómez Pérez, Ulrich Germann, Rémi Calizzano, Nils Feldhus, Stefanie Hegele, Florian Kintzel, Katrin Marheinecke, Julian Moreno-Schneider, Dimitris Galanis, Penny Labropoulou, Miltos Deligiannis, Katerina Gkirtzou, Athanasia Kolovou, Dimitris Gkoumas, Leon Voukoutis, Ian Roberts, Jana Hamrlová, Dusan Varis, Lukáš Kačena, Khalid Choukri, Valérie Mapelli, Mickaël Rigault, Jūlija Meļņika, Miro Janosik, Katja Prinz, Andres Garcia-Silva, Cristian Berrio, Ondrej Klejch, and Steve Renals (2021). "European Language Grid: A Joint Platform for the European Language Technology Community". In: *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations (EACL 2021)*. Kyiv, Ukraine: ACL, pp. 221–230. URL: https://www.aclweb.org/anthology/2021.eacl-demos.26.pdf.

[27] https://www.gaia-x.eu, https://www.data-infrastructure.eu

[28] https://www.opengpt-x.de

Chapter 6
Interoperable Metadata Bridges to the wider Language Technology Ecosystem

Penny Labropoulou, Stelios Piperidis, Miltos Deligiannis, Leon Voukoutis, Maria Giagkou, Ondřej Košarko, Jan Hajič, and Georg Rehm

Abstract One of the objectives of the European Language Grid is to help overcome the fragmentation of the European Language Technology community by bringing together language resources and technologies, information about them, Language Technology consumers, providers and the wider public. This chapter describes the mechanisms ELG has put in place to build interoperable bridges to related initiatives, infrastructures, platforms and repositories in the wider Language Technology landscape. We focus on the different approaches implemented for the exchange of metadata records about, in a generic sense, resources and exemplify them with the help of four use cases through which the ELG catalogue has been further populated. The chapter presents the protocols used for the population processes as well as the adaptations of the ELG metadata schema and platform policies that proved necessary to be able to ingest these new records. Last, we discuss the challenges emerging in large-scale metadata aggregation processes and propose a number of alternative options to address them.

1 Introduction

One of the objectives of the European Language Grid is to help overcome the fragmentation of the European Language Technology community by bringing together language resources and technologies, information about them, Language Technology consumers, providers and the wider public.

Additionally, ELG is meant to support digital language equality in Europe (STOA 2018; European Parliament 2018), i. e., to create a situation in which *all* European

Penny Labropoulou · Stelios Piperidis · Miltos Deligiannis · Leon Voukoutis · Maria Giagkou
Institute for Language and Speech Processing, R. C. "Athena", Greece, penny@athenarc.gr, spip@athenarc.gr, mdel@athenarc.gr, leon.voukoutis@athenarc.gr, mgiagkou@athenarc.gr

Ondřej Košarko · Jan Hajič
Charles University, Czech Republic, kosarko@ufal.mff.cuni.cz, hajic@ufal.mff.cuni.cz

Georg Rehm
Deutsches Forschungszentrum für Künstliche Intelligenz GmbH, Germany, georg.rehm@dfki.de

G. Rehm (ed.), *European Language Grid*, Cognitive Technologies,
https://doi.org/10.1007/978-3-031-17258-8_6

languages are supported through technologies equally well. Technological support for human languages has been characterised by a stark predominance of LTs for English, while almost all other languages are only marginally supported and, thus, in danger of digital extinction (Kornai 2013; Rehm et al. 2014, 2020b; ELRC 2019; Calzolari et al. 2011; Soria et al. 2012). More than ten years after the initial findings (Rehm and Uszkoreit 2012), Europe's languages are still affected by this stark imbalance in 2022, as attested in the most recent series of Language Reports (Giagkou et al. 2022) prepared by the European Language Equality[1] project, which develops a strategic research, innovation and implementation agenda as well as a roadmap for achieving full digital language equality in Europe by 2030. In collaboration with ELG, one of the first steps towards Digital Language Equality has been the creation of an inventory of language resources and technologies available for Europe's languages and its regular monitoring.

In tandem with its operation as an integrated LT platform, through a battery of selection, conversion and ingestion processes described in this chapter, ELG aims to act as a one-stop shop and single entry point to homogenised descriptions of language resources and technologies. Section 2 positions the ELG approach towards this goal in the broader context of the exchange of metadata between catalogues and repositories. Section 3 presents four use cases through which the ELG catalogue has been populated with metadata records from other sources, highlighting the features that have influenced the different solutions we adopted. Section 4 presents the adaptations made in the ELG metadata schema and platform policies to take into account the outputs of these import procedures. Finally, in Section 5 we discuss, based on the experience gained in this process, the challenges that need to be addressed in the aggregation of metadata from multiple sources in order to share and promote the use and re-use of resources, data and software among community members.

2 Approach

There are a wide range of digital catalogues, repositories and, in general, infrastructures[2] that support the publication and dissemination of digital artefacts and resources, which can be classified along various dimensions. Institutional catalogues hosting all types of resources (publications, datasets, tools, etc.) produced by practitioners affiliated with an institution, catalogues that focus on resources produced by specific communities (e. g., OLAC[3] for resources related to language and linguistics, CLARIN[4] and ELRA[5] for language resources, Europeana[6] for cultural works,

[1] https://european-language-equality.eu

[2] For the sake of brevity, we will use the cover term "catalogue" for all institutions of this kind.

[3] http://www.language-archives.org

[4] https://www.clarin.eu

[5] http://elra.info

[6] https://www.europeana.eu

ELIXIR[7] for bioinformatics, LLOD cloud[8] for linguistic linked data, etc.), catalogues that collect specific content types (e. g., Hugging Face[9] for Machine Learning models and datasets, ELRC-SHARE[10] for Machine Translation-related resources or portals for open government data).[11]

At the same time, we witness a strong movement towards the sharing of resources from multiple sources and various disciplines through a common point of access, so that they are easily discoverable, accessible and re-usable by all interested stakeholders, fostering interdisciplinary research and cross-community collaborations as well as Open Science (e. g., European Commission 2022). Google has implemented its Dataset Search[12], a service dedicated to facilitating the discovery of datasets stored across the World Wide Web based on keyword search (Benjelloun et al. 2020). The European Open Science Cloud (EOSC)[13], initiated by the European Commission, is conceived as a federated and open multi-disciplinary environment for hosting and processing research data and all other digital objects produced along the research life cycle, e. g., methods, software and publications (Abramatic et al. 2021). Some European countries have launched corresponding national initiatives, including the National Research Data Infrastructure in Germany (NFDI).[14] Gaia-X[15] seeks to establish a federated ecosystem in which data is made available, collated, shared and processed in trustworthy environments, associated with the concept of data spaces, a type of data relationship between trusted partners, each of whom apply the same high policies, standards and technical components to the description, storage and sharing of their data and other resources.

All these initiatives offer catalogues, or inventories, employing, in many cases, different metadata schemas for the description of resources. The differences between the schemas can be attributed to the varying requirements defined by the relevant object of description (e. g., dataset vs. software or publication or geospatial data), the need to cover a wide range of users (for general catalogues) in contrast to the specialised practices common among scholars of a discipline, as well as to the different purposes that catalogues may serve (e. g., preservation, dissemination, or processing). Sharing metadata across catalogues presupposes interoperability, in particular, *semantic* interoperability. Initiatives for the adoption of common standards in metadata vocabularies, documentation of the vocabularies themselves, and the creation and publication of mappers between them are among the primary instruments to achieve such interoperability (Chan and Zeng 2006; Zeng and Chan 2006; Haslhofer and Klas 2010; Alemu et al. 2012; Broeder et al. 2019).

[7] https://elixir-europe.org

[8] https://linguistic-lod.org/llod-cloud

[9] https://huggingface.co

[10] https://www.elrc-share.eu

[11] https://www.re3data.org/browse/ provides a registry of research data repositories.

[12] https://datasetsearch.research.google.com

[13] https://eosc-portal.eu

[14] https://www.nfdi.de

[15] https://www.gaia-x.eu

Equally important is the establishment of protocols and mechanisms for the sharing of metadata, and subsequently of the resources themselves. The OAI-PMH protocol[16] is one of the most popular mechanisms used for repository interoperability at the metadata level. The ResourceSync[17] specification is a framework for the synchronisation of both metadata and resources. Finally, APIs are frequently offered nowadays as a solution for downloading dumps of metadata records.

ELG has established technical bridges with other infrastructures and initiatives in order to enrich its catalogue with information about data resources and tools from other catalogues and repositories. The catalogues of interest to ELG are usually discipline-specific, targeting the LT/NLP and neighbouring areas, such as Machine Learning, Artificial Intelligence as well as social sciences and humanities. Potentially interesting resources for LT development purposes are also hosted in general repositories and catalogues, the identification and filtering of which poses challenges which are briefly discussed in Section 3.

3 Establishing Interoperable Connections: Four Use Cases

Depending on the source repositories' respective contents, metadata schemas and vocabularies, and the available export functionalities of their catalogues, we have adopted different approaches towards establishing interoperable connections, a selection of which is presented in the following use cases. For each use case, we describe the source repository's technical and metadata features, explain how these impact the import of metadata records into ELG and present the methodology and tools used in the integration process.

3.1 Use Case 1: OAI-PMH (CLARIN Nodes and ELRC-SHARE)

The CLARIN (Common Language Resources and Technology Infrastructure) Research Infrastructure (Hinrichs and Krauwer 2014; Eskevich et al. 2020) supports the sharing, use and sustainability of digital language resources and tools for research in the social sciences and humanities. It is established in the form of a networked federation of centres (Wittenburg et al. 2010), consisting of language data repositories, service centres and knowledge centres, with single sign-on access for all members of the academic community in all participating countries.

As part of the technical interoperability specifications, CLARIN data repositories are required to expose their metadata records to the Virtual Language Observatory[18] using OAI-PMH. With regard to metadata interoperability, CLARIN has designed

[16] https://www.openarchives.org/pmh/

[17] http://www.openarchives.org/rs/1.1/resourcesync

[18] https://vlo.clarin.eu

and implemented the Component MetaData Infrastructure (CMDI)[19], a framework for the description and reuse of metadata "components" (semantic groups of elements) which can be combined to build "profiles", i. e., metadata templates for specific resource types by specific communities or groups (Broeder et al. 2008, 2012). Both are stored and shared through a dedicated registry, with metadata records being shared in the form of XML files compatible with one of these profiles.

The ELG platform implements an OAI-PMH client for harvesting metadata from external repositories which expose their metadata via OAI-PMH. The process of harvesting requires the registration of a third-party provider as an "OAI-PMH Provider" in the ELG catalogue. As soon as communication is established, the third-party provider shares their OAI-PMH endpoint, which ELG will call at regular intervals (currently once a week) in order to harvest the metadata the external repository exposes. Thus, for linking with the CLARIN infrastructure, the OAI-PMH harvesting protocol is the ideal candidate.

The metadata schema is a crucial parameter to be taken into account in the harvesting process. The ELG harvester accepts metadata records compliant with the minimal version of the ELG metadata schema (see Section 5 in Chapter 2). LINDAT/CLARIAH-CZ[20], the Czech CLARIN national node, does indeed expose its metadata records described using the META-SHARE minimal schema through its OAI-PMH endpoint (Gavrilidou et al. 2012). The fact that the ELG schema (Labropoulou et al. 2020) builds upon META-SHARE proved valuable in the conversion process of the original LINDAT/CLARIAH-CZ metadata into the ELG schema (see Chapter 8, Section 4, p. 157 ff., for more technical details).

CLARIN-DSpace, the repository software[21] (forked from DSpace[22]) developed mainly by the LINDAT/CLARIAH-CZ team, is used by several CLARIN centres for their repositories (Straňák et al. 2019). After pulling the latest changes, these repositories are ready-to-import into ELG using the same harvesting mechanism and procedure. At the time of writing, the mechanism described above is also used for harvesting CLARIN-PL[23] and CLARIN-SI[24].

The same harvesting approach was followed for the harvesting of metadata records from the ELRC-SHARE repository, which is used for the storage of and access to language resources collected through the European Language Resource Coordination[25] initiative (Lösch et al. 2018) and for feeding the CEF Automated Translation (CEF.AT) platform.[26] ELRC-SHARE (Piperidis et al. 2018) uses a metadata schema based on the META-SHARE schema tuned to text resources for Machine

[19] https://www.clarin.eu/content/component-metadata

[20] https://lindat.mff.cuni.cz

[21] https://github.com/ufal/clarin-dspace

[22] https://duraspace.org/dspace/

[23] https://clarin-pl.eu/dspace/

[24] https://www.clarin.si/repository/xmlui/?locale-attribute=en

[25] https://lr-coordination.eu

[26] https://ec.europa.eu/cefdigital/wiki/display/CEFDIGITAL/eTranslation

Translation purposes. Again, the mapping of the metadata records from the original schema to ELG was undertaken by the two teams.

3.2 Use Case 2: Custom API and Proprietary Schema (Hugging Face)

A different procedure is used for catalogues that expose metadata records through custom APIs and proprietary metadata schemas. This procedure is used only for catalogues that are of high interest to the ELG objectives. The Hugging Face catalogue (Wolf et al. 2020) is such a case. It is a large collection of machine learning models and datasets that can be used for training models, with a focus on the Transformer architecture. Since 2021 ELG and Hugging Face have been collaborating with the goal of importing metadata records from the Hugging Face catalogue into ELG.

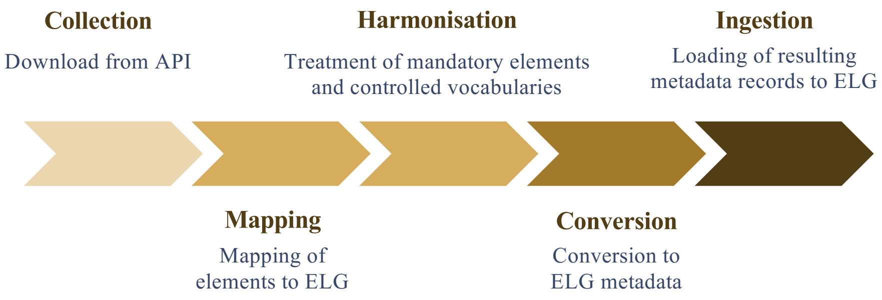

Collection
Download from API
Harmonisation
Treatment of mandatory elements and controlled vocabularies
Ingestion
Loading of resulting metadata records to ELG
Mapping
Mapping of elements to ELG
Conversion
Conversion to ELG metadata

Fig. 1 Workflow for the import of metadata records from Hugging Face to ELG

One of the goals of Hugging Face is to enable its users to upload datasets and models following a set of specifications so that they can be deployed for testing and building other models or integrating models in their applications. Although they encourage users to add descriptions for the resources, this is not enforced. Furthermore, the suggested metadata elements do not follow a standard schema. Users are asked to upload a "card" for datasets[27] or models[28], with a combination of free text fields and a set of tags (e. g., language, licence) with values from recommended controlled vocabularies, which are, however, not strictly validated.

Hugging Face exposes two APIs with JSON files for datasets and models respectively. These JSON files include a subset of the metadata elements displayed in their catalogue, however, not all records have values for all of the elements. Since importing into ELG presupposes that the metadata records comply with the ELG metadata schema, which means that at least the mandatory elements of the minimal version (see Section 5 in Chapter 2) are filled in, the conversion and import of records from Hugging Face into ELG has so far been limited to datasets with at least the de-

[27] https://huggingface.co/docs/datasets/dataset_card.html

[28] https://huggingface.co/docs/hub/model-repos

scription, language and licence elements filled in as these are deemed the minimum threshold for findability and usability purposes in the context of ELG.

A conversion process has been set up based on the mapping of the elements and, in the case of controlled vocabularies, their values. Further enrichment of the resulting records has been performed for specific elements. The most prominent case was that of the licencing information, since ELG requires, besides its name, a URL with the text of the licence. Hugging Face includes a list of licence identifiers taken from the SPDX list[29] (which are also used in ELG), but it allows users as well to add a licence name without further information. Thus, in addition to the mappings of the licence identifiers from Hugging Face into the ones used in ELG, we looked for the licence URL of unmapped values; if no URL was found, the resource was not imported into ELG. Finally, where required, default values have been used for mandatory elements whose values could not be inferred from the original metadata records (e. g., all datasets have been assigned the text value for media type). Figure 1 shows the workflow that was followed in this process.

3.3 Use Case 3: General Catalogues and Standard Schemas (Zenodo)

Catalogues with heterogeneous resources from multiple sources and disciplines present various challenges. We use Zenodo[30] to discuss these challenges.

Zenodo[31] is a repository for storing and sharing EC-funded research results to support Open Science established and run by CERN, which was created in response to the European Commission's (EC) assignment to the OpenAIRE project.[32]. Since its launch, Zenodo has grown steadily and is currently used for the publication of all types of resources beyond EC-funded ones by research communities and individuals. The constant update of the Zenodo catalogue and its uptake by researchers for the upload of datasets, and, more recently, software, makes it particularly interesting for ELG purposes. The size and increasing number, however, of catalogue contents makes the selection of resources very challenging. During the first phase of the ELG project, we used a manual process for the identification of resources, which is described in Chapter 8. This process, though, does not allow for regular updates and has been abandoned in favour of an automatic process.

[29] https://spdx.org/licenses/

[30] https://zenodo.org

[31] https://about.zenodo.org

[32] https://www.openaire.eu

Zenodo exposes its metadata records through two channels: a REST API[33], which outputs records as JSON files, and an OAI-PMH API[34] in a set of standard metadata formats, i. e., DC[35], DataCite[36], MARC21[37] and DCAT[38].

With regard to the ELG import mechanism, our preferred solution is OAI-PMH, a standard protocol for interoperability and exchange of metadata records, which includes a mechanism for regular harvesting. However, the Zenodo OAI-PMH endpoint does not allow the selection based on resource types, which would allow us to focus on "datasets" and "software". The only option is to download the whole set of metadata records in order to subsequently filter them. Furthermore, harvesting from the OAI-PMH endpoint is rate limited, hence not appropriate for large numbers of metadata records. We have, therefore, resorted to a combined solution:

- We downloaded a full dump of 2,060,674 metadata records included in Zenodo up until 31 August 2021. This dump, which is available from Zenodo, contains all records in JSON format, was filtered according to resource-type.
- For records added to Zenodo after this date, we are incrementally harvesting from the OAI-PMH endpoint. Through this channel, a set of additional 147,621 records has been harvested in a three-month period.

The next step is that of identifying the candidate resources for ELG. From the 2,208,295 metadata records available up until 31 December 2021, those of resource type "dataset" and "software" amount to 592,509 entries. This number is rather high, and since the majority of these records are of little or no interest to ELG users[39], we are experimenting with automated filtering methods to identify the records of interest.

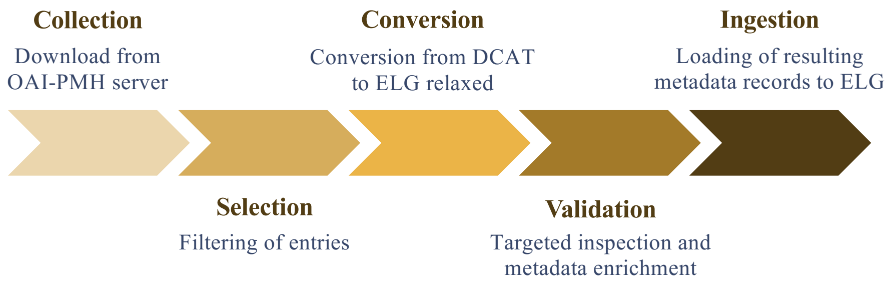

Collection
Download from OAI-PMH server
Conversion
Conversion from DCAT to ELG relaxed
Ingestion
Loading of resulting metadata records to ELG
Selection
Filtering of entries
Validation
Targeted inspection and metadata enrichment

Fig. 2 Workflow for the import of metadata records from Zenodo to ELG

33 https://developers.zenodo.org/#rest-api
34 https://developers.zenodo.org/#oai-pmh
35 https://www.dublincore.org/specifications/dublin-core/dcmi-terms/
36 https://schema.datacite.org/meta/kernel-4.4/
37 https://www.loc.gov/marc/bibliographic/
38 https://www.w3.org/TR/vocab-dcat-3/
39 As a comparison, the ELG catalogue has approx. 13,000 metadata records at the time of writing.

The conversion of the metadata records is based on the DCAT metadata schema (Albertoni et al. 2022), which is in widespread use. We expect that mapping DCAT to ELG will enable the re-use of these converters as a base for import from other repositories. Moreover, DCAT is the schema with the richest information among the ones exposed from Zenodo, and the only one that includes a direct link to the downloadable files ("downloadURL" element), an important feature for ELG consumers.

Mapping from DCAT is, however, not straightforward. DCAT is an RDF vocabulary, and restrictions and extensions are implemented in the form of profiles and applications. The OAI-PMH endpoint makes the metadata records available in XML format; the XSD schema used by Zenodo is not publicly available[40]. A closer inspection of the XML files has revealed discrepancies in the representation of some elements. For instance, "subject" (defined in DCAT as a SKOS[41] Concept) appears in Zenodo XML files either as a SKOS Concept or as an element with the IRI of the subject value in the form of an attribute. We have analysed the Zenodo XML files, to the extent possible, and based our mapping on this analysis. We also had to apply some modifications in the ELG schema so that we could take into account the DCAT features (Section 4.1). Finally, a converter for the elements in the JSON files offered through the REST API for the first batch of files has also been implemented.

As a result of this endeavour, the procedure for regular updates from Zenodo is foreseen as a workflow integrating the following steps: harvesting from the Zenodo endpoint, offline filtering and conversion of the metadata records, possibly with some manual targeted inspection, and import into ELG (Figure 2).

3.4 Use Case 4: Collaborative Community Initiatives (ELE, ELG)

We also populated the ELG catalogue using bulk lists of metadata records, potentially containing limited information, that serve as seeds for further enrichment. We present here two such cases, one set of resources collected collaboratively in ELE and a second set collected by the ELG consortium.

The European Language Equality (ELE) project (Rehm and Way 2023)[42], which collaborates with ELG to promote digital language equality in Europe, launched a project-internal initiative in 2021 to collect as many LRTs as possible available for the languages under investigation by the project.[43] Operationally, a web form was set up, which included a subset of the mandatory metadata elements of the ELG schema. Given the size and breadth of this activity (dozens of respondents throughout Europe for approx. 80 official, regional, minority languages), we considered requiring every informant to fill in even the minimal version of the metadata schema for every single resource identified too demanding and not paricularly realistic, perhaps

[40] The XSD schema included in the OAI-PMH API for DCAT is in fact that of DataCite v4.1.

[41] https://www.w3.org/2004/02/skos/

[42] https://european-language-equality.eu

[43] https://european-language-equality.eu/languages/

even negatively impacting the collection process itself, potentially resulting in fewer resources being reported by the informants if the process of registering a resource took too much time. The modifications required to accommodate this collaborative scenario resulted in a "relaxed" version of the schema (see Section 4.1).

The results of this collection process were exported in a tabular format. Before the conversion and final import of the approx. 6,500 records into ELG, a long and demanding process of curation was undertaken using semi-automatic methods. The final output was imported into ELG through various scripts (Figure 3).

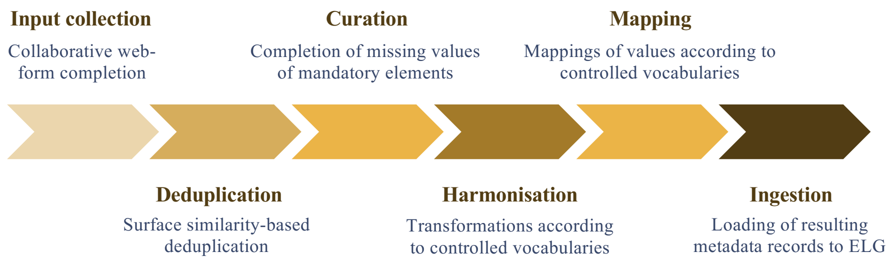

Input collection
Collaborative web-form completion
Curation
Completion of missing values of mandatory elements
Mapping
Mappings of values according to controlled vocabularies
Deduplication
Surface similarity-based deduplication
Harmonisation
Transformations according to controlled vocabularies
Ingestion
Loading of resulting metadata records to ELG

Fig. 3 Workflow for the import of ELE results to ELG

The curation process included normalising, correcting, and enriching values of elements that were absent or not used consistently. Despite the effort to control the input through prompting for the selection of values from recommended vocabularies and filling in mandatory values, web forms do not allow strict enforcement strategies, especially for cases of long lists of values or multiple values. For example, although a set of "language" values was offered for selection in the form, the informants could also add other values, which resulted in values with alternative, unofficial or simply unusual names. Therefore, language information had to be normalised and mapped to the ISO 639 language codes, as required by ELG. Although the tabular format presents some advantages, given its simplicity and users' familiarity, it still poses a number of challenges for validation purposes, especially for elements with patterns, or with multiple values. For instance, the "email" element was filled in with free text values, URL links, etc., since no validation pattern was used for the element. For elements with multiple values, such as languages, functions, etc., different delimiters were used in between values and had to be normalised. Moreover, nested information cannot be represented in a flat form; for example, the values of language and region (where the language is spoken) were split in two complementary columns so that controlled vocabularies could be used, but there can be no guarantee that both columns are consistently filled in. For these cases, we had to check and ensure that the same number of values was consistently used across the two complementary columns and, moreover, that the values were matched correctly.

In a similar collaborative population setting, the catalogue was populated with European organisations that develop or use LTs or LRs, which were collected by the ELG team and the National Competence Centres (NCCs; see Chapter 11 for more details), thus enabling ELG to quickly become the "yellow pages" of organisations active in the broader LT community. As described in more detail in Chapter 9, lists of organisations from various sources have been merged, together with information on list items – mainly contact data and key terms describing their LT-related activities. The resulting enriched list, divided into sub-lists by country, was checked again by the respective NCCs, and, after checking the consistency, more than 1,700 records were converted into the ELG-compatible XML format and imported into ELG. At the time of writing, a similar procedure is being followed for LT-related R&D projects and their funding agencies.

3.5 Summary of Use Cases

Table 1 summarises the technical and the metadata conditions in each of the use cases presented in this section and the ways these are catered for in ELG. Depending on the export functionalities offered by the source, the ELG platform can establish a connection at regular intervals and benefit from continuous updates. Table 1 also shows the ELG metadata schema version that can be used, depending on the source metadata schema, as well as the quantity and information richness of metadata records.

Repository	**Export Functionality**	**Metadata Schema**	**ELG Schema Version**	**Update Frequency**
CLARIN nodes	OAI-PMH	META-SHARE	minimal	regular
ELRC-SHARE	OAI-PMH	ELRC-SHARE	minimal	regular
Hugging Face	REST API	Proprietary (JSON)	relaxed	one-off
Zenodo	REST API	Proprietary (JSON)	relaxed	one-off
Zenodo	OAI-PMH	DCAT (XML)	relaxed	regular
ELE survey	–	Subset of ELG schema	relaxed	one-off
ELG collection	–	Subset of ELG schema	relaxed	one-off

Table 1 Overview of use cases

4 Implementing Metadata Interoperability

Primarily motivated by our various interoperability use cases, some of which are described in Section 3, we modified the ELG platform import procedures and policies, especially with regard to the metadata schema and the publication life cycle (described in Chapter 2), so that they are able to handle the different interoperability scenarios. These adaptations are not restricted to the requirements of the use cases but lay the foundational principles for accommodating a broader range of metadata import scenarios.

4.1 ELG Metadata Schema – Relaxed Version

The "relaxed" version of the ELG metadata schema aims to accommodate mismatches between the ELG schema and schemas used for metadata records that are automatically imported into the ELG catalogue, especially those from catalogues with limited information or catalogues populated with metadata records of interest to a broader range of communities (e. g., Zenodo, EOSC, etc.) and, thus, using more general schemas, e. g., DCAT (Albertoni et al. 2022) or DataCite[44] (DataCite Metadata Working Group 2021). This version of the schema features additional alternative elements for mandatory metadata elements that may be missing from the source records or that have different data types.

The first case refers to two elements that are deemed important for ELG purposes: "media type" and "licence".

- The element "media type part" is crucial for ELG, as it is used for attaching important metadata properties, such as language, format, size, etc. Even in cases where these are included in source records, they may come with *different* classification vocabularies and semantics and, therefore, cannot be imported into ELG. For these cases, the additional alternative value "unspecified media part" can be used.
- The element "licence" is crucial for re-usability purposes; for a licence, both a name and a URL hyperlink to the respective legal document are required. However, in many cases, such as legacy resources, or records in catalogues allowing free text as the value of "licence", the name and URL cannot be determined automatically. This is why we introduced the "access rights" element that takes a free text value as an alternative to "licence", specifying the rights of access and use at a higher level of abstraction.

The second case groups together elements which take a value from controlled vocabularies in ELG, while in other schemas they have a free text value (e. g., "service function", "size unit", etc.) and combined elements that cannot be distinguished from the source metadata record (e. g., when size is encoded as free text combining amount and size unit together). To address the first case, we modified the data type of the element so that it takes a value from a recommended vocabulary or free text entered by the user; to address the second case, we introduced a new element that takes free text as a value (e. g., "sizeText" can be used as an alternative to the combination of "amount" and "size unit").

4.2 Publication Policies for Imported Metadata Records

ELG rates the quality of the metadata records highly. High quality metadata contributes to the discovery and usage of the resources themselves. A standardised pub-

[44] https://schema.datacite.org

lication life cycle has been established in ELG for metadata records (see Chapter 2, Section 6, 24 ff.). However, the same level of quality cannot be enforced across all metadata records. This is also taken into account in the publication policies. Thus, while metadata records registered by individuals go through a validation process, for records automatically imported from other catalogues the same manual validation processes cannot be set up in a feasible way, i. e., the quality and extent, in terms of information, of external metadata records remains under the responsibility of the respective source catalogue. Depending on the harvesting process and source catalogue, a three-level classification of metadata records is used:

- *Metadata records harvested automatically from collaborating catalogues (CLARIN nodes, ELRC-SHARE)*, which have similar metadata requirements as ELG. These records are added by individuals, the resource is stored in the repository. This is why these metadata records are considered trustworthy, and the records are published in the ELG catalogue as is, i. e., without any human validation.
- *Metadata records automatically imported from catalogues with "lighter" metadata requirements (Hugging Face, Zenodo)* have originally been added to the source catalogue by individuals together with the physical resource. The metadata record and resource is considered trustworthy but it may lack information which is important for ELG purposes, and thus marked as "for information" to indicate to ELG users that important information may be missing.
- *Metadata records that resulted from bulk collection initiatives (ELE collection, ELG collection)* are often incomplete, i. e., only a subset of the required information was collected and converted to the ELG schema. These records adhere to the relaxed ELG schema, the physical resource may be stored anywhere online. These records do not undergo the validation process, they are marked and can be claimed for further enrichment by their rightful owners (see Chapter 9, Section 3.3, p. 179). When a user claims a metadata record, the technical ELG team is notified and can approve or reject the claim, taking into account the professional email account of the user; if the claim is approved, the metadata record is unpublished and assigned to the user for further editing. Once the user finishes the editing, the record is submitted for publication and goes through the normal publication procedure. Users are notified about the claim procedure of these metadata records via e-mail.

5 Interoperability across Repositories

The interoperability across multiple repositories and platforms is of utmost importance in a broader, federated environment of data and services, as envisaged in initiatives like EOSC (European Open Science Cloud, see, e. g., Corcho et al. 2021), NFDI, Gaia-X or the European Commission's Data Spaces and in accordance with the FAIR principles (Wilkinson et al. 2016), see Section 2. In the following, we discuss some of the open issues that need to be addressed in order to achieve this based on the endeavours presented in this chapter.

5.1 Technical Interoperability across Repositories

The first prerequisite for the sharing of metadata records and the construction of a common master inventory based on the contents of all participating repositories is that of exchange services. The OAI-PMH protocol, despite its limitation to the exchange of metadata, constitutes the most widespread and hence usually preferred option. REST services are becoming more popular, but they are not yet standardised and thus require customised solutions. Rehm et al. (2020a) explore technical and semantic interoperability in more detail.

5.2 Semantic Interoperability across Repositories

The use of shared vocabularies for the documentation of resources is the next necessary step towards interoperability. The standardisation and documentation of metadata schemas is a requirement that many initiatives have articulated (Hugo et al. 2020; Behnke et al. 2021). While certain metadata vocabularies, such as DC[45], DCAT, schema.org[46] and DataCite, have become de facto standards, these are general schemas that can be used to express core metadata elements required for the description of any type of digital resource. This, however, competes with the much more fine-grained documentation needs of specific communities and more detailed requirements set to achieve machine actionability. For example, "resource type" is an element that poses problems for all catalogues: in contrast to the general vocabularies (e. g., COAR resource type vocabulary[47], a limited set of values from DC[48], Zenodo[49]), communities prefer finer distinctions (cf. the values of "resource type" in the CLARIN VLO[50]). This creates a burden when moving from general to specialised catalogues (e. g., from Zenodo to ELG).

Bridges and mappers between vocabularies are developed, especially between the popular schemas.[51] Yet this is not a scalable approach, as for each new vocabulary a new mapper has to be built. Instead, a "shared semantic space" is needed as a joint, ontologically grounded and machine-readable vocabulary, into which all concepts and terminologies can be mapped (Rehm et al. 2020a). This space can be envisaged as a reference model able to represent all crucial information typically contained in the respective metadata schema. However, a single RDF/OWL ontology covering general and domain or community-specific semantic categories is an almost impossible task to achieve (Labropoulou et al. 2018). An alternative could be a Linked

[45] https://www.dublincore.org/specifications/dublin-core/dcmi-terms/

[46] https://schema.org

[47] https://vocabularies.coar-repositories.org/resource_types/

[48] https://www.dublincore.org/specifications/dublin-core/resource-typelist/

[49] https://developers.zenodo.org/#representation

[50] https://vlo.clarin.eu

[51] For the mapping of metadata schemas in the wider LT ecosystem, see McCrae et al. (2015b,a).

Data approach[52], in which different communities maintain their independent formal models and vocabularies and subsequently refer to reference vocabularies or concepts developed in a distributed fashion by the broader community. As an example of such an approach, a collaboration was initiated between ELG and the AI4EU project on the mapping of the ontologies used in the two platforms. This work is continued under the umbrella of the AI Ontology Working Group which includes members from the European AI on Demand Platform and collaborating projects.[53]

Even in this scenario, though, an important issue to be addressed is that of the appropriate semantic relations. Equivalence relations are not always one-to-one and also need to take into account the type of elements. Additionally, there are an abundance of similar vocabularies recommended by different communities or serving different documentation needs. For example, in terms of "language", a value taken from ISO 639[54] may suffice for general catalogues. But for the metadata of resources in language-related catalogues, such as ELG, a more detailed value space is required, that takes into account dialects and other varieties, and these are not included in ISO 639 (Gillis-Webber and Tittel 2019). In ELG we use the BCP 47 recommendation (Phillips and Davis 2009) alongside values taken from the Glottolog[55] vocabulary (Hammarström et al. 2021) so that we can exploit the finer distinctions made in it for language varieties. The fact that Glottolog includes a mapping to ISO 639-3 values, when these exist, facilitates this endeavour and the exchange of metadata records with catalogues that prefer using ISO 639.

5.3 Minimal Metadata Requirements

The different purposes served by the catalogues have an impact on the exchange of metadata records, too. For example, Zenodo is used for the publication of research outcomes by many different organisations and individuals. The fact that there is a very small set of mandatory elements as well as the fact that providers do not have a strong incentive to make their resources findable lowers the quality of the metadata descriptions. In a similar way, individuals that add their resources to the Hugging Face catalogue are mostly interested in testing their dataset and do not pay attention to its description. Many metadata elements that are important for ELG purposes, such as "language", are simply not included in the formal descriptions of these records. Often, even free text descriptions are of very low quality and cannot be used for discovery purposes. There is, therefore, a strong need for training resource owners on the importance of metadata together with the continuous curation by experts (Gordon and Habermann 2019). The "claim" procedure adopted in ELG is a step along these lines. Semi-automatic methods for enriching metadata records by extracting

[52] https://www.w3.org/DesignIssues/LinkedData.html

[53] https://www.ai4europe.eu/ai-community/working-groups-d/ontology

[54] https://www.iso.org/iso-639-language-codes.html

[55] https://glottolog.org

information from the datasets themselves, as well as other sources, will also play an important role in ensuring that minimal documentation requirements are met.

5.4 Duplicate Resources

Looking at the resources themselves, the exchange of metadata records across catalogues comes with the risk of creating duplicates and near-duplicates. The same resource may appear with slightly different names in catalogues and similar descriptions, while the same name is often used for subsets of the resource. The use of persistent identifiers (PIDs) has been proposed to address this, but it cannot be guaranteed that persistent identifiers are indeed unique. Explicit relations between similar resources (subsets, raw or annotated versions, versions and updates, etc.) must be formally recorded in the metadata so that they can be used for deduplication purposes. Establishing relations between the metadata records of the same resource in different catalogues should also be recorded.

6 Conclusions

In this chapter we have focused on the sharing of metadata between catalogues. This is only the basis for what is going to be the next level of sharing data and software which is the ultimate goal. This involves not only a shared semantic space to anchor and cross-link metadata vocabularies but also technical compatibility and cooperation. ELG has closely collaborated with other platforms to explore platform interoperability at various levels (Rehm et al. 2020a). Experiments were conducted with AI4EU[56], SPEAKER[57] and QURATOR[58] for the creation of cross-platform workflows, where data and services were accessed from one platform and either transferred to another platform or used for building a pipeline or workflow of different processing services in another platform. Our initial experiments, explored further by Moreno-Schneider et al. (2022), demonstrate that interoperability can be partially achieved, with a certain degree of manual and automatic interventions.

Finally, we should also mention an alternative that can be used for sharing resources and their documentations across platforms and communities. This consists of supporting cross-platform search through making search and discovery APIs used by a platform available to third parties so that they can integrate them in their own search space (Rehm et al. 2020a). This way, a single query would return matches from multiple platforms whose publicly available search APIs are integrated in the platform queried by the user. In this case, search results would show only a minimal

[56] https://www.ai4europe.eu

[57] https://www.speaker.fraunhofer.de

[58] https://qurator.ai

set of metadata redirecting the user to the platform that offers the respective resource. Again, a shared common space is required but only for a limited set of metadata – a similar situation to the general catalogues presented above, but only for a small subset. However, this option presents a scalability problem as soon as the number of collaborating platforms and respective search APIs grows.

Decentralised infrastructures such as Gaia-X, in which individual trusted platforms follow a common standard (i. e., the Gaia-X federation services) and become a networked system freely sharing and exchanging data and services across multiple actors, offer a viable solution addressing this challenge. OpenGPT-X[59] is a German national project in which large language models are currently being developed, especially for German but also for English and other European languages. In this project, which has started in January 2022, we will have the chance to implement the emerging Gaia-X specifications in the ELG platform so that it joins this emerging ecosystem.

References

Abramatic, Jean-François, Jan Hrušák, and Sarah Jones, eds. (2021). *European Open Science Cloud (EOSC) Executive Board: Final Progress Report*. Publications Office. DOI: 10.2777/46019.

Albertoni, Riccardo, David Browning, Simon Cox, Alejandra Gonzalez-Beltran, Andrea Perego, and Peter Winstanley, eds. (2022). *Data Catalog Vocabulary (DCAT) – Version 3*. W3C Working Draft. URL: https://www.w3.org/TR/vocab-dcat-3/.

Alemu, Getaneh, Brett Stevens, and Penny Ross (2012). "Towards a conceptual framework for user-driven semantic metadata interoperability in digital libraries: a social constructivist approach". In: *New Library World* 113.1/2, p. 15.

Behnke, Claudia, Kees Burger, Yann le Franc, Wim Hugo, Pekka Järveläinen, Jessica Parland-von Essen, and Gerard Coen (2021). "D2.6 First reference implementation of the data repositories features". In: DOI: 10.5281/zenodo.5362027. URL: https://zenodo.org/record/5362027/export/hx.

Benjelloun, Omar, Shiyu Chen, and Natasha Noy (2020). "Google Dataset Search by the Numbers". In: *The Semantic Web (ISWC 2020) – 19th International Semantic Web Conference*. Ed. by Jeff Z. Pan, Valentina A. M. Tamma, Claudia d'Amato, Krzysztof Janowicz, Bo Fu, Axel Polleres, Oshani Seneviratne, and Lalana Kagal. Vol. 12507. Lecture Notes in Computer Science. Athens, Greece: Springer, pp. 667–682. DOI: 10.1007/978-3-030-62466-8_41. URL: https://doi.org/10.1007/978-3-030-62466-8_41.

Broeder, Daan, Thierry Declerck, Erhard Hinrichs, Stelios Piperidis, Laurent Romary, Nicoletta Calzolari, and Peter Wittenburg (2008). "Foundation of a Component-based Flexible Registry for Language Resources and Technology". In: *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC 2008)*. Marrakech, Morocco: ELRA. URL: http://www.lrec-conf.org/proceedings/lrec2008/pdf/364_paper.pdf.

Broeder, Daan, Thorsten Trippel, Emiliano Degl'Innocenti, Roberta Giacomi, Maurizio Sanesi, Mari Kleemola, Katja Moilanen, Henri Ala-Lahti, Caspar Jordan, Iris Alfredsson, Hervé L'Hours, and Matej Ďurčo (2019). "SSHOC D3.1 Report on SSHOC (meta)data interoperability problems". In: DOI: 10.5281/ZENODO.3569868. URL: https://zenodo.org/record/3569868.

Broeder, Daan, Dieter van Uytvanck, Maria Gavrilidou, Thorsten Trippel, and Menzo Windhouwer (2012). "Standardizing a Component Metadata Infrastructure". In: *Proceedings of the Eighth In-*

[59] https://opengpt-x.de

ternational Conference on Language Resources and Evaluation (LREC 2012). Istanbul, Turkey: ELRA, pp. 1387–1390. URL: http://www.lrec-conf.org/proceedings/lrec2012/pdf/581_Paper.pdf.

Calzolari, Nicoletta, Valeria Quochi, and Claudia Soria, eds. (2011). *The Strategic Language Resource Agenda*. URL: https://www.academia.edu/1651334/The_Strategic_Language_Resource_Agenda.

Chan, Lois Mai and Marcia Lei Zeng (2006). "Metadata Interoperability and Standardization – A Study of Methodology Part I: Achieving Interoperability at the Schema Level". In: *D-Lib Magazine* 12.6. DOI: 10.1045/june2006-chan. URL: http://www.dlib.org/dlib/june06/chan/06chan.html.

Corcho, Oscar, Magnus Eriksson, Krzysztof Kurowski, Milan Ojsteršek, Christine Choirat, Mark van de Sanden, Frederik Coppens, and European Commission, Directorate-General for Research and Innovation (2021). *EOSC Interoperability Framework: Report from the EOSC Executive Board Working Groups FAIR and Architecture*. Publications Office. DOI: 10.2777/620649. URL: https://data.europa.eu/doi/10.2777/620649.

DataCite Metadata Working Group (2021). "DataCite Metadata Schema Documentation for the Publication and Citation of Research Data and Other Research Outputs v4.4". In: DOI: 10.14454/3W3Z-SA82. URL: https://schema.datacite.org/meta/kernel-4.4/.

ELRC (2019). *ELRC White Paper: Sustainable Language Data Sharing to Support Language Equality in Multilingual Europe*. Second online edition. URL: https://lr-coordination.eu/sites/default/files/Documents/ELRCWhitePaper.pdf.

Eskevich, Maria, Franciska de Jong, Alexander König, Darja Fišer, Dieter Van Uytvanck, Tero Aalto, Lars Borin, Olga Gerassimenko, Jan Hajic, Henk van den Heuvel, Neeme Kahusk, Krista Liin, Martin Matthiesen, Stelios Piperidis, and Kadri Vider (2020). "CLARIN: Distributed Language Resources and Technology in a European Infrastructure". In: *Proc. of the 1st Int. Workshop on Language Technology Platforms (IWLTP 2020, co-located with LREC 2020)*. Ed. by Georg Rehm, Kalina Bontcheva, Khalid Choukri, Jan Hajic, Stelios Piperidis, and Andrejs Vasiljevs. Marseille, France: ELRA, pp. 28–34. URL: https://aclanthology.org/2020.iwltp-1.5.

European Commission (2022). *European Research Area policy agenda: overview of actions for the period 2022–2024*. Publications Office. DOI: 10.2777/52110. URL: https://data.europa.eu/doi/10.2777/52110.

European Parliament (2018). *Language Equality in the Digital Age. European Parliament resolution of 11 September 2018 on Language Equality in the Digital Age (2018/2028(INI)*. URL: http://www.europarl.europa.eu/doceo/document/TA-8-2018-0332_EN.pdf.

Gavrilidou, Maria, Penny Labropoulou, Elina Desipri, Stelios Piperidis, Haris Papageorgiou, Monica Monachini, Francesca Frontini, Thierry Declerck, Gil Francopoulo, Victoria Arranz, and Valerie Mapelli (2012). "The META-SHARE Metadata Schema for the Description of Language Resources". In: *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC 2012)*. Istanbul, Turkey: ELRA, pp. 1090–1097. URL: http://www.lrec-conf.org/proceedings/lrec2012/pdf/998_Paper.pdf.

Giagkou, Maria, Stelios Piperidis, Georg Rehm, and Jane Dunne, eds. (2022). *Language Technology Support of Europe's Languages in 2020/2021*. Various project deliverables (language reports); EU project European Language Equality (ELE); Grant Agreement no. LC-01641480 – 101018166 ELE. European Language Equality Project. URL: https://european-language-equality.eu/deliverables/.

Gillis-Webber, Frances and Sabine Tittel (2019). "The Shortcomings of Language Tags for Linked Data When Modeling Lesser-Known Languages". In: *2nd Conference on Language, Data and Knowledge (LDK 2019)*. Ed. by Maria Eskevich, Gerard de Melo, Christian Fäth, John P. McCrae, Paul Buitelaar, Christian Chiarcos, Bettina Klimek, and Milan Dojchinovski. Vol. 70. OpenAccess Series in Informatics (OASIcs). Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 4:1–4:15. DOI: 10.4230/OASIcs.LDK.2019.4. URL: http://drops.dagstuhl.de/opus/volltexte/2019/10368.

Gordon, Sean and Ted Habermann (2019). *Visualizing The Evolution of Metadata*. Version Number: v0.0.1. DOI: 10.5281/zenodo.2538983. URL: https://doi.org/10.5281/zenodo.2538983.

Hammarström, Harald, Robert Forkel, Martin Haspelmath, and Sebastian Bank (2021). *Glottolog database 4.5*. Version Number: v4.5, Type: dataset. Leipzig, Germany: Max Planck Institute for Evolutionary Anthropology. DOI: 10.5281/ZENODO.5772642. URL: https://zenodo.org/record/5772642.

Haslhofer, Bernhard and Wolfgang Klas (2010). "A survey of techniques for achieving metadata interoperability". In: *ACM Computing Surveys* 42.2, pp. 1–37. DOI: 10.1145/1667062.1667064. URL: https://dl.acm.org/doi/10.1145/1667062.1667064.

Hinrichs, Erhard and Steven Krauwer (2014). "The CLARIN Research Infrastructure: Resources and Tools for eHumanities Scholars". In: *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC 2014)*. Reykjavik, Iceland: ELRA, pp. 1525–1531. URL: http://www.lrec-conf.org/proceedings/lrec2014/pdf/415_Paper.pdf.

Hugo, Wim, Yann Le Franc, Gerard Coen, Jessica Parland-von Essen, and Luiz Bonino (2020). "D2.5 FAIR Semantics Recommendations Second Iteration". In: DOI: 10.5281/zenodo.5362010. URL: https://zenodo.org/record/5362010.

Kornai, Andras (2013). "Digital Language Death". In: *PLoS ONE* 8.10. DOI: 10.1371/journal.pone.0077056. URL: https://doi.org/10.1371/journal.pone.0077056.

Labropoulou, Penny, Dimitris Galanis, Antonis Lempesis, Mark Greenwood, Petr Knoth, Richard Eckart de Castilho, Stavros Sachtouris, Byron Georgantopoulos, Stefania Martziou, Lucas Anastasiou, Katerina Gkirtzou, Natalia Manola, and Stelios Piperidis (2018). "OpenMinTeD: A Platform Facilitating Text Mining of Scholarly Content". In: *Proceedings of WOSP 2018 (co-located with LREC 2018)*. Miyazaki, Japan: ELRA, pp. 7–12. URL: http://lrec-conf.org/workshops/lrec2018/W24/pdf/13_W24.pdf.

Labropoulou, Penny, Katerina Gkirtzou, Maria Gavriilidou, Miltos Deligiannis, Dimitris Galanis, Stelios Piperidis, Georg Rehm, Maria Berger, Valérie Mapelli, Michael Rigault, Victoria Arranz, Khalid Choukri, Gerhard Backfried, José Manuel Gómez Pérez, and Andres Garcia-Silva (2020). "Making Metadata Fit for Next Generation Language Technology Platforms: The Metadata Schema of the European Language Grid". In: *Proceedings of the 12th Language Resources and Evaluation Conference (LREC 2020)*. Ed. by Nicoletta Calzolari, Frédéric Béchet, Philippe Blache, Christopher Cieri, Khalid Choukri, Thierry Declerck, Hitoshi Isahara, Bente Maegaard, Joseph Mariani, Asuncion Moreno, Jan Odijk, and Stelios Piperidis. Marseille, France: ELRA, pp. 3421–3430. URL: https://www.aclweb.org/anthology/2020.lrec-1.420/.

Lösch, Andrea, Valérie Mapelli, Stelios Piperidis, Andrejs Vasiļjevs, Lilli Smal, Thierry Declerck, Eileen Schnur, Khalid Choukri, and Josef van Genabith (2018). "European Language Resource Coordination: Collecting Language Resources for Public Sector Multilingual Information Management". In: *Proc. of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*. Miyazaki, Japan: ELRA. URL: https://aclanthology.org/L18-1213.

McCrae, John Philip, Philipp Cimiano, Victor Rodriguez-Doncel, Daniel Vila Suero, Jorge Gracia, Luca Matteis, Roberto Navigli, Andrejs Abele, Gabriela Vulcu, and Paul Buitelaar (2015a). "Reconciling Heterogeneous Descriptions of Language Resources". In: *Proceedings of the 4th Workshop on Linked Data in Linguistics: Resources and Applications*. Beijing, China: ACL, pp. 39–48. DOI: 10.18653/v1/W15-4205. URL: http://aclweb.org/anthology/W15-4205.

McCrae, John Philip, Penny Labropoulou, Jorge Gracia, Marta Villegas, Víctor Rodríguez-Doncel, and Philipp Cimiano (2015b). "One Ontology to Bind Them All: The META-SHARE OWL Ontology for the Interoperability of Linguistic Datasets on the Web". In: *The Semantic Web: ESWC 2015 Satellite Events*. Ed. by Fabien Gandon, Christophe Guéret, Serena Villata, John Breslin, Catherine Faron-Zucker, and Antoine Zimmermann. Lecture Notes in Computer Science. Springer International Publishing, pp. 271–282. URL: https://link.springer.com/chapter/10.1007/978-3-319-25639-9_42.

Moreno-Schneider, Julián, Rémi Calizzano, Florian Kintzel, Georg Rehm, Dimitris Galanis, and Ian Roberts (2022). "Towards Practical Semantic Interoperability in NLP Platforms". In: *Proceedings of the 18th Joint ACL-ISO Workshop on Interoperable Semantic Annotation (ISA 2022; co-located with LREC 2022)*. Ed. by Harry Bunt. Marseille, France, pp. 118–126. URL: http://www.lrec-conf.org/proceedings/lrec2022/workshops/ISA-18/pdf/2022.isa18-1.16.pdf.

Phillips, Addison and Mark Davis (2009). *Tags for Identifying Languages*. Tech. rep. RFC 5646. Internet Engineering Task Force. URL: https://datatracker.ietf.org/doc/rfc5646.

Piperidis, Stelios, Penny Labropoulou, Miltos Deligiannis, and Maria Giagkou (2018). "Managing Public Sector Data for Multilingual Applications Development". In: *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*. Ed. by Nicoletta Calzolari, Khalid Choukri, Christopher Cieri, Thierry Declerck, Sara Goggi, Koiti Hasida, Hitoshi Isahara, Bente Maegaard, Joseph Mariani, Hélène Mazo, Asuncion Moreno, Jan Odijk, Stelios Piperidis, and Takenobu Tokunaga. Miyazaki, Japan: ELRA. URL: http://www.lrec-conf.org/proceedings/lrec2018/pdf/648.pdf.

Rehm, Georg, Dimitrios Galanis, Penny Labropoulou, Stelios Piperidis, Martin Welß, Ricardo Usbeck, Joachim Köhler, Miltos Deligiannis, Katerina Gkirtzou, Johannes Fischer, Christian Chiarcos, Nils Feldhus, Julián Moreno-Schneider, Florian Kintzel, Elena Montiel, Víctor Rodríguez Doncel, John P. McCrae, David Laqua, Irina Patricia Theile, Christian Dittmar, Kalina Bontcheva, Ian Roberts, Andrejs Vasiljevs, and Andis Lagzdiņš (2020a). "Towards an Interoperable Ecosystem of AI and LT Platforms: A Roadmap for the Implementation of Different Levels of Interoperability". In: *Proc. of the 1st Int. Workshop on Language Technology Platforms (IWLTP 2020, co-located with LREC 2020)*. Ed. by Georg Rehm, Kalina Bontcheva, Khalid Choukri, Jan Hajic, Stelios Piperidis, and Andrejs Vasiljevs. Marseille, France, pp. 96–107. URL: https://www.aclweb.org/anthology/2020.iwltp-1.15.pdf.

Rehm, Georg, Katrin Marheinecke, Stefanie Hegele, Stelios Piperidis, Kalina Bontcheva, Jan Hajic, Khalid Choukri, Andrejs Vasiļjevs, Gerhard Backfried, Christoph Prinz, José Manuel Gómez Pérez, Luc Meertens, Paul Lukowicz, Josef van Genabith, Andrea Lösch, Philipp Slusallek, Morten Irgens, Patrick Gatellier, Joachim Köhler, Laure Le Bars, Dimitra Anastasiou, Albina Auksoriūtė, Núria Bel, António Branco, Gerhard Budin, Walter Daelemans, Koenraad De Smedt, Radovan Garabík, Maria Gavriilidou, Dagmar Gromann, Svetla Koeva, Simon Krek, Cvetana Krstev, Krister Lindén, Bernardo Magnini, Jan Odijk, Maciej Ogrodniczuk, Eiríkur Rögnvaldsson, Mike Rosner, Bolette Pedersen, Inguna Skadina, Marko Tadić, Dan Tufiş, Tamás Váradi, Kadri Vider, Andy Way, and François Yvon (2020b). "The European Language Technology Landscape in 2020: Language-Centric and Human-Centric AI for Cross-Cultural Communication in Multilingual Europe". In: *Proceedings of the 12th Language Resources and Evaluation Conference (LREC 2020)*. Ed. by Nicoletta Calzolari, Frédéric Béchet, Philippe Blache, Christopher Cieri, Khalid Choukri, Thierry Declerck, Hitoshi Isahara, Bente Maegaard, Joseph Mariani, Asuncion Moreno, Jan Odijk, and Stelios Piperidis. Marseille, France: ELRA, pp. 3315–3325. URL: https://www.aclweb.org/anthology/2020.lrec-1.407/.

Rehm, Georg and Hans Uszkoreit, eds. (2012). *META-NET White Paper Series: Europe's Languages in the Digital Age*. 32 volumes on 31 European languages. Heidelberg etc.: Springer.

Rehm, Georg, Hans Uszkoreit, Ido Dagan, Vartkes Goetcherian, Mehmet Ugur Dogan, Coskun Mermer, Tamás Váradi, Sabine Kirchmeier-Andersen, Gerhard Stickel, Meirion Prys Jones, Stefan Oeter, and Sigve Gramstad (2014). "An Update and Extension of the META-NET Study "Europe's Languages in the Digital Age"". In: *Proceedings of the Workshop on Collaboration and Computing for Under-Resourced Languages in the Linked Open Data Era (CCURL 2014)*. Ed. by Laurette Pretorius, Claudia Soria, and Paola Baroni. Reykjavik, Iceland, pp. 30–37. URL: http://georg-re.hm/pdf/CCURL-2014-META-NET.pdf.

Rehm, Georg and Andy Way, eds. (2023). *European Language Equality: A Strategic Agenda for Digital Language Equality*. Cognitive Technologies. Forthcoming. Springer.

Soria, Claudia, Núria Bel, Khalid Choukri, Joseph Mariani, Monica Monachini, Jan Odijk, Stelios Piperidis, Valeria Quochi, and Nicoletta Calzolari (2012). "The FLaReNet Strategic Language Resource Agenda". In: *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC 2012)*. Istanbul, Turkey: ELRA, pp. 1379–1386. URL: http://www.lrec-conf.org/proceedings/lrec2012/pdf/777_Paper.pdf.

STOA (2018). *Language equality in the digital age – Towards a Human Language Project*. STOA study (PE 598.621), IP/G/STOA/FWC/2013-001/Lot4/C2. URL: https://data.europa.eu/doi/10.2861/136527.

Straňák, Pavel, Ondřej Košarko, and Jozef Mišutka (2019). "CLARIN-DSpace repository at LINDAT/CLARIN : LINDAT/CLARIN FAIR repository for language data". In: *The grey Journal – International Journal on Grey Literature* 16, pp. 52–61.

Wilkinson, Mark D., Michel Dumontier, IJsbrand Jan Aalbersberg, Gabrielle Appleton, Myles Axton, Arie Baak, Niklas Blomberg, Jan-Willem Boiten, Luiz Bonino da Silva Santos, Philip E. Bourne, Jildau Bouwman, Anthony J. Brookes, Tim Clark, Mercè Crosas, Ingrid Dillo, Olivier Dumon, Scott Edmunds, Chris T. Evelo, Richard Finkers, Alejandra Gonzalez-Beltran, Alasdair J.G. Gray, Paul Groth, Carole Goble, Jeffrey S. Grethe, Jaap Heringa, Peter A.C 't Hoen, Rob Hooft, Tobias Kuhn, Ruben Kok, Joost Kok, Scott J. Lusher, Maryann E. Martone, Albert Mons, Abel L. Packer, Bengt Persson, Philippe Rocca-Serra, Marco Roos, Rene van Schaik, Susanna-Assunta Sansone, Erik Schultes, Thierry Sengstag, Ted Slater, George Strawn, Morris A. Swertz, Mark Thompson, Johan van der Lei, Erik van Mulligen, Jan Velterop, Andra Waagmeester, Peter Wittenburg, Katherine Wolstencroft, Jun Zhao, and Barend Mons (2016). "The FAIR Guiding Principles for Scientific Data Management and Stewardship". In: *Scientific Data* 3. DOI: 10.1038/sdata.2016.18. URL: http://www.nature.com/articles/sdata201618.

Wittenburg, Peter, Nuria Bel, Lars Borin, Gerhard Budin, Nicoletta Calzolari, Eva Hajicova, Kimmo Koskenniemi, Lothar Lemnitzer, Bente Maegaard, Maciej Piasecki, Jean-Marie Pierrel, Stelios Piperidis, Inguna Skadina, Dan Tufis, Remco van Veenendaal, Tamas Váradi, and Martin Wynne (2010). "Resource and Service Centres as the Backbone for a Sustainable Service Infrastructure". In: *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC 2010)*. Valletta, Malta: ELRA. URL: http://www.lrec-conf.org/proceedings/lrec2010/pdf/679_Paper.pdf.

Wolf, Thomas, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush (2020). "Transformers: State-of-the-art Natural Language Processing". In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. ACL, pp. 38–45. DOI: 10.18653/v1/2020.emnlp-demos.6. URL: https://aclanthology.org/2020.emnlp-demos.6.

Zeng, Marcia Lei and Lois Mai Chan (2006). "Metadata Interoperability and Standardization - A Study of Methodology Part II: Achieving Interoperability at the Record and Repository Levels". In: *D-Lib Magazine* 12.6. DOI: 10.1045/june2006-zeng. URL: http://www.dlib.org/dlib/june06/zeng/06zeng.html.

Part II
ELG Inventory of Technologies and Resources

Chapter 7
Language Technology Tools and Services

Ian Roberts, Andres Garcia Silva, Cristian Berrìo Aroca, Jose Manuel Gómez-Pérez, Miroslav Jánošík, Dimitris Galanis, Rémi Calizzano, Andis Lagzdiņš, Milan Straka, and Ulrich Germann

Abstract At the time of writing, the European Language Grid includes more than 800 LT services of varied types, including machine translation (MT), automatic speech recognition (ASR), text-to-speech synthesis (TTS), and text analysis ranging from simple tokenisers and part-of-speech taggers through to complete named entity recognition and sentiment analysis systems. This chapter gives a high-level summary of the development of the ELG service catalogue over time and digs deeper to discuss the process of service integration by looking at a few example services.

1 Introduction

The European Language Grid platform is able to support a wide variety of different types of Language Technology tools and services (see Chapter 3 for a more detailed description). Service types are classified based on the type of data they process as

Ian Roberts
University of Sheffield, UK, i.roberts@sheffield.ac.uk

Andres Garcia Silva · Cristian Berrìo Aroca · Jose Manuel Gómez-Pérez
Expert AI, Spain, agarcia@expert.ai, cberrio@expert.ai, jmgomez@expert.ai

Miroslav Jánošík
HENSOLDT Analytics GmbH, Austria, miroslav.janosik@hensoldt-analytics.com

Dimitris Galanis
Institute for Language and Speech Processing, R. C. "Athena", Greece, galanisd@athenarc.gr

Rémi Calizzano
Deutsches Forschungszentrum für Künstliche Intelligenz GmbH, Germany, remi.calizzano@dfki.de

Andis Lagzdiņš
Tilde, Latvia, andis.lagzdins@tilde.lv

Milan Straka
Charles University, Czech Republic, straka@ufal.mff.cuni.cz

Ulrich Germann
University of Edinburgh, UK, ulrich.germann@ed.ac.uk

G. Rehm (ed.), *European Language Grid*, Cognitive Technologies,
https://doi.org/10.1007/978-3-031-17258-8_7

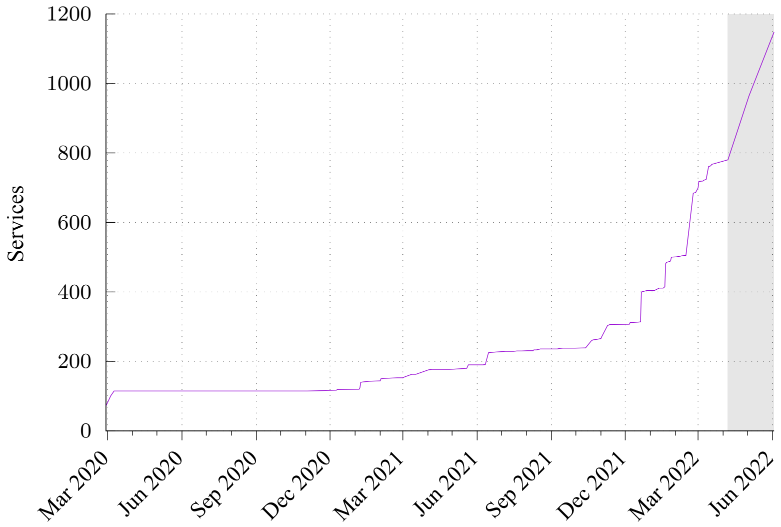


Fig. 1 Number of tools and services integrated into the European Language Grid over time; the grey shaded area denotes services whose integration is in progress at the time of writing and will be complete by the time of publication

input – text, audio, image data, etc. – and what they produce as *output* – annotations, text, audio, etc. This covers all the well-known service types such as Machine Translation (MT – text in, text out), Automatic Speech Recognition (ASR – audio in, text out), and Information Extraction/Text Analysis (IE – text in, annotations out), but also allows for services such as entity detection in *audio* data (audio in, annotations out), text-to-speech synthesis (TTS – text in, audio out), or optical character recognition (OCR – images in, text out).

Over the course of the original ELG EU project (Figure 1) the platform has grown from around 100 services available in the initial alpha release in 2020 to over 500 at the start of 2022 and almost 800 at the time of writing, with more being added all the time. The early stages of the project concentrated on services supplied by the ELG project consortium partners – such as ASR from HENSOLDT Analytics, MT from the University of Edinburgh and Tilde, TTS from Tilde, and a wide variety of Text Analysis services from Expert.AI, the University of Sheffield and DFKI (Roberts et al. 2020). More recently, an increasing number of services have been supplied by the ELG-funded pilot projects (see Part IV) and the platform has also begun to see contributions from third parties with no direct connection to the ELG consortium itself (Roberts et al. 2021, 2022). Of particular note is a set of over 500 MT services covering all pairs of EU official languages from the Neural Translation for the EU project, discussed in more detail in Section 2.[1] One third of these services have been integrated to date, with the remaining two thirds scheduled for integration during April and May 2022 (the grey shaded region in the graph), bringing the total number

[1] https://nteu.eu

			English	German	Italian	Spanish	French	Dutch	Swedish	Finnish	Polish	Czech	Greek	Portuguese	Danish	Bulgarian	Romanian	Estonian	Latvian	Slovenian	Croatian	Lithuanian	Slovak	Hungarian	Maltese	Irish	**Total A**	Total B (24 langs.)	Others (69 langs.)	**Total**
Text analysis	Linguistic pre-processing	Part-of-Speech Tagging	8	3	3	3	3	4	3	3	2	2	2	3	3	2	2	2	2	2	2	1	2	1	1	1	**60**	15	34	**109**
		Morphology	5	2	2	2	2	2	3	3	1	1	1	1	3	1	1	5	1	1	1	1	1	1	1	1	**43**	13	27	**83**
		Lemmatization	3	2	2	2	2	2	2	2	1	1	1	2	2	1	1	2	1	1	1	1	1	1	1	1	**36**	11	32	**79**
		Tokenization	6	4	3	2	2	3	1	1	1	1	1	1	1	1	1	2	1	1	1	1	1	1	1	1	**39**	10	27	**76**
		Sentence splitting	1	1				1																			**3**			**3**
		Chunking	1										1														**2**			**2**
	Total pre-processing		**24**	**12**	**10**	**9**	**9**	**12**	**9**	**9**	**5**	**5**	**6**	**7**	**9**	**5**	**5**	**11**	**5**	**5**	**5**	**4**	**5**	**4**	**4**	**4**	**183**	**49**	**120**	**352**
	+ Classification		16	6	16	5	4	3	3	2	3	2	4	3	2	2	2	1	1	1	2	1	2	2			**83**	8	40	**131**
	+ Entity annotation		17	7	4	4	6	5	5	2	1	2	3	2	1	1	2				1		1	1			**65**	14	16	**95**
	+ Linking & disambiguation		7	2	3	4	4	1						1													**22**	2	5	**29**
	+ Sentiment/Opinion mining		13	3	2	2	2	1	1		1		1	2						1	1						**30**		10	**40**
	+ Text transformation		5	3	1	2	2	1						1													**15**	1	5	**21**
	+ Parsing		1	2	1	1	1	1	1	1	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	1	**26**	10	27	**63**
	+ Other text analysis		14	8	2	3	1	2	6	6			3	1	2												**48**	4	5	**57**
Total Text Analysis			**97**	**43**	**39**	**30**	**29**	**26**	**25**	**20**	**11**	**10**	**19**	**18**	**15**	**9**	**10**	**13**	**7**	**8**	**10**	**6**	**9**	**8**	**5**	**5**	**472**	**88**	**228**	**788**
+ Machine Translation into …			90	42	27	31	29	35	32	33	36	36	24	27	27	32	29	26	26	28	25	26	25	26	24	24	**760**	88	83	**931**
+ Speech recognition & analysis			2	2	2	3	2	2	1	2	2	1	2				2	1	2			1					**27**	9	21	**57**
+ Other services			11	7	5	4	8	4	2	1	4	2	3	2	4	4	2	3	4	2	2	4	2	2	2	1	**85**	10	77	**172**
Grand Total			**200**	**94**	**73**	**68**	**68**	**67**	**60**	**56**	**53**	**49**	**48**	**47**	**46**	**45**	**43**	**43**	**39**	**38**	**37**	**37**	**36**	**36**	**31**	**30**	**1344**	**195**	**409**	**1948**

Table 1 A snapshot of all services in the ELG platform, grouped by function and supported language. This includes all services integrated as at the end of March 2022, plus 368 additional MT services whose integration is ongoing. EU official languages (type A) are listed individually; type B represents other languages used in the EU, accession candidate countries, or EEA/EFTA members; “others” refers to languages from the rest of the world. For Machine Translation, the columns in this table represent the *target* language, see Table 2 for a breakdown by *source*.

of integrated service entries in ELG up to at least 1,148 by June 2022. We hope this trend will accelerate now that the third platform release is complete.

Furthermore, the figure of 1,148 hides the fact that a number of services combine several different functions (such as tokenisation, sentence splitting, part-of-speech tagging, entity detection, linking and disambiguation) into a single process and/or offer the same function in more than one language. Counting each language/function pair individually gives a more informative picture of the scope and coverage of ELG. For example, the platform currently provides one service that does dependency parsing for Portuguese; it also provides one service that does lemmatisation for Portuguese. The user who is looking for these two functions does not care whether they are implemented by one service or by two, only whether or not the European Language Grid can meet their needs.

By this measure, as of the end of March 2022, ELG offers 1,576 distinct service function/language combinations – already exceeding the 1,300 predicted by the project in mid-2021 (Rehm et al. 2021) – and is on track to offer at least 1,948 by June, which are summarised in Table 1. Reading from the bottom up, the 1,948 total breaks down into 931 MT (47.7% of the total), 788 text analysis (40%), 57 speech recognition and audio analysis, and 172 services of other types such as text to speech and OCR. The middle section of Table 1 breaks the 788 text analysis services down into broad sub-categories, and the top section breaks the largest sub-category (linguistic pre-processing) down into individual functions.

The largest *single* category of services is MT, with 770 catalogue entries representing 931 actual translation services (since some of the models are multilingual, with the same endpoint accepting input in several different languages and translating them all to the same target). The available text analysis services range from low-level text processing tasks such as tokenisation, part-of-speech tagging or morphological analysis, through named entity annotation and on to higher-level services such as parsing, sentiment analysis and entity linking against knowledge bases. Dependency parsing in particular is supported for 60 languages courtesy of the UD-Pipe parser from Charles University in Prague. For speech, the platform currently supports speech transcription for 31 languages thanks to tools from HENSOLDT Analytics and Tilde, alongside other speech processing tools such as the keyword spotting tool described in Section 3.

Breaking the numbers down on another dimension, the ELG platform now hosts at least one service providing support for each of 114 distinct languages. English is unsurprisingly the most highly represented, but there is good support for other major EU languages – German, French, Spanish, and Italian all have support for at least 20 service functions aside from machine translation – and in total 28 languages have support for at least ten functions.

Of course there is a long tail on both axes, with 16 of the 48 distinct service functions available in only one language each and 25 in fewer than five languages. On the other hand 39 out of the 114 languages are supported by only one function, and 51 by fewer than three. Full multilinguality is still in the future, but for the languages with larger numbers of speakers at least, significant progress has been and is being made.

Target → Source ↓	English	German	Czech	Polish	Dutch	Finnish	Swedish	Bulgarian	Spanish	Romanian	French	Slovenian	Italian	Danish	Portuguese	Latvian	Estonian	Lithuanian	Hungarian	Croatian	Slovak	Greek	Irish	Maltese	Total A	Total B (20)	Total Other (7)	Total
English	▨	7	4	4	3	4	5	3	3	4	3	2	2	3	3	3	3	3	2	2	2	2	2	2	71	20	11	102
German	6	▨	1	1	2	3	1	1	1	2	1	1	1	1	1	1	1	1	2	2	1	1	1	1	34	7	3	44
Czech	5	1	▨	2	1	1	1	2	1	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	30	2	5	37
Polish	4	1	2	▨	1	1	1	2	1	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	29	2	6	37
Dutch	2	2	1	1	▨	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	25	4	2	31
Finnish	4	3	1	1	1	▨	3	1	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	1	31	1	1	33
Swedish	4	1	1	1	1	3	▨	1	1	1	1	1	1	2	1	1	1	1	1	1	1	1	1	1	29	5	1	35
Bulgarian	3	1	2	2	1	1	1	▨	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	27	2	3	32
Spanish	3	1	1	1	1	1	1	1	▨	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	25	3	1	29
Romanian	2	1	1	1	1	1	1	1	1	▨	1	1	1	1	1	1	1	1	1	1	1	1	1	1	24		1	25
French	3	1	1	1	1	2	1	1	1	1	▨	1	1	1	1	1	1	1	1	1	1	1	1	1	26	1	4	31
Slovenian	2	1	2	2	1	1	1	1	1	1	1	▨	1	1	1	1	1	1	1	1	1	1	1	1	26		2	28
Italian	2	1	1	1	1	1	1	1	1	1	1	1	▨	1	1	1	1	1	1	1	1	1	1	1	24		4	28
Danish	2	1	1	1	1	1	2	1	1	1	1	1	1	▨	1	1	1	1	1	1	1	1	1	1	25	4	1	30
Portuguese	3	1	1	1	1	1	1	1	1	1	1	1	1	1	▨	1	1	1	1	1	1	1	1	1	25		1	26
Latvian	3	1	1	1	1	1	1	1	1	1	1	1	1	1	1	▨	1	1	1	1	1	1	1	1	25		2	27
Estonian	3	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	▨	1	1	1	1	1	1	1	25		1	26
Lithuanian	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	▨	1	1	1	1	1	1	24		2	26
Hungarian	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	▨	1	1	1	1	1	24		1	25
Croatian	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	▨	1	1	1	1	24			24
Slovak	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	▨	1	1	1	24			24
Greek	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	▨	1	1	24			24
Irish	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	▨	1	24			24
Maltese	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	▨	24			24
Total A	65	32	29	29	26	31	30	27	25	27	26	26	24	26	25	25	25	25	25	25	24	24	24	24	669	51	52	772
Total B	16	4	2	3	4	1	1	2	3																36	20	12	68
Total Other	9	6	5	4	5	1	1	3	3	2	3	2	3	1	2	1	1	1	1		1				55	17	19	91
Grand Total	90	42	36	36	35	33	32	32	31	29	29	28	27	27	27	26	26	26	26	25	25	24	24	24	760	88	83	931

Table 2 A snapshot of supported MT language pairs as at the end of March 2022, with the addition of the remaining NTEU services for all pairs of EU official languages

2 Machine Translation

The ELG platform includes MT tools for 781 individual source/target language pairs, totalling 931 distinct services. Table 2 shows the breakdown; while English still dominates, it is much less ubiquitous than in the past, with only 21% of services involving English (102 from English, 90 into English, for a total of 192 out of the 931 available services). All pairs of EU official languages ("type A" in Table 2) are supported. In addition there is support for unofficial or regional European languages such as Basque, Galician and Luxembourgish and languages of accession candidates or free trade partners such as Icelandic, Norwegian[2] and Serbian[3] as well as languages important for trade and political reasons such as Modern Standard Arabic, Hindi, Ukrainian and Russian.

[2] Both Nynorsk and Bokmål varieties.

[3] Both Latin and Cyrillic script.

In addition to the MT services contributed by the ELG consortium partners Tilde (Pinnis and Bergmanis 2020) and University of Edinburgh (Junczys-Dowmunt et al. 2018; Germann et al. 2020; Germann 2020), two contributors in particular deserve a special mention here: the OPUS-MT ELG pilot project and the EU project Neural Translation for the European Union (NTEU).

The OPUS-MT ELG pilot project (Chapter 24, p. 325 ff., also see Tiedemann and Thottingal 2020) is responsible for 312 of the total 931 translation service options. To reduce the overall load on the ELG computing infrastructure, many of these language pairs are supported by multilingual models, where a single Docker container can accept input and/or produce output in many related languages. For example, there is a single OPUS model for "West Germanic", which can translate either way between any pair of English, German, Dutch, Luxembourgish, Afrikaans, Low Saxon, Gronings and Hunsrik. Some language pairs are supported by multiple models with different performance characteristics, for example, English to German is supported by a monolingual English-German model, a one-to-many "English to West Germanic", and the aforementioned many-to-many West Germanic model. Which model is most appropriate for a given task will vary, for example, if the input is known to be good-quality English then the monolingual model may be best, but if the input is a mix of languages, or English written by native speakers of other Germanic languages, then the multilingual model may be more accurate. Enabling users to test out different services on their own real data and switch between them with no technical changes to their code is one of the greatest benefits of the ELG approach.

NTEU is a project with a different focus, it was funded to produce high-quality translation tools for *all* possible pairs of EU official languages, to reduce the need for relay translation through a better-resourced language such as English (Bié et al. 2020; García-Martínez et al. 2021). This gives a total of 552 translation models (24 source languages each translating into the other 23 targets), so to spread the load of developing the models, NTEU involved three partner organisations, each responsible for models translating into eight target languages (one third of the total EU24). At the time of writing, one of the three sets of models has been published as ELG-integrated services and the other two sets are expected to be available by the time this book is published. The inclusion of these services marks an important milestone for ELG for two key reasons. First it shows the strong commitment of ELG to full multilinguality in the European Digital Single Market, and second it is the single largest contribution to the ELG platform originating outside the original ELG project consortium and pilot project ecosystem, demonstrating that ELG truly is a platform for the whole EU language technology community.

3 Automatic Speech Recognition

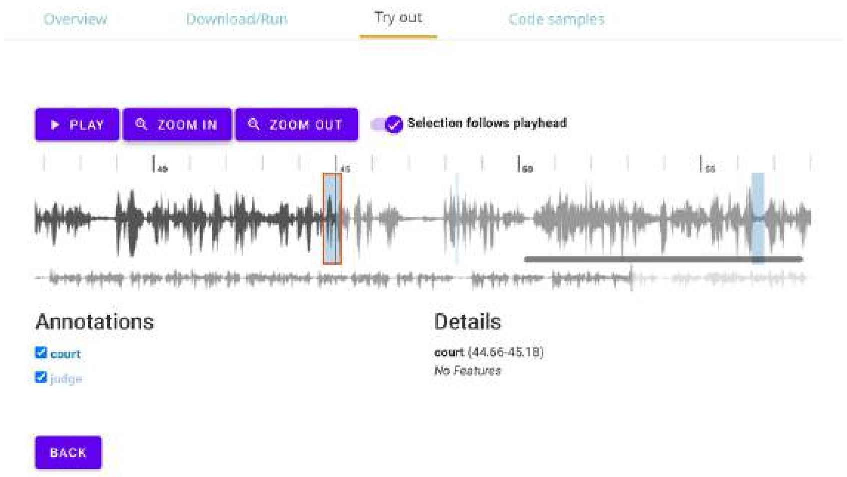
For automatic speech recognition, ELG currently hosts 48 services covering 30 languages and dialects. The majority of these have been provided by HENSOLDT Analytics, the speech recognition specialist in the ELG project consortium. In addition, there have also been important contributions from Tilde for the Baltic languages, and from two of the pilot project organisations: Elhuyar for Basque (see Chapter 15, p. 271 ff.) and Lingsoft for Scandinavian languages (see Chapter 20, p. 301 ff.). Lingsoft have also begun to deliver *domain-specific* ASR services, for example a service tuned to recognise clinical speech in Finnish. As general purpose ASR systems increasingly become commodities, the creation and provision of domain-specific models provides an important niche for smaller ASR providers.

These organisations are all commercial service providers; though the tools themselves are based on open source frameworks such as Kaldi[4], the models are the proprietary intellectual property of the respective provider.

3.1 Case Study: Speech Tools from HENSOLDT

In addition to the actual ASR, the components provided by HENSOLDT also perform several preprocessing steps: audio is downsampled and converted to the native format of the respective models (typically 16kHz, 16 bit, mono, signed). Segmentation and classification of the input audio is carried out next. Any segment classified as containing an insufficient amount of speech is discarded and not processed by the ASR. Disfluencies and non-speech within segments identified as audio-segments are processed by the ASR system via specific non-speech models. Segmentation as well as classification are parameterised and can be adapted to specific audio conditions (the components provided within ELG use standard settings). Processing within the HENSOLDT ASR is staged in a pipelined manner for optimal throughput. Processing parameters can be employed to balance processing speed and accuracy. Like Lingsoft, HENSOLDT also provides *domain-specific* models which can be included in the respective Docker components. The ASR engine itself is *aware* of processing throughput as well as of the various models used. It can be adjusted to provide realtime processing as well as to reload different sub-models as soon as they become available. While the current services use one standard model, this allows for future updates of vocabularies and language models in a transparent manner. Output of the HENSOLDT ASR component can be provided in 1-best, n-best or lattice formats. The former is currently used in the deployed components, however, lattice-based output is used indirectly for use of the ASR component for keyword-spotting (KWS) applications only. A sample result of the detection of keywords via ASR can be seen in Figure 2.

[4] http://kaldi-asr.org



Annotations

court

judge

Details

court (44.66-45.18)

No Features

Fig. 2 Example of the word “court” having been detected as a keyword using HENSOLDT ASR

4 Text Analytics

After the set of MT services, the second largest group of services in the ELG platform are concerned in one way or another with the analysis and annotation of text, as discussed in Section 1. These cover a wide range from low-level text pre-processing tasks such as tokenisation and sentence splitting, through named entity annotation and linking tools (in many languages and domains), to dependency parsing, summarisation, sentiment analysis, and special purpose services such as the detection of misinformation or hate speech, and spelling and grammar checking.

Text analysis services have been provided by most members of the ELG project consortium, Expert.AI contributing their Cogito Discover toolkit, the University of Sheffield providing many services based on their GATE framework, Charles University providing their UDPipe dependency parser and other tools (e. g., Straka and Straková 2020; Straka et al. 2019b; Straka 2018; Straková et al. 2019; Straka et al. 2019a) and HENSOLDT (Dikici et al. 2019), ILSP (e. g., Prokopis and Piperidis 2020; Pontiki et al. 2018; Papanikolaou et al. 2016; Pontiki and Papageorgiou 2015) and DFKI (e. g., Schulz et al. 2022; Aksenov et al. 2021; Leitner et al. 2019) providing a variety of tools from their respective inventories. In addition, several of the pilot projects have contributed services in this class, notably

- *European Clinical Case Corpus* (Chapter 17, p. 283 ff.) – Fondazione Bruno Kessler. Clinical named entity recognisers in six languages.
- *Italian EVALITA Benchmark Linguistic Resources, NLP Services and Tools* (Chapter 19, p. 295 ff.) – University of Turin. A variety of services based on systems that participated in the various EVALITA shared tasks throughout the

years such as misogyny and hate speech detection and gender prediction, all in the Italian language.

- *Lingsoft Solutions as Distributable Containers* (Chapter 20, p. 301 ff.) – Lingsoft. General text analysis, proofing tools (spelling and grammar checking) and morphology analysis, in English and Scandinavian languages. This includes regional variations, such as distinct services for Swedish as used in Sweden and Swedish as used in Finland, and domain variations with specific services for medical domain text.
- *Universal Semantic Annotator* (Chapter 28, p. 349 ff.) – Sapienza University of Rome. This service performs word sense disambiguation, semantic role labelling and parsing for a wide variety of different languages.

4.1 Case Study: Cogito Discover from Expert.AI

Cogito Discover is Expert.AI's scalable software platform for automatic semantic metadata generation and auto-classification that can be easily integrated in the production environment of document-processing applications or workflows. It can be deployed on premise and in cloud environments and is available for both Linux and Windows systems. Cogito Discover services that are included in ELG are:

- Language detection: Identify the main language used in a text.
- Part-of-speech annotation: Annotations at different levels (token, word/compound word, group, clause, sentence) with grammatical types.
- Named Entity Recognition: Annotation of entities, i. e., people, organisations, places, known concepts, unknown concepts and also tags, i. e., URLs, email addresses, phone numbers, addresses, dates, time, measures, money, percentage, file folder.
- Semantic annotation: This service returns the concepts spotted in a text which are modelled in the Cogito Discover knowledge graph.
- Lemmatisation: This service returns the lemma of each concept spotted in the text that is modelled in the Cogito Discover knowledge graph.
- Keyword extraction: Annotation of the most relevant information, i. e., main syncons, main lemmas, main multiword expressions.
- Sentiment analysis: Provides a sentiment score (positive or negative) for the entities recognised in the text, and an overall score for the whole set of entities in the document.
- Summarisation: Annotation of the most relevant information, i. e., main syncons, main lemmas, main multiword expressions, main sentences and main domains.
- Categorisation: Classify documents using the IPTC taxonomy.

Most services are available in 12 languages: English, Italian, Spanish, German, French, Dutch, Portuguese, Chinese, Arabic, Russian, Japanese and Korean.

For its deployment in ELG, Expert.AI generated a Docker image containing a Cogito Discover installation, the linguistic packages, and a general adapter that manages the communication between the ELG platform and Cogito Discover. The general adapter was developed using the ELG Spring Boot Starter described in Chapter 4 (Part I, p. 67 ff.)[5], which makes it as easy as possible to create ELG-compliant tools in Java using Spring Boot.

4.2 Case Study: GATE from University of Sheffield

The University of Sheffield has been developing and maintaining the GATE framework for Natural Language Processing[6] for over 20 years. The basic framework is open source software written in Java and comes with a wide variety of plugins, some implementing specific NLP algorithms and some providing the generic base on which other specific rule-based and machine learning-based tools can be built.

The GATE ecosystem includes its own software-as-a-service platform called GATE Cloud (Tablan et al. 2013). An early focus of Sheffield's work in the ELG project was to develop a bridge to GATE Cloud, i. e., a proxy that accepts ELG API requests and dispatches them to a service endpoint on GATE Cloud, translating the resulting annotations into the ELG API response format. The development of this bridge has enabled the rapid deployment of many GATE Cloud hosted services into the ELG catalogue with little demand on the computing capacity of the ELG platform itself. At the time of writing, there are 66 GATE-based services integrated in ELG via the bridging proxy.

However, GATE Cloud itself has rate limits, so alongside the bridge component, Sheffield has developed a generic tool that can take any NLP application built against the GATE framework and bundle the application and all the plugins on which it depends as a Docker image that can run the application in-process within the ELG infrastructure. This mechanism has been used to wrap up certain particularly significant GATE-based applications so they can run directly in the ELG Kubernetes cluster and take advantage of the ELG platform's auto-scaling capabilities (see Chapter 5).

As the ELG EU project draws to a close, things have started to come full circle, as a number of recent additions *to* GATE Cloud have in fact been implemented as ELG-compatible Docker images, with a bridge in the other direction to enable a GATE application to call out to an endpoint that exposes the ELG internal LT service API. Some of these ELG-compatible images have been contributed back to ELG.

In addition, Sheffield has promoted the use of ELG-compatible services and Docker images in a number of other projects, notably the Horizon 2020 projects WeVerify[7] and RISIS2[8]. Many of Sheffield's contributions to these projects have

[5] https://gitlab.com/european-language-grid/platform/elg-spring-boot-starter

[6] General Architecture for Text Engineering, https://gate.ac.uk, see Cunningham et al. (2013).

[7] Wider and Enhanced Verification For You, https://weverify.eu, see Marinova et al. (2020).

[8] Research Infrastructure for Science and Innovation Policy Studies, https://www.risis2.eu, see Reale et al. (2019).

been implemented as ELG-compatible Docker images, with bridging components written for those projects to act as clients of the ELG API. The same mechanism has been used as part of a long-term collaboration between the University of Sheffield and King's College London, to integrate medical domain LT services developed in Python at King's into an existing GATE-based processing workflow. The use of the ELG standardised API makes it easy to integrate a variety of services implemented in different programming languages in a minimally-invasive way.

4.3 Case Study: Microservices At Your Service

With the third release in 2022, the ELG platform has begun to see contributions from third parties beyond the initial ELG consortium and pilot projects. One notable source is the project Microservices At Your Service[9], funded by the European Commission's Connecting Europe Facility (CEF) programme and led by Lingsoft (one of the organisations funded for a pilot project in the first ELG open call, see Chapter 20, p. 301 ff.). The project describes its mission as "bridging the gap between NLP research and industry" and it aims to identify open source text analysis tools that could benefit the community, package them as Docker images, and publish them for wider use. The project has selected the ELG platform as its primary vehicle for publication of the tools, and uses the ELG API as its standard specification for interoperability.

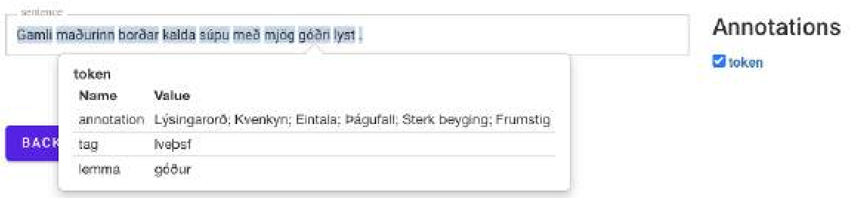
The project concentrates primarily on Finnish, Estonian, Icelandic, Spanish and Portuguese, plus some tools for minority languages from the same regions such as Faroese, Galician and Catalan. So far more than 14 services have been published, including:

- A proxy to the Finto-AI subject indexing service[10], in Finnish, Swedish and English (Suominen et al. 2022)
- Named entity recognition tools for Swedish and Norwegian, originally from the respective national libraries of the two countries (Kummervold et al. 2021)
- A tokeniser and morphological analysis tool for Estonian (Kaalep and Vaino 2001)
- A variety of tools for Icelandic from the University of Reykjavík, including a tokeniser, part-of-speech tagger, shallow parser and named entity recogniser, as well as machine translation models between Icelandic and English

One of the Icelandic services, a part-of-speech tagger and lemmatizer, is shown in Figure 3.

[9] https://www.lingsoft.fi/en/microservices-at-your-service-bridging-gap-between-nlp-research-and-industry

[10] https://ai.finto.fi



token

Name	Value
annotation	Lýsingarorð; Kvenkyn; Eintala; Þágufall; Sterk beyging; Frumstig
tag	lveþsf
lemma	góður

Fig. 3 Icelandic lemmatizer and part-of-speech tagger from Microservices At Your Service

5 Other Service Types

Right from the start of the ELG project, it was clear that the three principal service classes (ASR, MT, Text Analytics), while significant, would never be exhaustive. An important goal of ELG was to remain flexible enough to be able to easily integrate new classes of services and tools that had not been foreseen in the original proposal. The API specifications were designed with this flexibility in mind, being based solely on the kinds of data each service expects and returns, rather than placing any requirements on what the service *does* with that data.

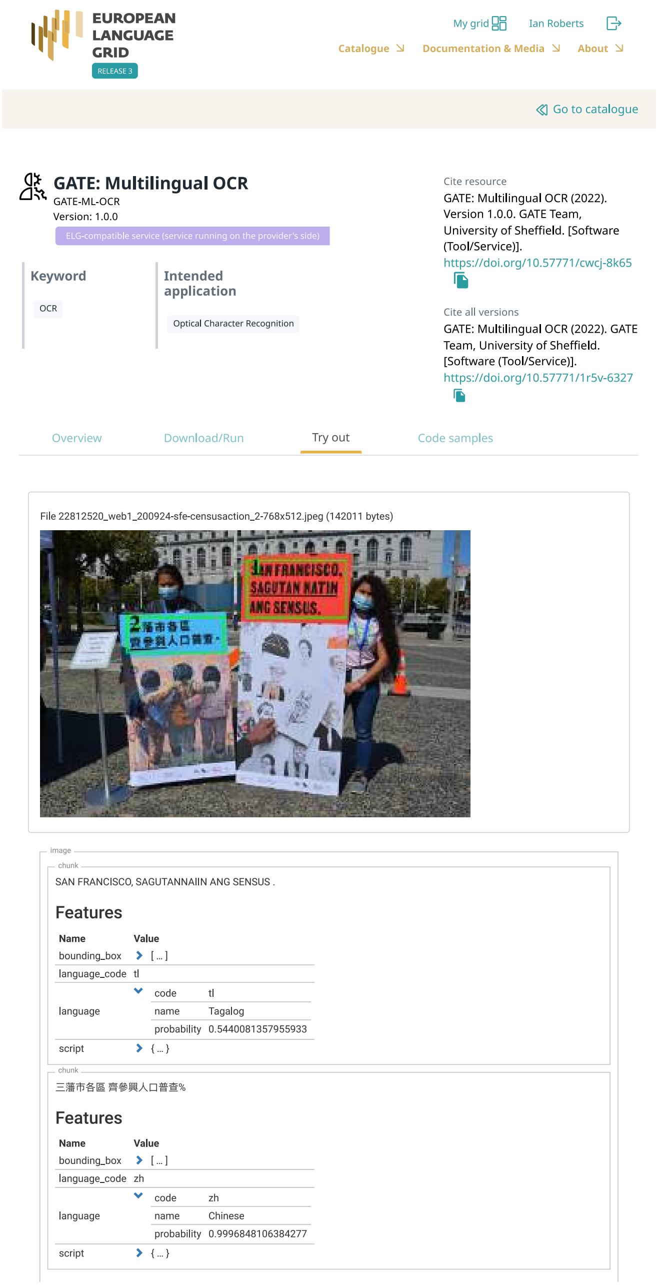
Three classes of "other" services have emerged since the beginning of the project:

- *Text-to-speech* services that take text and synthesise audio.
- *Audio analysis* services that take audio input and return standoff annotations over time segments of the audio stream.
- *Image analysis* services, in particular optical character recognition (OCR).

Text-to-speech services have been provided by Tilde within the ELG project consortium (for Latvian and Lithuanian), and by the Elhuyar pilot project (for Basque). The audio analysis services are the keyword spotting tools from HENSOLDT Analytics described along with their speech recognition systems in Section 3.

The University of Sheffield has contributed a multilingual image OCR service developed as part of the Horizon 2020 EU project WeVerify. The service is based on a multi-step pipeline of neural models, first running a segmentation model to identify regions within the image that contain text, then a classifier to identify the writing system and language of each text block, and finally an appropriate text recognition model on each block depending on the identified script (Arabic, Bengali-Assamese, Chinese, Latin, Devanagari, Kanna, Hangul or Cyrillic). An example can be seen in Figure 4. The models have been deliberately designed *not* to use the "attention" mechanism typical of other deep neural models, as this was found to give only marginal improvements in performance at the cost of significantly increased memory and compute requirements.

Part of the reason for ELG funding the open call for pilot projects was precisely to elicit suggestions of new classes of services that were not previously known to the project consortium. Two pilots in particular delivered on this: Text2TCS (Section 5.1) and Coreon's MKS as LLOD (Section 5.2).



GATE: Multilingual OCR
GATE-ML-OCR
Version: 1.0.0
ELG-compatible service (service running on the provider's side)

Keyword
OCR

Intended application
Optical Character Recognition

Cite resource
GATE: Multilingual OCR (2022). Version 1.0.0. GATE Team, University of Sheffield. [Software (Tool/Service)]. https://doi.org/10.57771/cwcj-8k65

Cite all versions
GATE: Multilingual OCR (2022). GATE Team, University of Sheffield. [Software (Tool/Service)]. https://doi.org/10.57771/1r5v-6327

File 22812520_web1_200924-sfe-censusaction_2-768x512.jpeg (142011 bytes)

image
chunk
SAN FRANCISCO, SAGUTANNAIIN ANG SENSUS .

Features

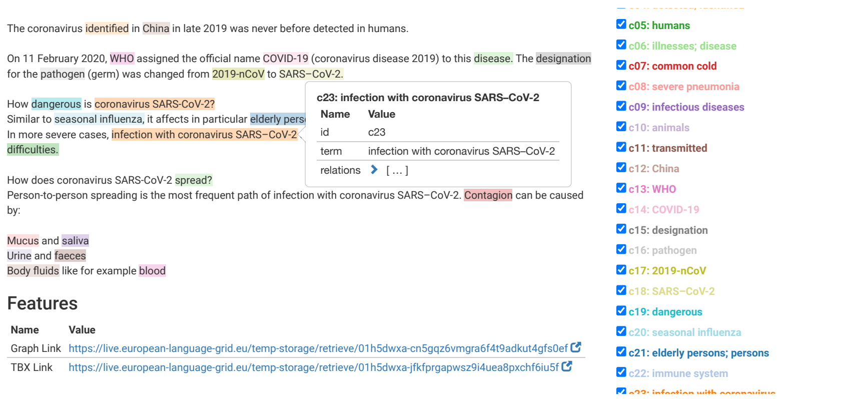
Name	Value	
bounding_box	[...]	
language_code	tl	
language	code	tl
	name	Tagalog
	probability	0.5440081357955933
script	{ ... }	

chunk
三藩市各區 齊參興人口普查%

Features

Name	Value	
bounding_box	[...]	
language_code	zh	
language	code	zh
	name	Chinese
	probability	0.9996848106384277
script	{ ... }	

Fig. 4 The Multilingual OCR service showing detection of two blocks of text in different scripts (the bounding boxes *are* part of the "try out" UI, they have not been added to this figure)



The coronavirus identified in China in late 2019 was never before detected in humans.

On 11 February 2020, WHO assigned the official name COVID-19 (coronavirus disease 2019) to this disease. The designation for the pathogen (germ) was changed from 2019-nCoV to SARS–CoV-2.

How dangerous is coronavirus SARS-CoV-2?
Similar to seasonal influenza, it affects in particular elderly pers
In more severe cases, infection with coronavirus SARS–CoV-2
difficulties.

How does coronavirus SARS-CoV-2 spread?
Person-to-person spreading is the most frequent path of infection with coronavirus SARS–CoV-2. Contagion can be caused by:

Mucus and saliva
Urine and faeces
Body fluids like for example blood

c23: infection with coronavirus SARS–CoV-2

Name	Value
id	c23
term	infection with coronavirus SARS–CoV-2
relations	› […]

Features

Name	Value
Graph Link	https://live.european-language-grid.eu/temp-storage/retrieve/01h5dwxa-cn5gqz6vmgra6f4t9adkut4gfs0ef
TBX Link	https://live.european-language-grid.eu/temp-storage/retrieve/01h5dwxa-jfkfprgapwsz9i4uea8pxchf6iu5f

- ✓ c05: humans
- ✓ c06: illnesses; disease
- ✓ c07: common cold
- ✓ c08: severe pneumonia
- ✓ c09: infectious diseases
- ✓ c10: animals
- ✓ c11: transmitted
- ✓ c12: China
- ✓ c13: WHO
- ✓ c14: COVID-19
- ✓ c15: designation
- ✓ c16: pathogen
- ✓ c17: 2019-nCoV
- ✓ c18: SARS–CoV-2
- ✓ c19: dangerous
- ✓ c20: seasonal influenza
- ✓ c21: elderly persons; persons
- ✓ c22: immune system

Fig. 5 Text2TCS service results in the "try out" GUI, showing links to the termbase and graph

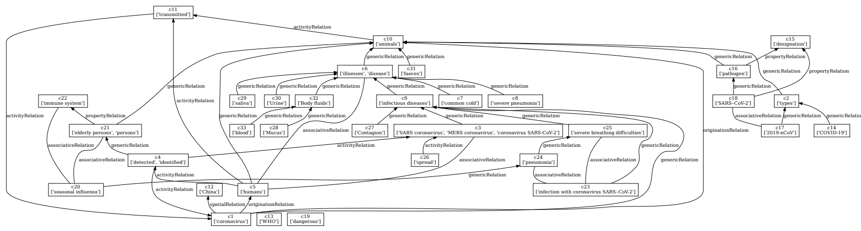


Fig. 6 The termbase graph generated from the sample input text (Figure 5)

5.1 Pilot Project: Terminological Concept Systems from Natural Language Text from University of Vienna

The Text2TCS project (see Chapter 18, Part IV, p. 289 ff.) aimed to develop a tool for deriving terminological concept systems from natural language text. This required the generation not only of typical standoff annotations representing the mentions of the detected terms in the source text, but also two additional output files for the termbase in TBX format[11] and a visualisation of the terminology as a PNG image.

These additional outputs did not naturally fit the JSON-based data interchange formats of the ELG API. It would have been possible to force them into this format by, for example, encoding the PNG data in base 64 encoding, but instead the ELG team took this as the impetus to introduce the "temporary storage" helper service for use by LT service containers. The operation of the temporary storage service is very simple. LT services can send arbitrary binary data to a well-known URL `http://storage.elg/store` (a private host name that resolves only within the ELG Kubernetes cluster), and will receive in return a publicly-resolvable URL which can be returned to the caller of the LT service for them to use to retrieve the same

[11] https://www.tbxinfo.net

data. Storage URLs include a cryptographically-secure random token to make them un-guessable, and they expire by default 15 minutes from their generation, at which time the stored data is permanently deleted.

Figures 5 and 6 show how this appears in the ELG portal when a user tests the Text2TCS service using the "try out" mechanism.

The temporary storage service provides an elegant solution to the problem of allowing LT services to return binary data without introducing additional complexity for the majority of services that do not have this requirement.

5.2 Pilot Project: MKS as Linguistic Linked Open Data from Coreon

The pilot project MKS as LLOD by knowledge management company Coreon (see Chapter 23, Part IV, 319 ff.) is an interesting case that in some ways sits at the boundary between services and resources. The aim of the project was to take Coreon's existing knowledge representation systems, known as MKS for Multilingual Knowledge System, and expose them as Linguistic Linked Open Data (LLOD). There is already a (de jure *and* de facto) standard API for querying linked (open) data resources, i. e., the SPARQL query language[12], so rather than defining a new format under the ELG umbrella, we decided to adopt the existing standard.

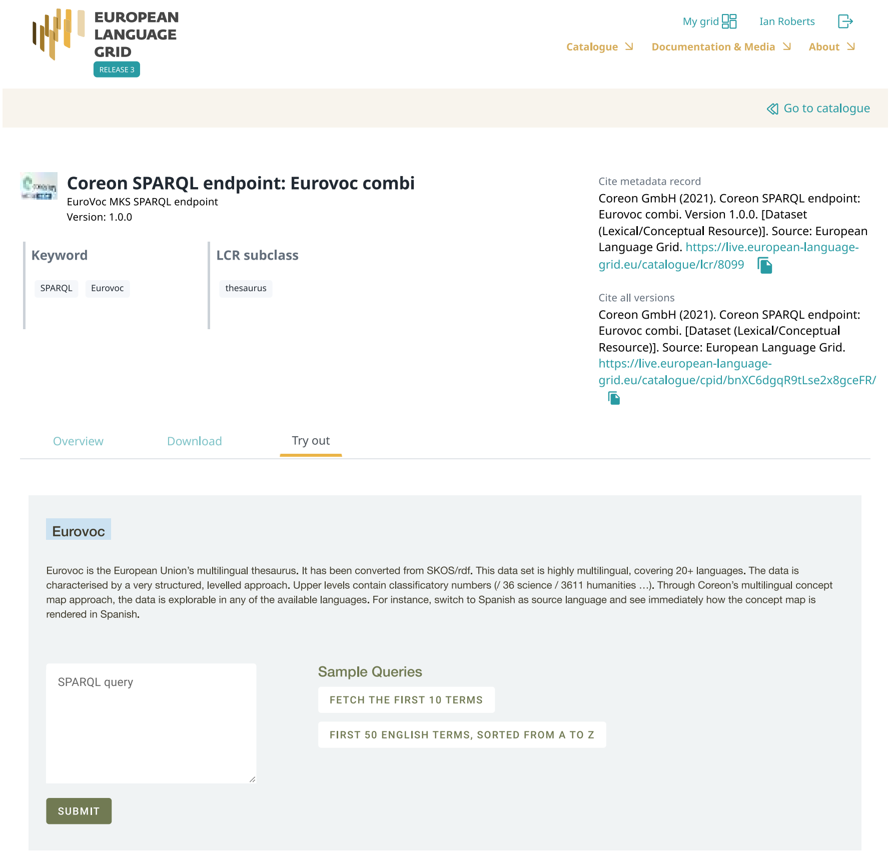
For ELG, the question was how best to represent this kind of resource in the ELG metadata scheme. On the one hand, the object that was being provided by Coreon was conceptually a data resource, albeit one accessed via a query API rather than via direct download, but on the other hand the technical method of integration would be through providing a SPARQL *service* for users to query. The eventual solution was in fact a mixture of both.

The Coreon SPARQL endpoint was integrated into the ELG infrastructure and set up so that SPARQL queries could be authenticated using access tokens issued by the ELG Keycloak identity provider, exactly as for other ELG LT services. In parallel, Coreon developed a "try out" UI to allow users to make test queries through the ELG catalogue interface. The two were then tied together as follows:

1. The "try out" UI was registered in its own right as a "service" in the ELG catalogue, whose function is "resource access".
2. Each SPARQL endpoint was then registered as an individual "ELG-compatible Lexical or Conceptual Resource" (LCR), with a link to the "try out" UI as "this resource is queried by that service".

Logic was introduced in the ELG catalogue to recognise when a user visits an ELG-compatible LCR that has an associated query service, and to inject the query UI as a "try out" tab which is configured with the necessary information and access token to be able to query the SPARQL endpoint (see Figure 7 for the final result).

[12] https://www.w3.org/TR/sparql11-overview/



Coreon SPARQL endpoint: Eurovoc combi
EuroVoc MKS SPARQL endpoint
Version: 1.0.0

Keyword
SPARQL Eurovoc

LCR subclass
thesaurus

Cite metadata record
Coreon GmbH (2021). Coreon SPARQL endpoint: Eurovoc combi. Version 1.0.0. [Dataset (Lexical/Conceptual Resource)]. Source: European Language Grid. https://live.european-language-grid.eu/catalogue/lcr/8099

Cite all versions
Coreon GmbH (2021). Coreon SPARQL endpoint: Eurovoc combi. [Dataset (Lexical/Conceptual Resource)]. Source: European Language Grid. https://live.european-language-grid.eu/catalogue/cpid/bnXC6dgqR9tLse2x8gceFR/

Eurovoc

Eurovoc is the European Union's multilingual thesaurus. It has been converted from SKOS/rdf. This data set is highly multilingual, covering 20+ languages. The data is characterised by a very structured, levelled approach. Upper levels contain classificatory numbers (/ 36 science / 3611 humanities …). Through Coreon's multilingual concept map approach, the data is explorable in any of the available languages. For instance, switch to Spanish as source language and see immediately how the concept map is rendered in Spanish.

Fig. 7 Coreon SPARQL endpoint as an ELG-compatible Lexical/Conceptual Resource

6 Conclusions

Overall, the ELG project has succeeded in its aim to offer a broad variety of different service types covering many languages, and supplied by a range of different providers both academic and industrial. All the major classes of LT services are well represented in the ELG catalogue including ASR, MT and text analysis, with further classes of interest emerging during the course of the project. The generic design of the LT service execution APIs means that even services that do not exactly fit an existing class can be easily accommodated in the ELG platform, for example the HENSOLDT services for keyword spotting in audio required no API changes at all, only an adaptation of the "try out" GUI mechanism.

Inevitably, the majority of early contributions to the ELG platform were from the original ELG project consortium members. This was expected and planned for in the original project proposal, and the pilot project funding system was designed to help broaden the contributor pool more quickly by incentivising providers to adopt the ELG formats and specifications. It has succeeded in this aim, and many more details can be found in the various pilot project chapters in Part IV. As the funded project draws to a close and the ELG platform begins to transition to its long term sustainable mode of operation, we are seeing an increasing number of third-party contributions from beyond the original consortium and pilot projects, which stands the ELG in good stead for its sustainability as a platform over the coming years.

References

Aksenov, Dmitrii, Peter Bourgonje, Karolina Zaczynska, Malte Ostendorff, Julián Moreno-Schneider, and Georg Rehm (2021). "Fine-grained Classification of Political Bias in German News: A Data Set and Initial Experiments". In: *Proceedings of the 5th Workshop on Online Abuse and Harms (WOAH 2021)*. Ed. by Aida Mostafazadeh Davani, Douwe Kiela, Mathias Lambert, Bertie Vidgen, Vinodkumar Prabhakaran, and Zeerak Waseem. Bangkok, Thailand: ACL, pp. 121–131. URL: https://aclanthology.org/2021.woah-1.13.pdf.

Bié, Laurent, Aleix Cerdà-i-Cucó, Hans Degroote, Amando Estela, Mercedes García-Martínez, Manuel Herranz, Alejandro Kohan, Maite Melero, Tony O'Dowd, Sinéad O'Gorman, Mārcis Pinnis, Roberts Rozis, Riccardo Superbo, and Artūrs Vasiļevskis (2020). "Neural Translation for the European Union (NTEU) Project". In: *Proceedings of the 22nd Annual Conference of the European Association for Machine Translation*. Lisboa, Portugal: European Association for Machine Translation, pp. 477–478. URL: https://aclanthology.org/2020.eamt-1.60.

Cunningham, Hamish, Valentin Tablan, Angus Roberts, and Kalina Bontcheva (2013). "Getting More Out of Biomedical Documents with GATE's Full Lifecycle Open Source Text Analytics". In: *PLOS Computational Biology* 9.2, pp. 1–16. DOI: 10.1371/journal.pcbi.1002854.

Dikici, Erinç, Gerhard Backfried, and Jürgen Riedler (2019). "The SAIL LABS Media Mining Indexer and the CAVA Framework". In: *Interspeech 2019, 20th Annual Conference of the International Speech Communication Association*. Ed. by Gernot Kubin and Zdravko Kacic. Graz, Austria: ISCA, pp. 4630–4631. URL: https://researchr.org/publication/DikiciBR19.

García-Martínez, Mercedes, Laurent Bié, Aleix Cerdà, Amando Estela, Manuel Herranz, Rihards Krišlauks, Maite Melero, Tony O'Dowd, Sinead O'Gorman, Marcis Pinnis, Artūrs Stafanovič, Riccardo Superbo, and Artūrs Vasiļevskis (2021). "Neural Translation for European Union (NTEU)". In: *Proceedings of Machine Translation Summit XVIII: Users and Providers Track*. Association for Machine Translation in the Americas, pp. 316–334. URL: https://aclanthology.org/2021.mtsummit-up.23.

Germann, Ulrich (2020). "The University of Edinburgh's submission to the German-to-English and English-to-German Tracks in the WMT 2020 News Translation and Zero-shot Translation Robustness Tasks". In: *Proceedings of the Fifth Conference on Machine Translation*. ACL, pp. 197–201. URL: https://aclanthology.org/2020.wmt-1.18.

Germann, Ulrich, Roman Grundkiewicz, Martin Popel, Radina Dobreva, Nikolay Bogoychev, and Kenneth Heafield (2020). "Speed-optimized, Compact Student Models that Distill Knowledge from a Larger Teacher Model: the UEDIN-CUNI Submission to the WMT 2020 News Translation Task". In: *Proceedings of the Fifth Conference on Machine Translation*. ACL, pp. 191–196. URL: https://aclanthology.org/2020.wmt-1.17.

Junczys-Dowmunt, Marcin, Roman Grundkiewicz, Tomasz Dwojak, Hieu Hoang, Kenneth Heafield, Tom Neckermann, Frank Seide, Ulrich Germann, Alham Fikri Aji, Nikolay Bogoychev, André F. T. Martins, and Alexandra Birch (2018). "Marian: Fast Neural Machine Translation in C++”. In: *Proceedings of ACL 2018, System Demonstrations*. Melbourne, Australia: ACL, pp. 116–121. URL: http://www.aclweb.org/anthology/P18-4020.

Kaalep, Heiki-Jaan and Tarmo Vaino (2001). “Complete Morphological Analysis in the Linguist’s Toolbox”. In: *Congressus Nonus Internationalis Fenno-Ugristarum Pars V*, pp. 9–16.

Kummervold, Per E, Javier De la Rosa, Freddy Wetjen, and Svein Arne Brygfjeld (2021). “Operationalizing a National Digital Library: The Case for a Norwegian Transformer Model”. In: *Proceedings of the 23rd Nordic Conference on Computational Linguistics (NoDaLiDa)*. Reykjavik, Iceland: Linköping University Electronic Press, Sweden, pp. 20–29. URL: https://aclanthology.org/2021.nodalida-main.3.

Leitner, Elena, Georg Rehm, and Julián Moreno-Schneider (2019). “Fine-grained Named Entity Recognition in Legal Documents”. In: *Semantic Systems. The Power of AI and Knowledge Graphs. Proceedings of the 15th International Conference (SEMANTiCS 2019)*. Ed. by Maribel Acosta, Philippe Cudré-Mauroux, Maria Maleshkova, Tassilo Pellegrini, Harald Sack, and York Sure-Vetter. Lecture Notes in Computer Science 11702. Karlsruhe, Germany: Springer, pp. 272–287. URL: https://link.springer.com/content/pdf/10.1007%2F978-3-030-33220-4_20.pdf.

Marinova, Zlatina, Jochen Spangenberg, Denis Teyssou, Symeon Papadopoulos, Nikos Sarris, Alexandre Alaphilippe, and Kalina Bontcheva (2020). “Weverify: Wider and Enhanced Verification for You Project Overview and Tools”. In: *2020 IEEE International Conference on Multimedia Expo Workshops (ICMEW)*, pp. 1–4. DOI: 10.1109/ICMEW46912.2020.9106056.

Papanikolaou, Konstantina, Harris Papageorgiou, Nikos Papasarantopoulos, Theoni Stathopoulou, and George Papastefanatos (2016). ““Just the Facts” with PALOMAR: Detecting Protest Events in Media Outlets and Twitter”. In: *Tenth International AAAI Conference on Web and Social Media*. Vol. 10. 2, pp. 135–142.

Pinnis, Mārcis and Toms Bergmanis (2020). “Tilde’s Neural Machine Translation Technology”. In: *Latvian Academy of Sciences Yearbook 2020*. Latvian Academy of Sciences, pp. 85–89.

Pontiki, Maria and Harris Papageorgiou (2015). “Opinion Mining and Target Extraction in Greek Review Texts”. In: *Proceedings of the 12th International Conference on Greek Linguistics (ICGL 12)*. Vol. 2. Freie Universität. Berlin, Germany, pp. 871–883.

Pontiki, Maria, Konstantina Papanikolaou, and Haris Papageorgiou (2018). “Exploring the Predominant Targets of Xenophobia-motivated Behavior: A Longitudinal Study for Greece”. In: *Proceedings of the Natural Language Processing meets Journalism Workshop (NLPJ 2018)*. Ed. by Octavian Popescu and Carlo Strapparava. ELRA.

Prokopis, Prokopidis and Stelios Piperidis (2020). “A Neural NLP toolkit for Greek”. In: *11th Hellenic Conference on Artificial Intelligence*, pp. 125–128. URL: http://nlp.ilsp.gr/setn-2020/3411408.3411430.pdf.

Reale, Emanuela, Grazia Battiato, and Serena Fabrizio (2019). “RISIS2: an innovative research infrastructure as a support for STI research community”. In: *ISSI*, pp. 2658–2659. DOI: 10.5281/zenodo.3478408.

Rehm, Georg, Stelios Piperidis, Kalina Bontcheva, Jan Hajic, Victoria Arranz, Andrejs Vasiļjevs, Gerhard Backfried, José Manuel Gómez Pérez, Ulrich Germann, Rémi Calizzano, Nils Feldhus, Stefanie Hegele, Florian Kintzel, Katrin Marheinecke, Julian Moreno-Schneider, Dimitris Galanis, Penny Labropoulou, Miltos Deligiannis, Katerina Gkirtzou, Athanasia Kolovou, Dimitris Gkoumas, Leon Voukoutis, Ian Roberts, Jana Hamrlová, Dusan Varis, Lukáš Kačena, Khalid Choukri, Valérie Mapelli, Mickaël Rigault, Jūlija Meļņika, Miro Janosik, Katja Prinz, Andres Garcia-Silva, Cristian Berrio, Ondrej Klejch, and Steve Renals (2021). “European Language Grid: A Joint Platform for the European Language Technology Community”. In: *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations (EACL 2021)*. Kyiv, Ukraine: ACL, pp. 221–230. URL: https://www.aclweb.org/anthology/2021.eacl-demos.26.pdf.

Roberts, Ian, Andres Garcia Silva, Miroslav Janosik, Nils Feldhus, Dimitris Galanis, Andis Lagzdiņš, and Rémi Calizzano (2022). *Deliverable D4.3 Services, Tools and Components (Final Release)*. Project deliverable; EU project European Language Grid (ELG); Grant Agreement no. 825627 ELG. URL: https://www.european-language-grid.eu/wp-content/uploads/2022/04/ELG-Deliverable-D4.3-final.pdf.

Roberts, Ian, Andres Garcia Silva, Miroslav Janosik, Andis Lagzdiņš, Nils Feldhus, Georg Rehm, Dimitris Galanis, Dusan Varis, and Ulrich Germann (2020). *Deliverable D4.1 Services, Tools and Components (First Release)*. Project deliverable; EU project European Language Grid (ELG); Grant Agreement no. 825627 ELG. URL: https://www.european-language-grid.eu/wp-content/uploads/2021/02/ELG-Deliverable-D4.1-final.pdf.

Roberts, Ian, Andres Garcia Silva, Miroslav Janosik, Andis Lagzdiņš, Nils Feldhus, Georg Rehm, Dimitris Galanis, Dusan Varis, and Ulrich Germann (2021). *Deliverable D4.2 Grid Content: Services, Tools and Components (Interim Release)*. Project deliverable; EU project European Language Grid (ELG); Grant Agreement no. 825627 ELG. URL: https://www.european-language-grid.eu/wp-content/uploads/2022/04/ELG-Deliverable-D4.2-final.pdf.

Schulz, Konstantin, Jens Rauenbusch, Jan Fillies, Lisa Rutenburg, Dimitrios Karvelas, and Georg Rehm (2022). "User Experience Design for Automatic Credibility Assessment of News Content About COVID-19". In: *Proceedings of HCI International 2022 – Late Breaking Papers*. Accepted for publication. 26 June-01 July 2022.

Straka, Milan (2018). "UDPipe 2.0 Prototype at CoNLL 2018 UD Shared Task". In: *Proceedings of CoNLL 2018: The SIGNLL Conference on Computational Natural Language Learning*. Stroudsburg, PA, USA: ACL, pp. 197–207.

Straka, Milan and Jana Straková (2020). "UDPipe at EvaLatin 2020: Contextualized Embeddings and Treebank Embeddings". In: *Proceedings of LT4HALA 2020 – 1st Workshop on Language Technologies for Historical and Ancient Languages*. Marseille, France: ELRA, pp. 124–129.

Straka, Milan, Jana Straková, and Jan Hajič (2019a). "Czech Text Processing with Contextual Embeddings: POS Tagging, Lemmatization, Parsing and NER". In: *Proceedings of the 22nd International Conference on Text, Speech and Dialogue (TSD 2019)*. Cham, Heidelberg, New York etc.: Springer, pp. 137–150.

Straka, Milan, Jana Straková, and Jan Hajič (2019b). "UDPipe at SIGMORPHON 2019: Contextualized Embeddings, Regularization with Morphological Categories, Corpora Merging". In: *Proceedings of the 16th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*. Stroudsburg, PA, USA: ACL, pp. 95–103.

Straková, Jana, Milan Straka, and Jan Hajič (2019). "Neural Architectures for Nested NER through Linearization". In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Stroudsburg, PA, USA: ACL, pp. 5326–5331.

Suominen, Osma, Mona Lehtinen, and Juho Inkinen (2022). *Annif and Finto AI: Developing and Implementing Automated Subject Indexing*. Macerata. DOI: 10.4403/jlis.it-12740.

Tablan, Valentin, Ian Roberts, Hamish Cunningham, and Kalina Bontcheva (2013). "GATECloud.net: A Platform for large-scale, Open-Source Text Processing on the Cloud". In: *Philosophical Transactions of the Royal Society A: Math., Phys. and Eng. Sciences* 371.20120071.

Tiedemann, Jörg and Santhosh Thottingal (2020). "OPUS-MT – Building open translation services for the World". In: *Proceedings of the 22nd Annual Conference of the European Association for Machine Translation (EAMT)*. Lisboa, Portugal: European Association for Machine Translation, pp. 479–480. URL: https://helda.helsinki.fi/bitstream/handle/10138/327852/2020.eamt_1_499.pdf.

Chapter 8
Datasets, Corpora and other Language Resources

Victoria Arranz, Khalid Choukri, Valérie Mapelli, Mickaël Rigault, Penny Labropoulou, Miltos Deligiannis, Leon Voukoutis, and Stelios Piperidis

Abstract This chapter provides an overview of what is available in ELG in terms of datasets, corpora and other language resources (LRs) and how this has been achieved. We look at the procedures and steps that have been followed to complete the full resource ingestion cycle, which goes from repository and LR identification to metadata description and ingestion. We explain the approaches, priorities and methodology. The chapter also outlines the repositories that have been integrated into ELG, discussing the different procedures followed (metadata conversion, extraction, and completion, as well as harvesting) and the reasons behind these choices. Furthermore, the ELG catalogue content is described, with details on key elements and features as well as accomplishments. The last two sections are devoted to the crucial legal issues behind such a complex platform and its data management plan, respectively.

1 Introduction

As introduced in Part I, one of the ELG platform's primary functions is enabling sharing, distribution and deployment of Language Resources and Technologies (LRT). ELG provides access to thousands of datasets, by far the largest collection of relevant datasets for the European Language Technology community. Users can search for, download as well as provide different types of resources. As can be seen further down, ELG has identified, filtered, described and centralised a vast amount of datasets and other resources from different inventories and repositories, providing an easy to use point of search for the LT community. Its aim is to become the "yellow pages" and the primary platform for the European Language Technology community (see Chapter 9). Our work in terms of curating and further enriching ELG is ongoing, with new ingestions and collaborations at the time of writing.

Victoria Arranz · Khalid Choukri · Valérie Mapelli · Mickaël Rigault
ELDA, France, arranz@elda.org, choukri@elda.org, mapelli@elda.org, mickael@elda.org

Penny Labropoulou · Miltos Deligiannis · Leon Voukoutis · Stelios Piperidis
Institute for Language and Speech Processing, R. C. "Athena", Greece,
penny@athenarc.gr, mdel@athenarc.gr, leon.voukoutis@athenarc.gr, spip@athenarc.gr

G. Rehm (ed.), *European Language Grid*, Cognitive Technologies,
https://doi.org/10.1007/978-3-031-17258-8_8

This chapter describes the work carried out so far as well as currently ongoing efforts towards the population of the ELG catalogue with Language Resources (datasets and language models). This work has consisted in 1. the identification of sources (inventories and repositories), language resources and models, 2. their analysis, 3. the selection of elements to be ingested, as well as 4. the conversion or harvesting of their metadata descriptions and 5. the ingestion of these descriptions, and actual LRs, if relevant. All these steps are complex and intertwined tasks that are operationalised in a collaborative manner.

As a core element of ELG, the term "Language Resource" (LR, LRs) is used for resources composed of linguistic material used in the development, improvement or evaluation of Language Technologies (LT, LTs), but also, in a broader sense, in language and language-mediated research studies and applications; examples include datasets of various types, such as textual, multimodal or multimedia corpora, lexical data, grammars, language models, etc. In related initiatives and the literature, the term is often used with a broader meaning, encompassing also tools and services used for the processing and management of datasets, and standards, guidelines and similar documents that support the research, development and evaluation of LTs. In the ELG metadata model (see Labropoulou et al. 2020, and also Chapter 2), we use the term as first defined for the META-SHARE metadata model (Gavrilidou et al. 2012), i. e., including both data resources and LT tools/services. The alternative term Language Resource/Technology (LRT) is also used in the context of ELG (Rehm et al. 2021). However, in this chapter we use LR as referring to datasets and language models only; tools and services in ELG are discussed in Chapter 7.

2 Identification of Language Resources and Repositories

ELG aims to become the primary marketplace for the European LT community. The organisations making use of it range from commercial to non-commercial, including research centres and companies, as well as initiatives and infrastructures, among others. Linking all these players and supporting them in their interaction is a two-fold mission, which involves helping them make their tools, services and data available and also establishing the means for them to find and have access to those they may require in their work.

To cover all relevant existing language resource repositories, ELG defined an identification and collection methodology. First, the ELG project consortium members performed a round of identification and analysis contributing their own resources. Second, we reached out to the ELG National Competence Centres (NCCs, see Chapter 11) to gather more input and pointers to additional existing repositories and resource inventories. This identification task has been run in parallel with a priority definition task, which has been adjusted regularly according to achievements and to the community's needs and demands.

2.1 Identification by the Consortium

ELG examined the available inventories and repositories of all potential LT/LR providers and users. The initial results have been completed with further collaborative input from the NCCs (see Section 2.2) and ELG's sister project European Language Equality (ELE, see Section 2.3.2). With regard to the typology of LRs searched for, all types and modalities deemed useful for some sort of LT application were considered. These comprise corpora, lexicons, terminologies, and derived resources (such as language models for ASR or TMX models for MT), and also focus on media such as speech/audio, text, video/audio-visual, images, OCR and sign language datasets (images, videos). The identification strategy was adjusted following initial findings. For example, users' needs guided us to take into account high-priority dataset types such as language models, and has led us to look into repositories which contain and even focus on such types of resources (see Section 4.2).

2.2 Identification by the National Competence Centres

In addition to the work described above (Section 2.1), a survey was carried out to gather more input from the NCCs and from other collaborators, often related to their local and regional repositories (Rehm and Marheinecke 2019). This way we have been able to identify new repositories and, moreover, we were also provided with extensive documentation by the NCCs (content, contacts, etc.). The collaboration with the NCCs has been valuable. We plan to continue the joint work to maximise ELG's coverage.

2.3 Collaboratively Filling the Gaps

With its (at the time of writing) 8,873 dataset descriptions and following the ingestion of several repositories, ELG is at a compelling stage for taking the next steps in its dataset provision strategy. It must be stressed that our collaboration with other initiatives has also had an impact on these numbers. Bearing that in mind, the population of ELG now follows the analysis and identification of gaps from several perspectives:

1. The ELG consortium members' analysis of contributions and ingestion statistics in the platform.
2. The analysis of gaps carried out under a joint strategy, such as the ELE project and the ELG pilot projects (see Part IV), which have contributed datasets and also shared their own needs with regard to ELG, thus supporting ELG on its LRT collection venture from the point of view of the provider and the user.

3. The analysis of feedback received from technology developers and data users who shared their needs with us.

2.3.1 Contributions from the ELG Pilot Projects

The ELG pilot projects were intended to demonstrate the usefulness of ELG by contributing datasets or services to the platform or by making use of existing datasets or services for the development of innovative LT applications. These contributions provided by the pilot projects benefit both the community that will have access to the assets provided as well as the pilot projects themselves that will gain visibility with their work and by displaying it in ELG. These projects are an excellent proof of concept for the ELG platform and those pilot projects that provide datasets often target – and fill – specific gaps. At the time of writing, the already concluded pilot projects have finished their work, which has resulted in a set of 52 datasets available through ELG. The pilot projects are described in detail in Part IV of this book.

2.3.2 Contributions from the European Language Equality Project

ELG collaborates with the European Language Equality (ELE) project[1] to promote digital language equality in Europe. In 2021, ELE organised an online survey addressed primarily to the more than 30 language experts of the consortium to collect information on language resources and technologies available for the languages[2] under investigation (see Chapter 6 for more details). Through a web form, the ELE consortium partners responsible for one or more of the languages addressed by the project were able to record and report new language resources and also new resource repositories. This additional and collaborative collection procedure resulted in approx. 6,300 records (Arranz et al. 2022), which have already been cleaned up, normalised and curated and finally ingested into ELG (4,127 metadata records for data resources and 2,215 metadata records for tools). Just like ELG organisation pages, metadata records can be claimed by the resource creators or other rightful owners (see Chapter 9, Section 3.3, p. 179) and enriched with further information. This is why all contact persons included in these metadata records have been notified of their publication in ELG; we encouraged them to claim their resources and enrich the descriptions. Complete metadata descriptions are an important aspect of ensuring findability and future reuse of the resources (see Chapter 2, Section 7).

[1] https://european-language-equality.eu

[2] https://european-language-equality.eu/languages/

2.3.3 Platform Users

Finally, users of the ELG platform can also provide feedback about their interaction with ELG or about unmet expectations with regard to the availability of datasets or LT services. With regard to the latter, if users raise a certain need for specific datasets in relation to specific technologies, the ELG team can investigate whether relevant datasets or resources exist.

3 Integrating Repositories into ELG

The individual ELG releases follow an evolutionary strategy with regard to the population of the catalogue. This strategy has evolved as procedures have been put in place and new priorities and needs identified. ELG Release 1 (R1) followed a rather pragmatic approach, exploring procedures while targeting large repositories under the management of ELG consortium members. This allowed us to set up procedures, locate flaws and address problems (e. g., pending legal issues). ELG Release 2 (R2) launched an ambitious acquisition of very large catalogues which were not compliant with ELG's structure and metadata schema. This was the case, for instance, for Quantum Stat and Zenodo (see Section 4 and Arranz et al. 2021). Repositories like Zenodo are extremely large digital libraries in which many different research artefacts are published, which is why it requires a certain amount of effort to find and extract artefacts that are relevant for ELG. Despite these challenges, the overall result is rewarding as it provides access to many LT-related datasets, which have not been directly discoverable so far and which are now made available to the community through ELG as a one-stop-shop. The LR provision strategy for ELG Release 3 (R3) has built on top of the processes firmly established in R2. It continued and finished up the integration of the already initiated repositories, it set up harvesting procedures for as many ingested repositories as possible and added further repositories.

3.1 Priorities in the Ingestion Work

The list of identified repositories comprised different types of portals, such as those storing data from evaluation campaigns or shared tasks (e. g., WMT resources, Yeganova et al. 2021), large catalogues of language resources (e. g., ELRA, Mapelli et al. 2022), networks of LR repositories (e. g., various META-SHARE nodes, Piperidis et al. 2014), databanks, initiatives supporting the collection of language data, etc. This initial list was prioritised by taking into account the following dimensions of the different repositories:

- Relevance of their content for ELG, its services and users.
- Access information (conditions of use, prioritising open licensing schemes).
- Languages covered (covering multiple different languages, filling detected gaps).

- LR typology (covering different modalities, filling detected gaps).
- Number of resources (prioritising repositories with larger numbers of resources).
- Metadata schema (prioritising schemas that allow automated conversions).

Following this prioritisation strategy, three repositories – all of which are run by members of the ELG project consortium – were initially selected for ingestion in ELG Release 1: ELRA[3], ELRC-SHARE[4] and the three META-SHARE nodes managed by DFKI[5], ELDA[6] and ILSP[7]. This choice was strategic, as a proof of concept for resource availability and metadata conversion, given that the involved partners were familiar with the content and metadata schemas of these repositories. All the datasets selected for metadata ingestion were filtered down for legal compliance to ensure that licensing or distribution conditions that could not be addressed by ELG at this early stage could be taken care of for a later release. ELG Release 2 continued with additional repositories under the management of ELG project consortium partners (ELRA-SHARE-LRs 2014, 2016, 2018 and 2020[8], and LINDAT/CLARIAH-CZ[9]) but also by extending its work on the META-SHARE network and looking into very large digital inventories such as Quantum Stat and Zenodo. The reasons behind these choices combined strategy and diversity, which were also the goal with repositories such as Hugging Face for ELG Release 3 (see Section 4.2.4).

3.2 Contributing Language Resources

Interested institutions or individuals can make datasets available for download, i. e., hosting datasets in the ELG platform, or they can simply point ELG users to external download locations. In both cases, a description of the resource in the form of a metadata record is needed that can be discovered through the ELG catalogue. Such metadata descriptions can be manually created in ELG using the corresponding editor, they can be prepared as an XML file, which is then uploaded and imported into ELG, or they can be automatically converted from existing metadata records that use a different schema and imported into ELG afterwards. The flexibility behind these different options to populate the ELG catalogue makes contributions very easy, they can be done according to the provider's needs and preferences.

ELRC-SHARE follows the metadata-only option; this repository is financed by the European Commission under the ELRC initiative (Lösch et al. 2021), datasets will be available through ELRC-SHARE for at least the duration of the ELRC contracts. For that reason, the master copies of the LRs provided to ELG remain within

[3] http://catalogue.elra.info

[4] https://elrc-share.eu

[5] http://metashare.dfki.de

[6] http://metashare.elda.org

[7] http://metashare.ilsp.gr:8080

[8] LRs contributed by LREC participants, see http://www.elra.info/en/lrec/shared-lrs/.

[9] LINDAT is the CLARIN Centre for Language Research Infrastructure in the Czech Republic.

ELRC-SHARE but corresponding metadata records are available through ELG, enabling their discovery through ELG and their download via a redirect to the corresponding ELRC-SHARE page. In addition to contractual reasons, some repositories prefer to host their LRTs themselves, such as the ELRA catalogue, which distributes its LRs under a typology of licences that cannot be fully covered or recreated by the ELG metadata schema for the time being. Repositories like Zenodo or Quantum Stat mostly provide links to the locations of their datasets, very often these are links to Github or Gitlab pages. Again, only metadata records with the links to the dataset locations have been ingested into ELG. Likewise, harvested repositories only export metadata records (e. g., different CLARIN nodes or Hugging Face).

4 Procedures to Ingest Language Resources

Different repositories need to be approached differently with the goal of extracting metadata records and ingesting them into ELG. This relates to a number of dimensions that have allowed us to categorise repositories and, thus, to set up procedures to process them. These relate to the *conversion*, *extraction and completion* as well as *harvesting* of LR metadata, further described in Sections 4.1, 4.2 and 4.3 below.

4.1 Metadata Conversion

We converted (through mapping) the metadata records of several repositories so that we could import them into the ELG catalogue, which follows the ELG metadata schema (Labropoulou et al. 2020). This was the case for the ELRA catalogue, the META-SHARE nodes and the initial ingestion of the ELRC-SHARE repository (managed through harvesting now, see below). This conversion work is complex, but it has paved the way for improvements and updates on both sides of the conversion line, on both the source and target metadata elements and descriptions.

4.1.1 From ELRA Catalogue to ELG

The conversion of the LR metadata entries in the ELRA catalogue into the ELG metadata format followed several steps:

- *Updating the ELRA catalogue XML Schema Definition (XSD):* The ELRA catalogue is based upon the META-SHARE structure, it has been adapted to ELRA's specific distribution requirements. Before proceeding with the metadata conversion, an analysis of discrepancies between the META-SHARE XSD and the ELRA catalogue XML files was performed. This allowed us to update the ELRA catalogue XSD and to export the XML files in META-SHARE 3.1 format.

- *Mapping between META-SHARE 3.1 and ELG-SHARE 1.0.2:* Once exported, the ELRA XML files were mapped to the ELG metadata schema 1.0.2. This mapping allowed us to adapt the validated ELRA XML files (in META-SHARE 3.1 format) and to make them compliant with the ELG-SHARE model. Several elements had to be adapted for that purpose.
- *Conversion from META-SHARE 3.1 to ELG Metadata Model 1.0.2:* Once the mapping between the ELRA catalogue and ELG was completed, we implemented an XSLT stylesheet to transform the META-SHARE 3.1 format to the ELG metadata model.

While the implementation of this first tool required quite a bit of effort, the experience gained was valuable for the subsequent implementation of other converters.

4.1.2 From META-SHARE to ELG

META-SHARE's DKFI, ELDA and ILSP nodes are based on META-SHARE XSD 3.0. An already existing XSLT stylesheet was used to convert from META-SHARE XSD 3.0 to 3.1. We implemented a second XSLT stylesheet to convert META-SHARE 3.1 XML files into ELG metadata 1.0.2 (as for the ELRA-SHARE conversion into ELG). This modular approach allowed us to use META-SHARE v3.1 as pivot schema, reusing the implemented XSLTs stylesheets for further conversions (such as ELRC-SHARE's below).

4.1.3 From ELRC-SHARE to ELG

ELRC-SHARE is also based on META-SHARE. The initial ingestion was carried out through conversion, a harvesting protocol was put in place later (see Section 4.3 and Chapter 6 in Part I). To benefit from the ELRA to ELG metadata converter, a subset of ELRC-SHARE LRs was converted first into the ELRA and then into the ELG format.

4.1.4 Import into ELG

The XML files converted from the metadata of the different repositories were then imported into ELG using the API developed for this purpose. Some inconsistencies remained that led to corrections both in the XML files and the ELRA catalogue.

4.2 Metadata Extraction and Completion

Now we look into those repositories that did not allow for a straightforward conversion or for which building converters was not a feasible option.

4.2.1 Zenodo

Zenodo[10] is a digital library launched in May 2013 within the OpenAire[11] project, to enable the compilation of research artefacts, such as publications, images, datasets, software, etc. A good number of those artefacts consists of LRs that may be of interest to the LT community. However, the extremely high number of artefacts in Zenodo together with the incompatibility of the Zenodo and ELG metadata schemes made the identification of relevant LRs a big challenge. We opted for a semi-automatic approach to collect what ELG considers as LRs, using a combination of Python and directly querying the Zenodo database, among others.[12] However, the compilation of metadata information still required manual intervention to ingest our selection of actual LRs as well as to add the minimal set of metadata elements which are mandatory for ELG and which do not exist in the Zenodo records. This semi-automated process required a lot of manual effort. We currently work on an automated harvesting-oriented approach (see Section 4.3 and Chapter 6 in Part I).

4.2.2 ELRA-SHARE-LRs

The ELRA-SHARE-LRs are provided by participants attending the Language Resources and Evaluation Conference (LREC). Participants can share the LRs they present at the conference either by uploading them in a special LREC repository or by linking them to their original download location using an online form. We selected a subset of these LRs by checking the compliance of licences with the ones accepted in ELG. Licences that are too vague were left aside (e. g., "Open Source", "Creative Commons" without further specification). Given that the original metadata was available as a spreadsheet, the sheet and conversion tool produced to gather Zenodo metadata (see above) was adapted. As the ELRA-SHARE-LRs metadata contained only a minimal set of information, missing but required information was added manually into the spreadsheet to comply with the mandatory ELG metadata (e. g., type of LR, linguality, annotation, data format, licence, etc.). Finally, the spreadsheet was converted into XML and ingested into ELG.

4.2.3 Quantum Stat

Quantum Stat enables LR producers to register datasets in the "Big Bad NLP Database".[13] The procedure for identifying, describing and ingesting datasets into ELG is as follows: first, an initial table with 481 datasets was exported and analysed for relevance to ELG by checking licensing information (whether licences are well

[10] https://zenodo.org

[11] https://www.openaire.eu

[12] https://developers.zenodo.org/#records

[13] https://datasets.quantumstat.com

identified), dataset type, and whether the resource can be downloaded. The datasets not complying to the LR description requirements were discarded and only compliant metadata information was kept. Then, as for ELRA-SHARE-LRs and Zenodo, the minimal set of metadata information was compiled, while also adding missing information before the actual conversion into XML and ingestion into ELG.

4.2.4 Hugging Face

Often described as a "model zoo", the Hugging Face[14] repository includes a large collection of machine learning models and datasets that can be used for training new models, with a focus on the Transformers architecture (Wolf et al. 2020). ELG collaborates with Hugging Face regarding the import of Hugging Face metadata records into ELG. One challenge relates to the fact that the description of resources in Hugging Face does not follow a specific methodology. To begin with, adding descriptions to resources is encouraged but not mandatory. Furthermore, the suggested metadata elements do not follow a standard schema. The manual work needed to process the filtered entries was considerable in order to enrich the information available. A conversion process was applied based on mapping the elements (see Chapter 6 for more details).

4.3 Metadata Harvesting

We implemented metadata harvesting solutions for ELRC-SHARE, LINDAT/CLARIAH-CZ, CLARIN-PL and CLARIN-SI as well as Zenodo, as described below.

4.3.1 ELRC-SHARE

Three groups of datasets were originally selected from the three prioritised repositories to be converted and ingested into ELG Release 1 (see Section 4.1). Of these, only ELRC-SHARE allowed for the import of the whole list given that its resources met the following conditions: their licensing conditions allowed it (all data were shared under CC-BY licences, they were open under the directive on the re-use of public sector information, or they belong to the public domain), and their metadata elements were compatible and fully covered by the ELG metadata schema. We have implemented an OAI-PMH[15] client that harvests metadata records compliant with the ELG metadata schema, and we use this for regular harvesting from ELRC-SHARE.

[14] https://huggingface.co

[15] Open Archives Initiative Protocol for Metadata Harvesting (2015).

4.3.2 LINDAT/CLARIAH-CZ

The LINDAT/CLARIAH-CZ repository makes its metadata available for harvesting through its OAI-PMH end-point.[16] Means for ingesting metadata complying to the META-SHARE schema[17] were already in place in ELG and the repository did provide a mapping from its internal metadata storage to META-SHARE. An attempt was made at reusing this conversion, but the result was deemed unacceptable as not all of the available metadata was mapped. After a few iterations we arrived at a mapping between concepts that are important and required in the ELG schema and the metadata stored in LINDAT/CLARIAH-CZ. LINDAT updated the metadata for several of its resources following the feedback received from ELG. Also, based on the feedback from LINDAT/CLARIAH-CZ, some changes were applied to the ELG schema. The implementation of this mapping represents around 1,200 changed lines of code, including some tooling to reflect some of the metadata issues discovered.[18]

4.3.3 CLARIN-PL and CLARIN-SI

The LINDAT/CLARIAH-CZ repository makes available an OAI-PMH endpoint which exposes ELG-compatible metadata records. The repository software developed by the LINDAT/CLARIAH-CZ team, based on DSpace, is also used by several other CLARIN centres for their repositories, i. e., their metadata records are ready to be imported into ELG using the same harvesting procedure. For ELG Release 3, this collaboration has resulted in the regular harvesting of the CLARIN centres in Slovenia (CLARIN-SI) and Poland (CLARIN-PL).[19]

4.3.4 Zenodo

As described in Chapter 6 (Part I), Zenodo is a particularly interesting catalogue for ELG purposes. Zenodo exposes its metadata records through a REST API[20] as JSON data and through an OAI-PMH API[21] in a set of standard metadata formats, i. e., DC[22], DataCite[23], MARC21[24] and DCAT[25]. Work is currently ongoing to replace the semi-manual import of Zenodo metadata records that started for ELG Release

[16] http://lindat.mff.cuni.cz/repository/oai/request?verb=Identify

[17] http://www.meta-share.org/p/93/Documentation

[18] https://github.com/ufal/clarin-dspace/pull/930

[19] http://www.clarin.si and https://clarin-pl.eu

[20] https://developers.zenodo.org/#rest-api

[21] https://developers.zenodo.org/#oai-pmh

[22] https://www.dublincore.org/specifications/dublin-core/dcmi-terms/

[23] https://schema.datacite.org/meta/kernel-4.4/

[24] https://www.loc.gov/marc/bibliographic/

[25] https://www.w3.org/TR/vocab-dcat-3/

2 with a more automated process taking advantage of the standard protocols and schemas offered by Zenodo. This task involves a number of challenges that we are currently addressing with regard to the selection of the source API, the selection and conversion of metadata, the selection of a subset of the downloaded metadata records and the setting-up of an automated procedure for regular harvesting.

5 Language Resources in the ELG Catalogue

After the most recent ingestions of datasets as well as the contributions from the pilot projects and ELE, the ELG catalogue has reached a total of 8,873 metadata entries in April 2022, far exceeding our expectations when we started the project. The majority of these are description records without the data being hosted in ELG (103 resources are fully available through ELG). However, even if not available through ELG directly, most datasets are available through the referenced repository page, often available for download, which is reflected in the ELG catalogue too. Figures 1 and 2 illustrate the breakdown of repository sources ingested so far together with the breakdown of the current numbers per source.

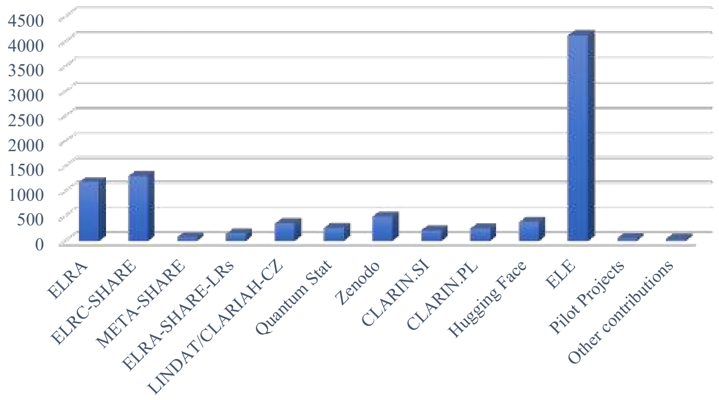


Fig. 1 Repository sources of the 8,873 datasets available in ELG in April 2022

Regarding resource types and their linguality, Figure 3 illustrates the numbers. As expected, the highest numbers apply to corpora (6,236 available in ELG), with twice as many monolingual corpora as bilingual ones (which in turn are three times as many as the multilingual ones). Lexical/Conceptual resources are also very well represented with 2,229 entries.

One of our bigger concerns at the time of Release 2 was the fact that there were barely any language descriptions (there were only 7). This has changed with the work towards ELG Release 3: at the time of writing, we count 408 language descriptions with the majority being monolingual. Further regarding language descriptions, the

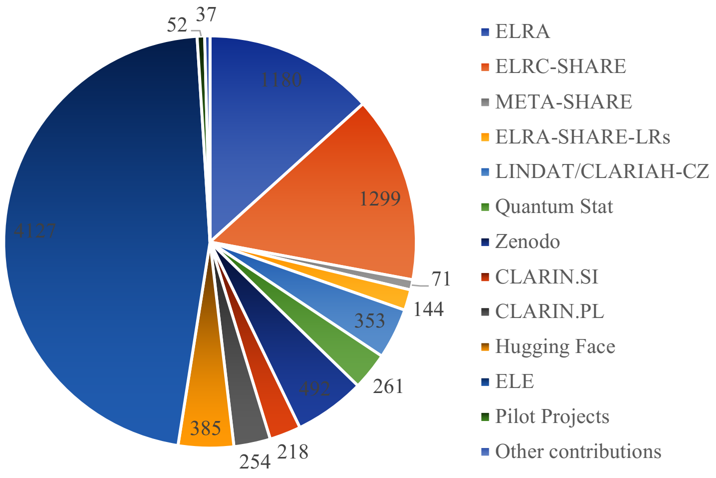


Fig. 2 Repository sources of the 8,873 datasets available in ELG in April 2022

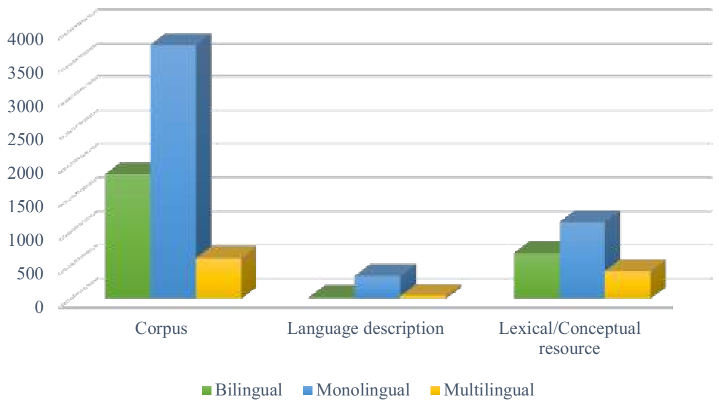


Fig. 3 Types of resources according to linguality

number of its "language models" subclass has increased to 358. This is good news as models are a popular and highly demanded resource type, currently providing the state of the art for many LT/NLP tasks. ELG is actively encouraging the use of its platform for the creation of models. The pilot projects have supported this resource type as well by contributing their models, too.

ELG also offers a very broad language coverage, with 450 languages represented by lexical/conceptual resources, and with corpora available in 438 languages, at the time of writing. The language models cover 156 languages, grammars are available for 25 languages. These are either monolingual or multilingual resources. Figure 4 shows the language resource type distribution for the EU official languages.

Finally, different media types are also represented in ELG. As expected, the largest number of resources belongs to the type "text" with more than 7,000 datasets.

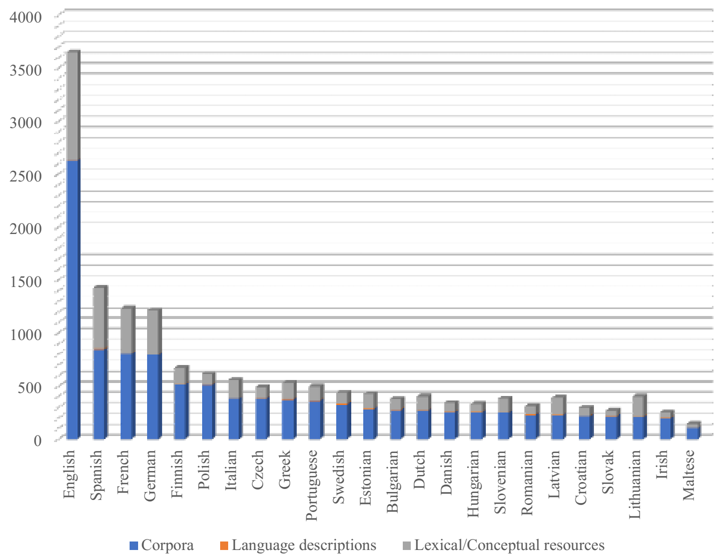


Fig. 4 Language resource type distribution for the official EU languages

Nonetheless, the type "audio" already offers more than 1,200 resources while currently 385 image and video resources are available.

6 Language Resources and Legal Issues

Managing legal issues in a large platform such as ELG implies taking care of a wide variety of legal aspects, often regarding licensing. It also implies taking into account processes that may differ from one provider to another. A provider may choose to distribute resources either through implicit or explicit licences, through specific conditions of use, or through considering a particular user status such as profiles or membership status. Moreover, the need to ensure GDPR compliance requires certain monitoring processes. For the development of the platform, the project has benefited from the support and advice of a dedicated team of legal experts who helped deploy the platform in a manner that is legally sound. This ranges from establishing the necessary legal context (e. g., Privacy Policy and Terms of Use) to stepping in for consultations. The legal team has also contributed to the preparation of a Data Management Plan (see Section 7). Below, we briefly describe some of the specific issues the ELG legal team has taken care of.

Advice on implicit versus explicit licences: One main distinction to make is the management of implied (or implicit) versus expressed (or explicit) licences. For implied licences, it has become a commonly and widely used practice to grant users access when they click on the licence terms acceptance button indicated on the repository pages.

Advice on conditions and terms of use: The conditions of use of a resource are another factor that has been defined and which may require further discussion and interaction between the provider and the user. Among the various elements to consider in licensing data or tools, we need to review the purpose of use (which could be commercial, for research, etc.), as well as the profile of the licencee (this is the type of institution, some resources may be restricted to particular types of institutions, e. g., *academic* or *commercial*)[26].

Financial and distributional issues: Not only legal issues may condition the delivery of resources to a user, but also the financial and distribution policies of the provider. Such policies involve a dedicated team, with expertise in technical, legal and financial domains. Parameters like the legal profile of the licencee, the purpose of use and the pricing policy need to be clearly displayed.

META-SHARE licensing: The selection of LRs for ingestion done for the three META-SHARE nodes needed to be revised due to licensing restrictions. These involved proprietary licences (e. g., MS-C-NoReD, MS-NC-NoReD and MS-Commons-BY-SA), as well as licences that required negotiations with providers. To address this, a study of the licences was performed by the ELG legal team for discussion with node managers. A proposal for licence mapping was drafted where non-restrictive licences were invited to move to Creative Commons licences. Restrictive licences were encouraged to move to more open licences, too.

Legal checking: The identification of various repositories demonstrated the importance of legal checking all throughout the information compilation process. In some cases (e. g., Zenodo), licences were well identified and could usually be integrated in the ELG metadata without further analysis. However, for other cases (e. g., ELRA-SHARE-LRs, Quantum Stat), legal information did not always comply with ELG requirements or was simply missing. Consequently, legal expertise was needed to either check and confirm the accuracy of present legal information, or to search for and gather the appropriate legal information.

Improvement of the licence list: When we processed the Zenodo datasets, we realised that several licences were not part of the ELG metadata values. Thus, the ELG legal expert was asked to compare the Zenodo list with the ELG list and make suggestions to integrate some of those licences into the ELG metadata. A list of 68 licences that did not correspond to ELG values was checked, out of which 40 could be added to the ELG licence list, whereas the other 28 did not need to be added because they were already used within ELG using other labels, they were not used, or they had no link.

Addition of conditions of use in the ELG metadata: We decided to add a new metadata field corresponding to the "conditions of use" associated to each identified licence to improve the search functionality for resources based on their licensing conditions. For "standard" licences, the conditions of use were added by the ELG team, based on information gathered from Creative Commons licences,

[26] https://live.european-language-grid.eu/terms-of-use