Populating the Smart Musical Instruments Ontology with Data

Luca Turchet
University of Trento
luca.turchet@unitn.it

Guixia Zhu
University of Trento
guixia.zhu@studenti.unitn.it

Paolo Bouquet
University of Trento
paolo.bouquet@unitn.it

***Abstract*—One of the main instances of Musical Things within the Internet of Musical Things (IoMusT) paradigm is the emerging family of Smart Musical Instruments (SMIs). This is a category of musical instruments encompassing sensors, actuators, embedded intelligence, and wireless connectivity to local networks and to the Internet. Recently, an ontology to represent this domain has been proposed, the Smart Musical Instruments Ontology. However, a database gathering SMIs instances was missing. Such a database would be useful to gather and organize information about this family of IoMusT devices which is predicted to become more widespread in the future, in both academic and industrial contexts. In this paper, we present a linked data service exposing metadata about SMIs structured according to the Smart Musical Instruments Ontology, as well as a Web application for its extension and retrieval. We first survey the existing instances of SMIs as well as the SMIs Ontology. We then outline use cases and potential applications utilizing the developed Web-based service, showing how to retrieve detailed information using metadata describing the SMIs domain.**

I. Introduction

The field of Semantic Web aims at accomplishing the vision of an extension of the existing World Wide Web through standards set by the World Wide Web Consortium (W3C) [1], [2]. This vision was born to transform the Web from a repository of human-readable information into an entity that allows for the machine-understandability of data. As a result, machines are enabled to make meaningful interpretations of data, which are in part alike to the way humans process information to achieve their goals. To make Internet data machine-readable, semantics are encoded with the data. This is achieved through technologies able to formally represent metadata, such as the Resource Description Framework (RDF) [3] for the representation of data as a set of triples subject–predicate–object, the Web Ontology Language (OWL) [4] for the definition of a vocabulary to state the meaning of represented data, the International Resource Identifiers (IRI) for the univocal identification of resources, as well as the SPARQL Update [5] and Query languages [6] for the storage and retrieval of data.

Semantic Web technologies and methodologies have been extensively applied to the field of the Internet of Things [7]. They have also impacted the field of Sound and Music Computing leading to the emergence of the Semantic Audio (SA) field [8]. SA is an interdisciplinary field providing techniques to extract structured meaningful information from audio, in both musical and non-musical contexts. The key concept in SA systems is the combination of two primary aspects: machine analysis and representation of the extracted information. Specifically, SA combines signal analysis to extract quantifiable acoustic features from audio, machine learning techniques to map the extracted acoustic features to perceptually, environmentally or musically meaningful features, and structured representations that place these features into possibly multi-relational or heterogeneous hierarchies [9], [10] typically using Semantic Web ontologies. SA methods find application in several music-related scenarios, from intelligent music production [11], to online music distribution [12]. SA technologies are increasingly being used in emerging paradigms linking the Sound and Music Computing field with that of the Internet of Things [13], such as the Internet of Musical Things (IoMusT) [14] or the Internet of Audio Things (IoAuT) [15], with the aim of fostering interoperability across heterogeneous devices.

One of the main instances of Musical Things within the IoMusT paradigm is the emerging family of Smart Musical Instruments (SMIs). This is a category of musical instruments encompassing sensors, actuators, embedded intelligence, and wireless connectivity to local networks and to the Internet [16]. Recently, an ontology to represent this domain has been proposed in [17], the Smart Musical Instruments Ontology. However, today a database gathering SMIs instances is missing. Such a database would be useful to gather and organize information about this family of IoMusT devices which is predicted to become more widespread in the future, in both academic and industrial contexts [14].

In this paper, we present a linked data service exposing metadata about SMIs structured according to the Smart Musical Instruments Ontology, as well as a Web application for its extension and retrieval. We first survey the existing instances of SMIs as well as the SMIs Ontology. We then outline use cases and potential applications utilizing the developed Web-based service, showing how to retrieve detailed information using metadata describing the SMIs domain.

II. Related works

A. Existing Smart Musical Instruments

To date, only a small number of SMIs have been developed in both academic and the industrial settings. We describe hereinafter examples of smart musical instruments and related use cases, which are included in the database developed in this study.

Sensus Smart Guitar. This instrument [18], crafted and developed by the company Elk, consists of a hollow body electro-acoustic guitar augmented with sensors embedded in various parts of the instrument, on-board processing, a system

ISSN 2305-7254

of multiple actuators attached to the soundboard, and interoperable wireless communication [19]. The utilized embedded system runs Elk Audio OS [20], a music operating system based on Linux, which guarantees round-trip latencies of 1 ms, supports music software plugins, as well as provides wireless connectivity via Bluetooth, Wi-Fi, and 5G. The internal sound engine that runs over Elk Audio OS affords a large variety of sound effects and sound generators, and is programmable via dedicated apps for desktop PCs, smartphones, and tablets. The low-latency wireless communication implemented in the smart guitar supports the exchange of MIDI, OSC and gRPC messages as well as audio signals. Some use cases for the Sensus Smart Guitar are reported in [19]. The instrument has been used to wirelessly control i) visuals delivered on screens by VJ programs such as VDMX by Vidvox, ii) audio plugins and functions of digital audio workstations running on laptops, as well as iii) elements of virtual environments provided through virtual reality headsets. The smart guitar has also been used to record audio files and share them directly on social networks.

HyVibe guitar. The HyVibe guitar [21] is an acoustic guitar which enables the production of multiple audio effects such as chorus, reverb, distortion and delay directly from the body of the guitar using a vibration control system [22]. The guitar also let players record song ideas and guitar licks to create backing tracks or share them on social networks using the cloud. The settings of the guitar can be controlled using a dedicated mobile app.

Other smart guitars. Two other smart guitar prototypes have been developed in academic contexts for investigating specific use cases. Both consist of a conventional classical guitar enhanced with an embedded processing board, Wi-Fi connectivity, and onboard sound delivery system. The first one, reported in [23], was utilized as a hub for collaborative music making in conjunction with connected performers using smartphones in co-located settings. The second guitar is described in [24] and was conceived to enable guitar players to retrieve songs from large online music databases using criteria different from conventional music search (e.g., input text), namely audio features (e.g., chords, keys, tempo) that were extracted from the audio signal resulting from the act of playing.

Smart cajón. The smart cajón prototype described in [25] and [26] is a conventional cajón augmented with sensors, motors for vibro-tactile feedback embedded in a cushion and Wi-Fi connectivity. The computational unit is the Bela board for low-latency audio and sensor processing. The audio engine is composed of a sampler and various audio effects (e.g., reverberation, delays, frequency shifter, equalizer). Sensor fusion and semantic audio techniques are utilized to estimate the location and nature of the performer strokes on the front and side panels. The stroke detection engine can be used to map strokes to different sound samples to simulate various percussive instruments, or produce automated score transcription. By mapping musical gestures to tactile stimuli using haptic wearable devices, it is possible to provide audience members with haptic feedback while performers play [27]. In another example application involving creative audience participation, the smart cajón provided notifications to the performer through tactile stimulation responding to audience intent produced via smartphones (e.g., start/stop playing).

Smart mandolin. This is a classic Neapolitan mandolin smartified with different types of sensors, a microphone, a loudspeaker, wireless connectivity to both local networks and the Internet, and a low-latency audio processing board [28]. Various implemented use cases were developed, which leverage the smart qualities of the instrument. These included the programming of the instrument via applications for smartphones and desktop computer, the wireless control of devices enabling multimodal performances such as screen projecting visuals, smartphones, and tactile devices used by the audience [27], as well as the interaction with online audio repositories in the context of technology-mediated audience participation [29].

Retrologue Hardware Synthesizer. This electronic SMI uses a virtual analog audio plugin in conjunction with a dedicated tangible interface and the Elk Audio OS platform for embedded audio [30]. The plugin is the Retrologue 2 by Steinberg. The adopted design approach started from an analog synthesizer, passed from its digital emulation, and returned to the analog domain via the real-time, physical control of the digital synthesizer [31].

B. The Smart Musical Instruments Ontology

The Smart Musical Instruments Ontology [32] was created to represent knowledge related to the emerging family of SMIs, with the aim of facilitating interoperability between SMIs as well as with other Musical Things interacting with them [17]. The ontology relates to several existing ontologies, including; the SOSA Ontology for the representation of sensors and actuators [33]; the Music Ontology that models the music value-chain from production to consumption [34]; the Audio Effects Ontology dealing with the description of digital audio effects [35], [36]; the Studio Ontology that represents the studio and music production environment [37]; the Audio Features Ontology that addresses descriptors representing specific characteristics of sound signals [38]; and the IoMusT Ontology for the representation Musical Things and IoMusT ecosystems [39].

The ontology is described by the metrics listed in Table I, which are based on the "knowledge coverage and popularity measures" proposed by Fernandez et al. [40]. The example in listing 1 shows how an SMI implementation is described in Turtle syntax, namely the Sensus Smart Guitar by Elk. This representation makes use of RDF Schema Language for describing properties and classes of RDF resources.

III. THE SMIS DATABASE

To create the database, we first surveyed the existing instances of SMIs and related services that are reported in Section II-A. Then we expressed the resulting data using the RDF and the domain ontology for describing SMIs reported in [39]. The data was then stored in an RDF triple store, which enables the retrieval of SMI-specific information using common query languages such as SPARQL. Specifically, we used the Apache Jena database, which offers the TDB triplestore [41]. TDB supports SPARQL Query and Update.

```
@prefix cc: <http://web.resource.org/cc/> .
@prefix dc: <http://purl.org/dc/elements/1.1/> .
@prefix mo: <http://purl.org/ontology/mo/> .
@prefix ns: <http://www.w3.org/2003/06/sw-vocab-status/ns#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix smi: <http://purl.org/ontology/iomust/smi#> .
@prefix spd: <http://purl.org/ontology/studio/sigproc/> .
@prefix wot: <http://xmlns.com/wot/0.1/> .
@prefix xml: <http://www.w3.org/XML/1998/namespace> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix core: <http://purl.org/ontology/ao/core#> .
@prefix dcam: <http://purl.org/dc/dcam/> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix keys: <http://purl.org/NET/c4dm/keys.owl#> .
@prefix main: <http://purl.org/ontology/studio/main/> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix skos: <http://www.w3.org/2004/02/skos/core#> .
@prefix sosa: <http://www.w3.org/ns/sosa/> .
@prefix time: <http://www.w3.org/2006/time#> .
@prefix vann: <http://purl.org/vocab/vann/> .
@prefix event: <http://purl.org/NET/c4dm/event.owl#> .
@prefix label: <http://purl.org/net/vocab/2004/03/label#> .
@prefix mixer: <http://purl.org/ontology/studio/mixer#> .
@prefix terms: <http://purl.org/dc/terms/> .
@prefix device: <http://purl.org/ontology/studio/device/> .
@prefix iomust: <http://purl.org/ontology/iomust/internet_of_things/> .
@prefix schema: <http://schema.org/> .
@prefix studio: <http://purl.org/ontology/studio/> .
.............

###  http://purl.org/ontology/studio/mixer#Sensus_Smart_Guitar
mixer:Sensus_Smart_Guitar rdf:type owl:NamedIndividual ,
                                   iomust:SmartInstrument ,
                                   smi:SMIElectricAcousticIntrument ;
                          iomust:isInvolvedIn mixer:sensusSmartGuitarManager ;
                          smi:SMIcomponent mixer:sensusSmartGuitarAudioInputInterface ,
                                           mixer:sensusSmartGuitarGestureInterface ,
                                           mixer:sensusSmartGuitarGestureSensorHandler ,
                                           mixer:sensusSmartGuitarHapticDeliverySystem ,
                                           mixer:sensusSmartGuitarHapticEngine ,
                                           mixer:sensusSmartGuitarMappingHandler ,
                                           mixer:sensusSmartGuitarSoundEngine ,
                                           mixer:sensusSmartGuitarVisualDeliverySystem ,
                                           mixer:sensusSmartGuitarElkPi ,
                                           mixer:sensusSmartGuitarVisualEngine ;
                          device:service mixer:contentReceiveService ,
                                         mixer:contentSendService ,
                                         mixer:effectsViaGestures ,
                                         mixer:realtimeStreamingService ,
                                         mixer:recordReproduceService ,
                                         mixer:recordService ,
                                         mixer:settingsManagerService ,
                                         mixer:speakerService ;
                          sosa:hosts mixer:sensusSmartGuitarSensors ,
                                     mixer:sensusSmartGuitarDistanceSensor ,
                                     mixer:sensusSmartGuitarIMU ,
                                     mixer:sensusSmartGuitarKnobOne ,
                                     mixer:sensusSmartGuitarKnobTwo ,
                                     mixer:sensusSmartGuitarPressureSensorsBody ,
                                     mixer:sensusSmartGuitarPressureSensorsFour ,
                                     mixer:sensusSmartGuitarPressureSensorsNeck ,
                                     mixer:sensusSmartGuitarPressureSensorsOne ,
                                     mixer:sensusSmartGuitarPressureSensorsThree ,
                                     mixer:sensusSmartGuitarPressureSensorsTwo ,
                                     mixer:sensusSmartGuitarRibbonSensorBody ,
                                     mixer:sensusSmartGuitarRibbonSensorNeck ,
                                     mixer:sensusSmartGuitarElkPi ,
                                     mixer:sensusSmartGuitarSwitchButtonOne ,
                                     mixer:sensusSmartGuitarSwitchButtonTwo ;
                          rdfs:comment "developed by Elk" ;
                          rdfs:label "Sensus Smart Guitar" ;
                          rdfs:seeAlso "https://elk.audio/sensus-smart-guitar/" .
```

Listing 1. Partial description in Turtle syntax of the Sensus Smart Guitar by Elk.

TABLE I. METRICS DESCRIBING THE SMIS ONTOLOGY.

Metric	Value
Number of classes	57
Number of properties	27
- *Datatype properties*	10
- *Object properties*	
Number of individuals	0
Direct popularity	0
Inverse popularity:	
- *Ontology direct imports*	12
- *Ontology indirect imports*	10
- *Classes*	440
- *DataType Properties*	170
- *Object Properties*	454
Minimum SMI triple count	30
Maximum SMI triple count	< 400

A. The web interface

We developed a Web-based application that allows for adding user generated metadata to the triple store. For this purpose, we used the Bootstrap CSS library for the graphical part, an the JQuery JavaScript library for the dynamic handling of the components. Figure 1 shows a screenshot of the developed interface related to the selection of the main components of an SMI. The interface incorporates the descriptions of the classes as reported in the ontology comments, which are accessible by hovering on the tip-tools placed near each class.

The Web interface also allows for querying the triple-store. The user is enabled with the possibility to both coding SPARQL queries and performing the queries by enabling a pre-selected number of options.

B. Querying the database

Semantic metadata in RDF makes it possible to perform complex queries over the data using a query language such as SPARQL, under the assumption that the metadata is accurately inserted in the database. We tested the SMI Ontology and the database by using a set of queries, implemented in SPARQL. The SPARQL syntax is closely related to RDF given the fact that most queries are based on a basic graph pattern in the form of a set of triples. Nevertheless, subjects, predicates and objects may be substituted by variables. Prefix bindings are created similarly to the N3 syntax. Results of the queries are retrieved by means of matching patterns against triples in an RDF store.

The following 10 queries were used. The corresponding SPARQL listings are reported below.

1) How many smart instruments are pure electronic?
2) Which type of gesture-tracking sensors compose the Smart Mandolin?
3) How many haptic actuators compose the Smart Cajón?
4) How many smart guitars are equipped with a piezo-electric microphone?
5) Which synthesizers are used in the sound engine of the Sensus Smart Guitar?
6) How many smart mandolins are using a delay audio plugin associated with a pressure sensor in the sensor interface?
7) How many smart instruments are using the audio plugin Retrologue by the company Steinberg?
8) Which online services are available for a smart guitar and what are their purposes?
9) Which kind of equipment is connected to a smart mandolin during this concert?
10) What SMIs are controlling the smartphones used by the audience in a participatory concert?

IV. CONCLUSION

In this paper we described the development of a Web service providing metadata about SMIs using Semantic Web technologies. We presented a database populated according to a survey of the SMIs domain, which was organized following the structure of the SMI Ontology. We showed that RDF metadata collected in such database allows for the retrieval of detailed information about a given SMI or a given set of SMIs. Such details about SMIs were retrieved by querying the triple store holding information about the SMIs domain, using the SPARQL query language.

The SMI Ontology is an implementation-driven ontology that evolves during its use while developing SMIs and SMIs-based applications. As a consequence, the ontology will be growing depending on the introduction of new software and hardware components that have not already been represented in the current ontology, and that may form a SMI. Moreover, the growth of the ontology will depend on the appearance of new components around which IoMusT ecosystems are structured, such as novel Musical Things, connectivity infrastructures, or innovative applications and services. Therefore, also the structure of the proposed database will change accordingly.

For distributed applications, the REST (Representational State Transfer) architectural style is widely used. In future work we will focus on making compatible the proposed data service with REST. Future also work includes automatic generation of RDF data from external resources as well as developing software agents that utilize the database for recommendation, retrieval and search. Furthermore, in the near future, we plan to release the developed triple store as a freely accessible online resource.

Finally, it is the authors' hope that this work could contribute to the creation of the "Web of Musical Things" envisioned in [14], as an extension of the Web of Things [42] to the musical domain.

REFERENCES

[1] T. Berners-Lee, J. Hendler, and O. Lassila, "The semantic web," *Scientific american*, vol. 284, no. 5, pp. 34–43, 2001.

[2] P. Hitzler, M. Krotzsch, and S. Rudolph, *Foundations of semantic web technologies*. CRC press, 2009.

[3] O. Lassila and R. Swick, "Resource description framework (rdf) model and syntax specification," 1998. [Online]. Available: http://www.w3.org/TR/REC-rdf-syntax/

```
PREFIX smi:<http://purl.org/ontology/iomust/smi#>
SELECT COUNT(?SMIPureElectricInstrument)
WHERE
{
  ?SMIPureElectricInstrument a smi:SMIElectricInstrument
  FILTER NOT EXISTS{ {?SMIPureElectricInstrument a smi:SMIElectricAcousticIntrument} UNION
  {?SMIPureElectricInstrument a smi:SMIVirtualRealityMusicalInstrument}}
}
```

Listing 2. SPARQL query retrieving all SMIs that are purely electronic

```
PREFIX device:<http://purl.org/ontology/studio/device/>
PREFIX mixer:<http://purl.org/ontology/studio/mixer#>
PREFIX owl:<http://www.w3.org/2002/07/owl#>
SELECT DISTINCT ?sensorType
WHERE
{
  ?sensorType a owl:Class.
  ?x a ?sensorType
  FILTER EXISTS {?x device:component_of mixer:smartMandolinGestureInterface}
}
```

Listing 3. SPARQL query retrieving the type of gesture-tracking sensors compose the Smart Mandolin

```
PREFIX smi:<http://purl.org/ontology/iomust/smi#>
PREFIX sosa:<http://www.w3.org/ns/sosa/>
PREFIX rdfs:<http://www.w3.org/2000/01/rdf-schema#>
PREFIX studio:<http://purl.org/ontology/studio/device/>
PREFIX mixer:<http://purl.org/ontology/studio/mixer#>
SELECT COUNT(DISTINCT ?smartCajonHapticActuator)
WHERE
{
  ?smartCajonHapticActuator a sosa:Actuator.
  ?smartCajonHapticActuator studio:component_of ?hapticSystem.
  ?hapticSystem a smi:HapticDeliverySystem.
  mixer:smartCajon smi:SMIcomponent ?hapticSystem.
}
```

Listing 4. SPARQL query retrieving the number of haptic actuators compose the Smart Cajón

```
PREFIX smi:<http://purl.org/ontology/iomust/smi#>
PREFIX rdfs:<http://www.w3.org/2000/01/rdf-schema#>
PREFIX device:<http://purl.org/ontology/studio/device/>
PREFIX microphone:<http://purl.org/ontology/studio/microphone/>
SELECT COUNT (DISTINCT ?smartGuitar)
WHERE
{
  ?smartGuitar rdfs:label ?smartGuitarLabel; smi:SMIcomponent ?audioInterface.
  ?microphone device:component_of ?audioInterface; a microphone:PizoelectricMicrophone.
  FILTER regex (?smartGuitarLabel, "Smart Guitar", "i")
}
```

Listing 5. SPARQL query retrieving the number of smart guitars are equipped with a piezo-electric microphone.

```
PREFIX smi:<http://purl.org/ontology/iomust/smi#>
PREFIX rdfs:<http://www.w3.org/2000/01/rdf-schema#>
PREFIX studio:<http://purl.org/ontology/studio/device/>
SELECT ?synth
WHERE
{
  ?soundEngine a smi:SoundEngine;
  studio:plugin ?synth. ?synth rdfs:label ?label.
  ?strumento rdfs:label "Sensus Smart Guitar"; smi:SMIcomponent ?soundEngine.
  FILTER regex (?label, "Synth", "i")
}
```

Listing 6. SPARQL query retrieving the synthesizers used in the sound engine of the Sensus Smart Guitar

```
PREFIX smi:<http://purl.org/ontology/iomust/smi#>
PREFIX rdfs:<http://www.w3.org/2000/01/rdf-schema#>
PREFIX sosa:<http://www.w3.org/ns/sosa/>
PREFIX aufx:<https://w3id.org/aufx/ontology/1.0#>
SELECT COUNT( DISTINCT ?smartMandolin)
WHERE
{
  ?smartMandolin rdfs:label ?smlabel. ?pressureSensor a smi:PressureSensors;
      sosa:isHostedBy ?smartMandolin;
      sosa:madeObservation ?observation. ?signal
      sosa:isResultOf ?observation.
  ?transform aufx:input_signal ?signal;
      smi:implementation ?plugin.
  ?plugin rdfs:label ?pluginLabel.
  FILTER regex (?smlabel, "Smart Mandolin", "i").
  FILTER regex (?pluginLabel, "Delay", "i")
}
```

Listing 7. SPARQL query retrieving the number of smart mandolins are using a delay audio plugin associated with a pressure sensor in the sensor interface.

```
PREFIX smi:<http://purl.org/ontology/iomust/smi#>
PREFIX rdfs:<http://www.w3.org/2000/01/rdf-schema#>
PREFIX studio:<http://purl.org/ontology/studio/device/>
PREFIX mixer:<http://purl.org/ontology/studio/mixer#>
SELECT COUNT(DISTINCT ?strumento)
WHERE
{
  ?soundEngine studio:plugin mixer:retrologue; a smi:SoundEngine.
  ?strumento smi:SMIcomponent ?soundEngine.
}
```

Listing 8. SPARQL query retrieving the number of SMIs using the audio plugin Retrologue by the company Steinberg.

```
PREFIX rdfs:<http://www.w3.org/2000/01/rdf-schema#>
PREFIX device:<http://purl.org/ontology/studio/device/>
SELECT DISTINCT ?service
WHERE
{
  ?smartGuitar rdfs:label ?smartGuitarLabel; device:service ?service.
  FILTER regex (?smartGuitarLabel, "Smart Guitar", "i")
}
```

Listing 9. SPARQL query retrieving all online services available for a smart guitar and their purposes.

```
PREFIX rdfs:<http://www.w3.org/2000/01/rdf-schema#>
PREFIX iomust:<http://purl.org/ontology/iomust/internet_of_things/>
PREFIX mixer:<http://purl.org/ontology/studio/mixer#>
PREFIX mo:<http://purl.org/ontology/mo/>
PREFIX owl:<http://www.w3.org/2002/07/owl#>
SELECT DISTINCT ?type
WHERE
{
  mixer:smartMandolinAugmentedParticipatoryLivePerformance mo:instrument ?smartMandolin.
  ?connection iomust:connects ?smartMandolin,?equipment. ?equipment a ?type.
        FILTER NOT EXISTS
        {
            mixer:smartMandolinAugmentedParticipatoryLivePerformance
    mo:instrument ?equipment. ?equipment rdfs:label ?label.
    FILTER regex (?label, "Smart Mandolin", "i")
  }
  FILTER (?type != owl:NamedIndividual)
}
```

Listing 10. SPARQL query retrieving all kind of equipment connected to a smart mandolin during a given concert.

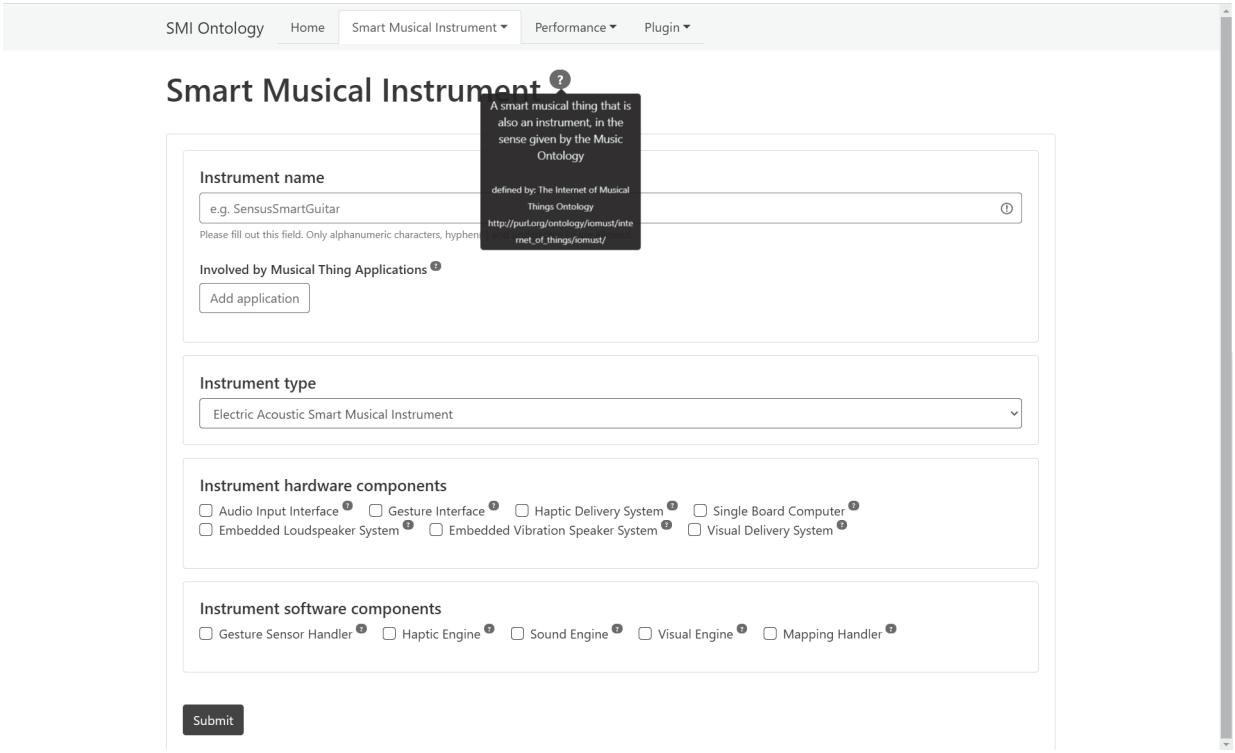

Fig. 1. Screenshot of the part of the database interface related to the selection of the main components of an SMI.

```
PREFIX rdfs:<http://www.w3.org/2000/01/rdf-schema#>
PREFIX iomust:<http://purl.org/ontology/iomust/internet_of_things/>
PREFIX mixer:<http://purl.org/ontology/studio/mixer#>
PREFIX mo:<http://purl.org/ontology/mo/>
SELECT DISTINCT ?instrument
WHERE
{
  mixer:smartMandolinAugmentedParticipatoryLivePerformance mo:instrument ?instrument.
  ?app iomust:involves ?instrument; iomust:runsOn ?smartThing.
  ?smartThing iomust:belongsTo ?person; rdfs:label ?label.
  ?person mo:listened mixer:smartMandolinAugmentedParticipatoryLivePerformance.
  FILTER regex (?label, "Smartphone", "i")
}
```

Listing 11. SPARQL query retrieving all SMIs controlling the smartphones used by the audience in a participatory concert.

[4] D. McGuinness and F. Van Harmelen, "Owl web ontology language overview," 2004. [Online]. Available: https://www.w3.org/TR/owl-features/

[5] A. Seaborne, G. Manjunath, C. Bizer, J. Breslin, S. Das, I. Davis, S. Harris, K. Idehen, O. Corby, K. Kjernsmo *et al.*, "Sparql/update: A language for updating rdf graphs," *W3c member submission*, vol. 15, 2008.

[6] E. Prud and A. Seaborne, "SPARQL query language for RDF," W3C, Tech. Rep., 2006.

[7] A. Rhayem, M. B. A. Mhiri, and F. Gargouri, "Semantic web technologies for the internet of things: Systematic literature review," *Internet of Things*, pp. 100 206:1–22, 2020.

[8] G. Fazekas and T. Wilmering, "Semantic Web and Semantic Audio Technologies," *Tutorial presented at the 132nd Convention of the Audio Engineering Society, Budapest, Hungary*, 2012.

[9] G. Fazekas, Y. Raimond, K. Jakobson, and M. Sandler, "An overview of Semantic Web activities in the OMRAS2 Project," *Journal of New Music Research special issue on Music Informatics and the OMRAS2 Project*, vol. 39, no. 4, pp. 295–311, 2011.

[10] G. Fazekas and M. Sandler, "Knowledge representation issues in audio-related metadata model design," in *Proc. of the 133rd Convention of the Audio Engineering Society, San Francisco, CA, USA*, 2012.

[11] B. De Man and J. Reiss, "A semantic approach to autonomous mixing," *Journal on the Art of Record Production (JARP)*, 2013.

[12] M. Sandler, D. De Roure, S. Benford, and K. Page, "Semantic web technology for new experiences throughout the music production-consumption chain," in *International Workshop on Multilayer Music Representation and Processing*. IEEE, 2019, pp. 49–55.

[13] D. Miorandi, S. Sicari, F. De Pellegrini, and I. Chlamtac, "Internet of things: Vision, applications and research challenges," *Ad Hoc Networks*, vol. 10, no. 7, pp. 1497–1516, 2012.

[14] L. Turchet, C. Fischione, G. Essl, D. Keller, and M. Barthet, "Internet of Musical Things: Vision and Challenges," *IEEE Access*, vol. 6, pp. 61 994–62 017, 2018. [Online]. Available: https://doi.org/10.1109/ACCESS.2018.2872625

[15] L. Turchet, G. Fazekas, M. Lagrange, H. Shokri Ghadikolaei, and C. Fischione, "The internet of audio things: state-of-the-art, vision, and challenges," *IEEE Internet of Things Journal*, 2020 (In press).

[16] L. Turchet, "Smart Musical Instruments: vision, design principles, and future directions," *IEEE Access*, vol. 7, pp. 8944–8963, 2019. [Online]. Available: https://doi.org/10.1109/ACCESS.2018.2876891

[17] L. Turchet, P. Bouquet, A. Molinari, and G. Fazekas, "The smart musi-

cal instruments ontology," *Journal of Web Semantics*, 2020 (submitted).

[18] (accessed July 2020). [Online]. Available: https://www.youtube.com/watch?v=fqzEQnsSIoY

[19] L. Turchet, M. Benincaso, and C. Fischione, "Examples of use cases with smart instruments," in *Proceedings of Audio Mostly Conference*, 2017, pp. 47:1–47:5. [Online]. Available: https://doi.org/10.1145/3123514.3123553

[20] (accessed July 2020). [Online]. Available: https://www.elk.audio

[21] (accessed July 2020). [Online]. Available: https://www.hyvibe.audio/smart-guitar/

[22] S. Benacchio, B. Chomette, A. Mamou-Mani, and F. Ollivier, "Modal proportional and derivative state active control applied to a simplified string instrument," *Journal of Vibration and Control*, vol. 22, no. 18, pp. 3877–3888, 2016.

[23] L. Turchet and M. Barthet, "An ubiquitous smart guitar system for collaborative musical practice," *Journal of New Music Research*, vol. 48, no. 4, pp. 352–365, 2019. [Online]. Available: http://dx.doi.org/10.1080/09298215.2019.1637439

[24] L. Turchet, J. Pauwels, C. Fischione, and G. Fazekas, "Cloud-smart musical instrument interactions: Querying a large music collection with a smart guitar," *ACM Transactions on the Internet of Things*, vol. 1, no. 3, pp. 1–29, 2020. [Online]. Available: https://dl.acm.org/doi/abs/10.1145/3377881

[25] L. Turchet, A. McPherson, and M. Barthet, "Co-design of a Smart Cajón," *Journal of the Audio Engineering Society*, vol. 66, no. 4, pp. 220–230, 2018. [Online]. Available: https://doi.org/10.17743/jaes.2018.0007

[26] ——, "Real-time hit classification in a Smart Cajón," *Frontiers in ICT*, vol. 5, no. 16, 2018. [Online]. Available: https://doi.org/10.3389/fict.2018.00016

[27] L. Turchet, T. West, and M. M. Wanderley, "Touching the audience: Musical Haptic Wearables for augmented and participatory live music performances," *Journal of Personal and Ubiquitous Computing*, pp. 1–21, 2020. [Online]. Available: https://doi.org/10.1007/s00779-020-01395-2

[28] L. Turchet, "Smart Mandolin: autobiographical design, implementation, use cases, and lessons learned," in *Proceedings of Audio Mostly Conference*, 2018, pp. 13:1–13:7. [Online]. Available: http://doi.acm.org/10.1145/3243274.3243280

[29] L. Turchet and M. Barthet, "Jamming with a smart mandolin and Freesound-based accompaniment," in *IEEE Conference of Open Innovations Association (FRUCT)*. IEEE, 2018, pp. 375–381.

[30] (accessed July 2020). [Online]. Available: https://elk.audio/retrologue-synth-desktop-synth/

[31] L. Turchet, S. J. Willis, G. Andersson, A. Gianelli, and M. Benincaso, "On making physical the control of audio plugins: the case of the Retrologue Hardware Synthesizer," in *Proceedings of Audio Mostly Conference*, 2020 (accepted).

[32] (accessed July 2020). [Online]. Available: https://w3id.org/smi#

[33] K. Janowicz, A. Haller, S. J. Cox, D. Le Phuoc, and M. Lefrançois, "SOSA: A lightweight ontology for sensors, observations, samples, and actuators," *Journal of Web Semantics*, 2018.

[34] Y. Raimond, S. Abdallah, M. Sandler, and F. Giasson, "The music ontology," in *Proceedings of International Society for Music Information Retrieval Conference*, 2007.

[35] T. Wilmering, G. Fazekas, and M. Sandler, "The audio effects ontology." in *Proceedings of the International Society for Music Information Retrieval conference*, 2013, pp. 215–220.

[36] ——, *AUFX-O: Novel Methods for the Representation of Audio Processing Workflows*, ser. Lecture Notes in Computer Science,. Springer, Cham, 2016, vol. 9982.

[37] G. Fazekas and M. Sandler, "The Studio Ontology Framework," in *Proceedings of the International Society for Music Information Retrieval conference*, 2011, pp. 24–28.

[38] A. Allik, G. Fazekas, and M. Sandler, "An ontology for audio features," in *Proceedings of the International Society for Music Information Retrieval Conference*, 2016, pp. 73–79.

[39] L. Turchet, F. Antoniazzi, F. Viola, F. Giunchiglia, and G. Fazekas, "The internet of musical things ontology," *Journal of Web Semantics*, vol. 60, p. 100548, 2020. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1570826820300019

[40] M. Fernández, C. Overbeeke, M. Sabou, and E. Motta, "What makes a good ontology? a case-study in fine-grained knowledge reuse," in *The Semantic Web*, A. Gómez-Pérez, Y. Yu, and Y. Ding, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 61–75.

[41] (accessed July 2020). [Online]. Available: https://jena.apache.org/documentation/tdb/

[42] D. Guinard, V. Trifa, F. Mattern, and E. Wilde, "From the internet of things to the web of things: Resource-oriented architecture and best practices," in *Architecting the Internet of things*. Springer, 2011, pp. 97–129.