TROMPA

TROMPA: Towards Richer Online Music Public-domain Archives

Deliverable 5.1 v2

Data Infrastructure

Grant Agreement nr	770376
Project runtime	May 2018 - April 2021
Document Reference	D5.1-2 - Data Infrastructure
Work Package	WP5 - TROMPA Contributor Environment
Deliverable Type	Report
Dissemination Level	PU
Document due date	01-11-2020
Date of submission	31-10-2020
Leader	VD
Contact Person	David Linssen (david@videodock.com)
Authors	Bauke Freiburg (VD), David Linssen (VD) Christiaan Scheermeijer (VD), David Weigl (MDW), Aggelos Gkiokas (UPF), Alastair Porter (UPF)
Reviewers	Cynthia Liem(TUD)

Executive Summary

- The Trompa Data Infrastructure allows for interconnection between the different TROMPA software components, the Pilot Applications created out of these components and a number of linked data repositories. Communication between services can be carried out through HTTP requests (the CE API has GraphQL and REST interfaces) or through integration of one of the generic CE frontend components in a user facing application.
- It consists of three major components:
  - TROMPA's Contributor Environment, which provides the storage, interlinking and retrieval of musical data through a Neo4j property graph database exposed for query via a GraphQL endpoint. The graph database's data model employs widespread, established Semantic Web vocabularies in order to promote interoperability and reusability of its data.
  - TROMPA CE Processing Library, which provides unified mechanisms for the application of specific algorithms on CE referenced data. It acts as a conceptual organisation of software algorithms that integrate with the CE through an HTTP interface, orchestrating automated processing of information resources referenced by the CE's knowledge graph. It is currently comprised of a number of component APIs for multimodal query, display, and annotation of music resources, and automated assessment of scores and performances.
  - Personal online datastores (SOLID pods) that provide a mechanism for the storage of personal data. These enable users to retain control and ownership of their contributions, and act as user-identity providers for authentication with TROMPA applications. Contributions generated by a user's interactions with such applications are stored in the user's Pod as Linked Data, referenced by a URI which can be requested through an HTTP interface. An access control layer allows the user to selectively share or retain private access to each generated data item, or to open it to the public.
- The infrastructure is fully functional and has been implemented in multiple TROMPA WP6 End User Pilot applications.
- The most current version of the CE API specification can be found online in the form of API documentation[1] Any future iterations of the API will be updated here during the lifetime of the TROMPA project.
- This deliverable is complementary to the following deliverables
  - **D2.3.1 Technical Requirements**[2] describes the conventions, internal data model, data integration requirements, ontology, interfaces, integration of algorithm process application and integration of generic front-end components in extensive detail and functions as the reference manual for the contributor Environment. It has been made public together with the publication of this document, and D2.6, which describe the technologies developed for service integration (the Audio Commons

[1] http://api.trompamusic.eu/
[2] https://trompamusic.eu/deliverables/TR-D2.3.1-Technical_requirements_v1.1.pdf

Mediator) and present draft guidelines for adding new services to the Audio Commons Ecosystem (respectively).

- Future roadmap of the data infrastructure is documented as part of the GitHub ce-api repository[3] which will be kept up to date during the lifetime of the project.

[3] https://github.com/trompamusic/ce-api/projects

Version Log		
#	Date	Description
v0.1	28 October 2020	Initial version submitted for internal review
v0.2	30 October 2020	Revised version after internal review
v1.0	31 October 2020	Final version submitted to EU

Table of Contents

Table of Contents **5**

1. Introduction **6**

2. Requirements **7**

3. Overview **7**

3.1. Contributor Environment Architecture 7

3.2. TROMPA Processing library 9

3.3. Handling personal data within TROMPA 9

4. Functionalities **10**

4.1. API Queries and GraphQL interface 11

4.1.1 Queries 11

4.1.2 Mutations 11

4.1.3 Subscriptions 11

4.2. Components, integration of WP 3&4 and additional tooling 12

4.2.1 Components and integration of the WP 3 & 4 outcomes 12

4.2.2 Additional tooling 13

4.2.2.1 CE-Client library 13

4.2.2.2 CE Data Import library 13

4.3. Authentication and Privacy 13

4.3.1. Authentication in the CE 13

4.3.2 Obtaining access 13

4.3.3. Personal Data Storage 13

5. Hardware Infrastructure **14**

5.1 Contributor Environment 14

6. Future changes **14**

6.1 Roadmap 14

References **15**

1. Introduction

This document describes the released version of the Trompa Data Infrastructure (as planned and described in the first version of this document). The Trompa Data Infrastructure allows for interconnection between the different TROMPA software components, the Pilot Applications created out of these components and a number of linked data repositories. It consists of an API application that exposes functionalities to update and query a graph database (Neo4j) that contains a dataset complying with the CE internal data model, which is based on the schema.org structured data vocabulary. Communication between services can be carried out through HTTP requests (the CE API has GraphQL and REST interfaces) or through integration of one of the generic CE frontend components in a user facing application. The TROMPA Data Infrastructure consists of three major components:

1. TROMPA's Contributor Environment, provides the storage, interlinking and retrieval of musical data.
2. TROMPA CE Processing Library, provides unified mechanisms for the application of specific algorithms on CE referenced data
3. Personal online datastores (SOLID pods) that provide a mechanism for the storage of personal data. These enable users to retain control and ownership of their contributions, and act as user-identity providers for authentication with TROMPA applications

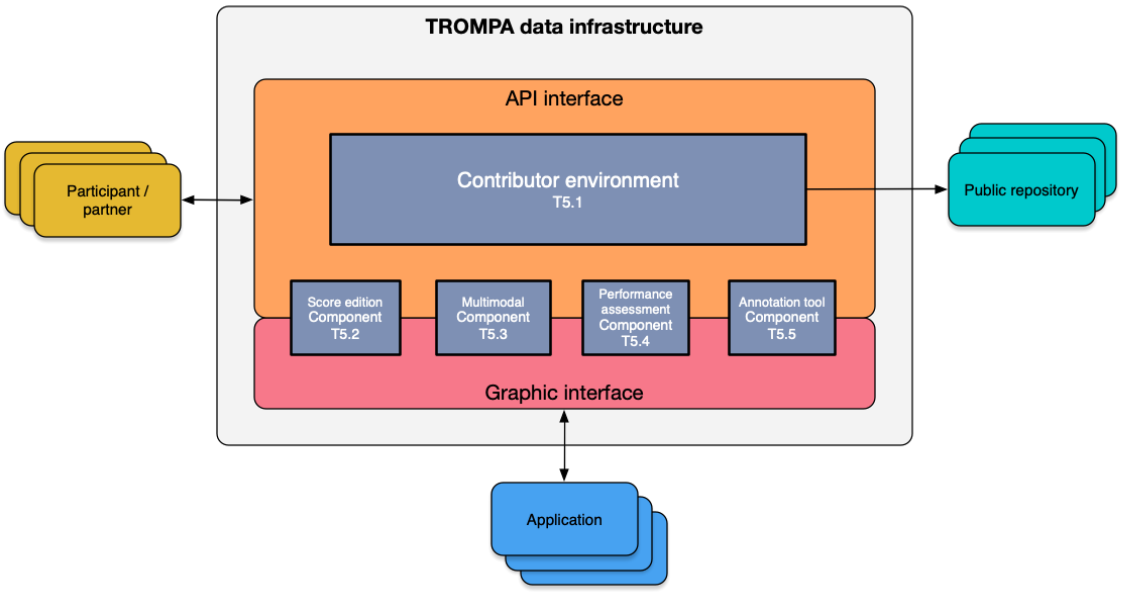

Figure 1.1. TROMPA Data Infrastructure

The Trompa Data Infrastructure, envisioned in the previous version of this document, has been delivered since March 19, 2020. The infrastructure is functional and has already been implemented in multiple TROMPA WP6 End User Pilot applications.

This document provides an overview of the CE architecture, the CE Processing Library, authentication and storing of personal data within TROMPA and provides an outlook on the expected evolution of the infrastructure.

2. Requirements

This deliverable is complementary to deliverable **D2.3.1 Technical Requirements**[4] which describes in detail the:

- Conventions,
- CE internal data model,
- Data integration requirements,
- Ontology,
- Interfaces
  - GraphQL interface for managing data,
  - REST interface to provide a unique URL for each node in the CE database and to provide JSON-LD output.
- Integration of algorithm process application,
- Integration of generic front-end components.

The D2.3.1 document functions as the reference manual for the Contributor Environment. It has been updated in parallel and will be made **publicly** available, in concordance with the release of this deliverable.

3. Overview

3.1. Contributor Environment Architecture

A primary motivation of the TROMPA project lies in the interconnection—rather than the integration and ingestion—of information in public-domain music repositories. It would be costly and counterproductive to attempt to supplant established repositories by copying entity descriptions and media representations into a centralized database under a unified data schema. Rather, we describe the contents of such repositories by reference, using URIs to address, interlink, and contribute layers of enriched descriptors and content to resources hosted in situ at their native (TROMPA-external) Web locations.

[4] https://trompamusic.eu/deliverables/TR-D2.3.1-Technical_requirements_v1.1.pdf

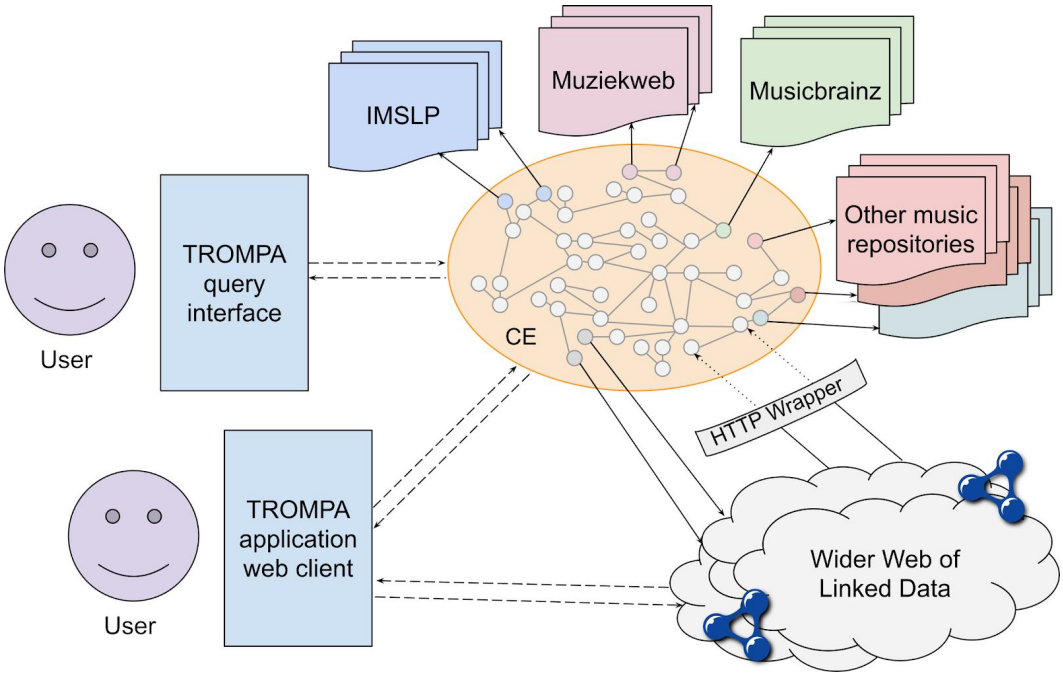

Figure 3.1. TROMPA's Contributor Environment houses a Neo4j graph database describing music resources hosted in external repositories. The HTTP wrapper assigns a URI to each node, exposing it as Linked Data (JSON-LD) when the URI is dereferenced. Dashed arrows: interaction; solid arrows: URI reference; dotted arrows: Linked Data-to-Neo4j translation provided by the HTTP wrapper.

Graph databases are ideally suited to support flexible, mutably specified interconnection of Web-based resources. TROMPA has opted to adopt a Neo4j property graph database for this purpose. This database, exposed for query via a GraphQL endpoint, forms the core of the TROMPA Contributor Environment (CE), a data infrastructure that also comprises a number of component APIs for multimodal query, display, and annotation of music resources, and automated assessment of scores and performances (Figure 1). Each node in the graph can be accessed via a persistent URI through an HTTP wrapper interface, providing a JSON-LD representation of the identified entity and its associated properties and values by reference to their persistent URIs, interweaving the CE graph with the wider Web of Linked Open Data.

The graph database's data model employs widespread, established Semantic Web vocabularies in order to promote interoperability and reusability of its data. Schema.org forms the core of this model, and is used as the primary means of describing Web resources (including those corresponding to persons, works, audiovisual recordings, and score encodings) and automated processes (as described in the next section). This vocabulary is complemented by the use of Dublin Core terms for bibliographic description; the Simple Knowledge Organization System (SKOS; Miles & Bechhofer, 2009) data model to describe interrelations between Web resources; the Web Annotation data model (Sanderson, Ciccarese, & Young, 2017) to annotate collection objects; and the PROV ontology to capture provenance traces relating to their creation and processing. Further pre-established vocabularies are used for data relating to domain-specialised tasks within the federated contribution layer (Section 3.3), including the Music, Timeline, and Segment ontologies (Raimond, Abdallah, S. Sandler, & Giasson, 2007; Fields, Page, De Roure, & Crawford, 2011) to

describe score-aligned performances for analysis or review in the music scholars (D6.3) and instrumental players (D6.5) use-cases.

3.2. TROMPA Processing library

Alongside the interconnection of publicly-licensed music information obtained from established Web repositories, TROMPA focuses on the enrichment of such information, through the application of Music Information Retrieval (MIR) technologies, and through the contributions of human music scholars, performers, and enthusiasts. Automated enrichment activities are centrally coordinated using the CE alongside the TROMPA Processing Library (TPL; D5.3).

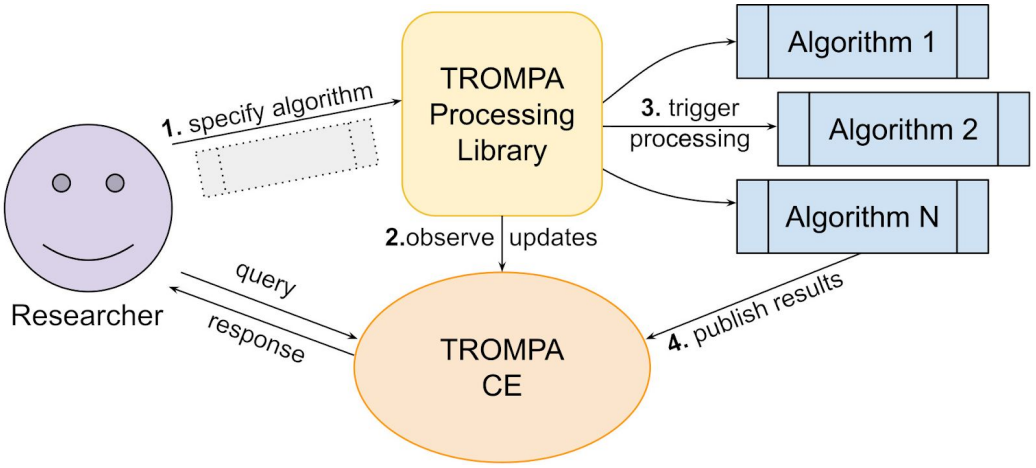

Figure 3.2. TROMPA Processing Library (TPL) workflow. (1.) Researcher specifies her algorithm for use with TPL. (2.) TPL continuously monitors the CE's graph. When a new node is ingested with a type matching a TPL algorithm's specification, the algorithm is triggered on the newly entered data (3.), before publishing the results to the CE (4.) , where they become available for query and may potentially trigger further processing orchestrated by the TPL.

The TPL acts as a conceptual organisation of software algorithms that integrate with the CE through an HTTP interface, orchestrating automated processing of information resources referenced by the CE's knowledge graph. A subscription mechanism (Figure 2) provides triggers for processing of newly created graph nodes fulfilling certain type constraints as they are added to the CE. References to processing outcomes—generally, MIR feature data—are themselves ingested into the knowledge graph, where they may trigger further activities orchestrated by the TPL, resulting in processing chains. These mechanisms allow researchers to modularly specify new algorithms or new software versions as they become available, and to run them on demand in response to the arrival of specific types of data.

3.3. Handling personal data within TROMPA

Where data is generated by automated processing of public-domain information, this data is published openly under public license by the TROMPA consortium. The situation is more complicated in the case of user contributions. Humans reporting on their subjective experiences, or

providing expert insights or artistic contributions, are understandably concerned about safeguarding their data, and indeed their rights to such safeguards are guaranteed by the EU GDPR and similar legislation.

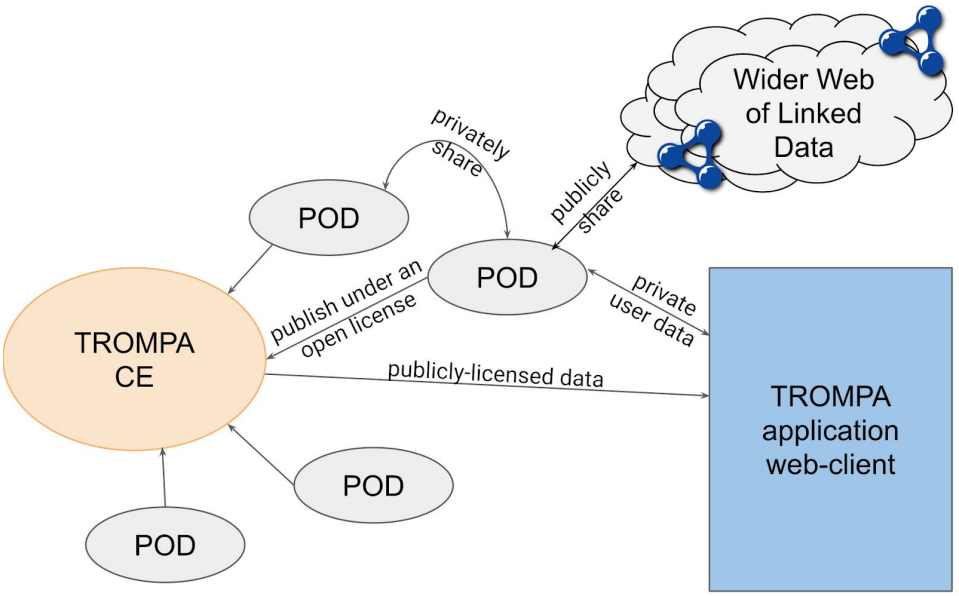

Figure 3.3. TROMPA's federated contribution model allows users to retain data ownership and access control, and to explicitly publish contributions to TROMPA under an open license.

To accommodate, TROMPA's data infrastructure employs a secondary, decentralized layer of personal online datastores (Solid Pods) that both enable users to retain control and ownership of their contributions, and act as user-identity providers for authentication with TROMPA applications (Mansour et al., 2016). Contributions generated by a user's interactions with such applications are stored in the user's Pod as Linked Data, referenced by a URI which can be requested through an HTTP interface. An access control layer allows the user to selectively share or retain private access to each generated data item, or to open it to the public. Users may additionally choose to publish their contributions with TROMPA under an open license (Weigl et al., 2020), at which point the relevant data is ingested into the CE's graph, making it discoverable by other TROMPA users. Through this mechanism, users are offered fine-grained access control over the information resources they generate through interaction with TROMPA applications, and retain ownership through the explicit, user-directed act of publication into the public domain.

4. Functionalities

This section provides a conceptual overview of the TROMPA data infrastructure. All functionalities are described in detail in D2.3.1 Technical Requirements including code examples.

The most up-to-date version of the API specification can be found online in the form of API documentation[5] which specifies the HTTP interface for building queries, performing mutations and setting up subscriptions for CE referenced data. It also provides information regarding authentication procedures.

4.1. API Queries and GraphQL interface

Three types of functionalities are accessible through the GraphQL API interface: Queries, Mutations and Subscriptions.

4.1.1 Queries

A query starts with the phrase 'query' and typically consists of:

- The name of the query (optional),
- Type of entity for which is queried,
- Conditions (optional),
- List of properties to be included in the response.

The result typically consists of a JSON object containing:

- The "data" object with the result(s),
- The name of the query responded to,
- The actual data, corresponding to the list of properties to be included.

4.1.2 Mutations

Mutations are queries that add, update or remove data in the database. The following mutations are possible:

- Creating, updating or deleting a node,
- Adding a relation between nodes (primitive types),
- Add a relation between nodes (Interfaced or Unioned types),
- Remove a relation between nodes.

4.1.3 Subscriptions

Subscriptions are available to listen to specific events. These events are triggered by adding nodes in the CE. The following subscriptions are available:

- ControlActionRequest,
- ControlActionMutation,
- ThingCreateMutation,
- MediaObjectCreateMutation,
- VideoObjectCreateMutation,

[5] http://api.trompamusic.eu/

- AudioObjectCreateMutation.

The ThingCreateMutation subscription is an abstract subscription which can be used to listen to one or multiple create mutations with a single subscription.

4.2. Components, integration of WP 3&4 and additional tooling

4.2.1 Components and integration of the WP 3 & 4 outcomes

The TROMPA data infrastructure allows for mid-level integration of components that will be further exploited in the WP6 pilots. In order to do so, the data produced in WP3 (musical repertoire, automatic descriptions and generated audio) and annotations delivered through WP4 have been made accessible and usable in reusable components, meeting common standards.

Component developers can query the CE database for **EntryPoints** that could potentially be interesting for its users. By implementing a user interface on the basis of the information in the (dynamic) template nodes **Property** and **PropertyValueSpecification**, the algorithm process (WP3/4) becomes available for a user.

After a user request is sent to the CE API, the Component is able set up a subscription to the instantiated **ControlAction** via websocket to any mutations to the created job. The CE would notify the Component of any updates done on the job, most likely by the algorithm process application. It is up to the algorithm process application to determine how fine-grained these updates are.

The Component can show these updates in its UI and act on process completion by making the results available to the user.

With the available result known, the Pilot is able to create additional relations from this result to other relevant nodes in the CE database, like *isBasedOn* to a **MusicComposition** or *copyrightHolder* to an **Organisation**. This will greatly improve the chances that subsequent users find and re-use the result file.

An implemented example of this mechanism can be found in the interaction between the Campaign Manager and the D4.1 Task Engine, both currently part of the WP6 Orchestra use case. In this example the Campaign Manager (CM), handling most user facing communication, uses the Data Infrastructure to communicate with the Task Engine to serve the proper tasks for crowd verification or annotation. The CM creates a **ControlAction** for each campaign. This **ControlAction** has a reference to the **EntryPoint** of potential tasks that need to be performed in order to complete the campaign. However, the Campaign Manager isn't aware of the content of the task nor when it is considered to be completed. Therefore the Campaign Manager uses a **ControlActionMutation** subscription to be able to determine the tasks status and present the user a "Thank you" message before navigating to the next task.

An up-to-date overview of all the available components will be published in month 34 of the project. The components in progress are available through the TROMPA GitHub repository[6].

[6] https://github.com/trompamusic

4.2.2 Additional tooling

4.2.2.1 CE-Client library

A CE Client is available as a Python library to read data from and write to the Contributor Environment. The library connects to an existing TROMPA CE Instance. For testing on a local environment the Docker containers[7] can be run.

Basic code examples and installation instructions for the library can be found on its GitHub page: https://github.com/trompamusic/trompa-ce-client

4.2.2.2 CE Data Import library

A data importing client is available as a Python library to import metadata to the Trompa Contributor Environment. Currently it supports importing metadata from MusicBrainz. Installation instructions for the library can be found on its GitHub page[8].

4.3. Authentication and Privacy

4.3.1. Authentication in the CE

All read operations in the Contributor Environment API are publicly accessible. However, in order to create, update, or delete nodes, the request **MUST BE** authenticated with a JWT token. Full authentication documentation is available here[9]. The following JWT Endpoints are available:

- For test environment use[10]
- For production environment use[11]:

4.3.2 Obtaining access

The **id** and **apiKey** can be requested from one of the TROMPA project partners. Alternatively, users can send an email to: info@videodock.com

4.3.3. Personal Data Storage

Solid[12] (Mansour et al., 2016) forms the basis of TROMPA's federated contribution layer (Section 3.3; see also Weigl et al., 2020). Solid is a Web decentralisation project building on a W3C standards-based Linked Data technology stack which aims to enable rich online interactions between users that retain data ownership with each individual user. This allows each user to retain fine-grained access control over their personal data, supporting sharing of data with specified users, and simple integration by reference with the CE. This emphasis on user choice and control of web-hosted data provide a pleasing fit to TROMPA's emphasis on FAIR and open data principles. Solid provides users with Personal Online Datastores (Solid Pods), which act as WebID identity

[7] https://github.com/trompamusic/ce-api
[8] https://github.com/trompamusic/ce-data-import
[9] https://github.com/trompamusic/ce-api/blob/staging/docs/authentication.md
[10] https://api-test.trompamusic.eu/jwt
[11] https://api.trompamusic.eu/jwt
[12] http://solidproject.org

providers as well as user-controlled storage spaces. This allows users to log in to TROMPA Web applications (alongside any other Solid-compatible applications) with their own WebID, as well as to privately share data with other users by reference to their WebIDs.

TROMPA's Pod provider (Solid server) is available online[13]. Due to the decentralised nature of Solid, users are free to choose this or any other Pod provider on the Web for interaction with TROMPA applications. Users with advanced technical expertise may additionally choose to self-host.

An exception has been made for the Choir WP6 use case, where users are currently using their Voctro Labs-supplied 'Voiceful' accounts to access the Choir prototype. For this use case, users can choose to share their voice recording data (for research purposes). This data will be stored in an Amazon AWS S3 bucket (maintained by Voctro Labs) and linked to the CE.

5. Hardware Infrastructure

5.1 Contributor Environment

The delivered TROMPA Data Infrastructure runs on AWS infrastructure. Both the CE API test and production environments are hosted in a AWS Fargate[14] service using a Docker container.

6. Future changes

6.1 Roadmap

With the delivery of the TROMPA Data Infrastructure and this document describing it, there are no further major changes envisioned within the scope of the TROMPA project. However, any future changes of the Data Infrastructure to satisfy pilot requirements (which are still in development) will be deposited on the GitHub page.[15]

[13] https://trompa-solid.upf.edu
[14] https://docs.aws.amazon.com/eks/latest/userguide/fargate.html
[15] https://github.com/trompamusic/ce-api

References

Fields, B., Page, K., De Roure, D., & Crawford, T. (2011). The segment ontology: Bridging music-generic and domain-specific. In *2011 IEEE International Conference on Multimedia and Expo* (pp. 1-6). IEEE. https://doi.org/10.1109/ICME.2011.6012204

Mansour, E., Sambra, A. V., Hawke, S., Zereba, M., Capadisli, S., Ghanem, A., ... & Berners-Lee, T. (2016). A demonstration of the solid platform for social web applications. In *Proceedings of the 25th International Conference Companion on World Wide Web* (pp. 223-226). https://doi.org/10.1145/2872518.2890529

Miles, A. & Bechhofer, S. (2009). SKOS Simple Knowledge Organization System Reference. *W3C Recommendation.* https://www.w3.org/TR/skos-reference/

Pugin, L., Zitellini, R., & Roland, P. (2014). Verovio: A library for Engraving MEI Music Notation into SVG. In *Proceedings of the 15th International Society for Music Information Retrieval Conference* (pp. 107-112). https://archives.ismir.net/ismir2014/paper/000221.pdf

Raimond, Y., Abdallah, S. A., Sandler, M. B., & Giasson, F. (2007). The Music Ontology. In *Proceedings of the 8th International Conference on Music Information Retrieval*. https://archives.ismir.net/ismir2007/paper/000417.pdf

Sanderson, R., Ciccarese, C., & Young, B. (2017). Web Annotation Data Model. *W3C Recommendation*. https://www.w3.org/TR/annotation-model/

Weigl, D. M. & Page, K. R. (2017). A framework for distributed semantic annotation of musical score: "Take it to the bridge!". In *Proceedings of the 18th International Society for Music Information Retrieval Conference*. http://archives.ismir.net/ismir2017/paper/000190.pdf

Weigl, D. M., Goebl, W., Hofmann, A., Crawford, T., Zubani, F., Liem, C. C. S., & Porter, A. (2020). Read/write digital libraries for musicology. In *Proceedings of the 7th International Conference on Digital Libraries for Musicology*. https://doi.org/10.1145/3424911.3425519