Semantic Intelligence for Real-time Automated Media Production

Pieter Bonte, Femke Ongenae, Jeroen Schaballie, Dörthe Arndt, Robby Wauters, Philip Leroux, Ruben Verborgh, Rik Van de Walle, Erik Mannens, and Filip De Turck

Ghent University - iMinds, Gaston Crommenlaan 8, 9000 Ghent, Belgium
Pieter.Bonte@intec.ugent.be

Abstract Intelligent and automatic overlays for video streams know an increasing demand in broadcasting and conference systems. These overlays provide information regarding the broadcast or the conference to better engage the end users. In this paper, a platform is presented that employs Linked Data to determine the content of the overlay, based on the current context. Semantic reasoning is utilized to decide which overlays should be shown and how the cameras should be automatically controlled to capture important events.

1 Introduction

High-quality data overlays for video streams are becoming increasingly popular in broadcasting and conference systems. Broadcasting systems know a trend towards visual radio, to engage the listeners with visual data[1]. Televic, who provides conference systems for the European Parliament[2], notices an increasing demand for supporting visual facts. To enable these overlays, a thorough understanding of the current context of the conference or broadcast is mandatory. This is achieved by capturing events that describe the environment, e.g., microphone activities or mentioned keywords, and integrating them with static data and other data sources, e.g., the room configuration or information about the artist currently playing. Based on this understanding , useful information can be retrieved and shown to the end users. The information consists of interesting facts describing the activities during the broadcast or the conference, e.g., the latest songs of an artist or the agenda points of a political party.

A second requirement for automated video is the automatic manipulation of the cameras. This allows to capture important detected events during the show or the conference. Semantic reasoning is utilized to determine the importance dynamically. For example, the cameras can be triggered through microphone activity, starting of a song or a new point on the agenda. The use of Linked Data allows to link the events to useful information and enables data from different sources to be searched and queried.

In this paper, a platform is presented that employs Linked Data to intelligently determine the content of an overlay, based on the context and the events during a broadcast or conference. Semantic reasoning is employed to intelligently decide which overlays to show and when to show them. Furthermore, a tool is provided to manipulate the automated reasoning process, allowing the end-user more control over the made decisions. A demo of the developed system can be found at: http://youtu.be/pz-ITFOmZkM.

[1] http://q-music.be/

[2] http://www.televic-conference.com/en/european-parliament

Work has been done to semantically enrich existing media production datasets and use Linked Data to enable intelligent search and data retrieval [2,4]. Our approach goes a step further by incorporating activities during the conference or broadcast.

2 Overlay Data

In the broadcast case, a written preparation is provided by the broadcaster, minutes before the start of a show. This preparation is analyzed and enriched to find keywords and extract information that could be used in the video overlay. DBpedia[3] is searched to retrieve useful information regarding these keywords. The broadcaster can select which data from the enrichment is useful for the overlays. In the conference case, the Linked Data that is explored consists of profile information regarding the members of the conference, information about their parties, links to discussed documents, etc.

During the broadcast or conference, the most fitting data is selected based on the incoming events, without the need for manual selection.

3 Event-Based Reasoning Platform

To integrate the low-level data, describing the events during the show or conference, MASSIF [3] was utilized. The platform consists of multiple reasoning services, containing their own ontology and reasoner. These *Services* each perform a distinct reasoning task and share their obtained knowledge over a Semantic Communication Bus (SCB) [1]. MASSIF utilizes *Adapters* to semantically enrich the low-level raw data.

3.1 Ontology

Each case uses its own ontology. The broadcast ontology imports the Music[4] and Friend of a Friend[5] (FoaF) ontologies. In the conference ontology, FoaF and Sioc Core[6] are imported.

3.2 Adapters

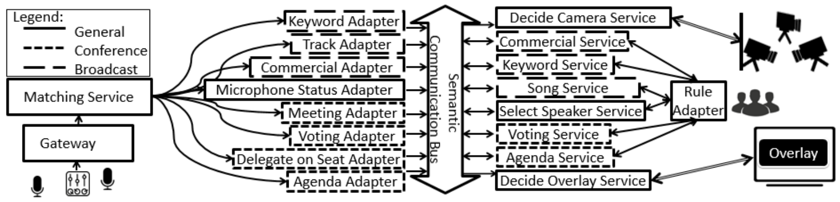

Figure 1. Mapping of the constructed *Services* & *Adapters* to MASSIF.

As shown in Figure 1, multiple *Adapters* were created to enrich data originating from various sources:

- *Microphone Status Adapter*: activation of the microphones.
- *Meeting Adapter*: the start and stop of meetings and intermediate updates.
- *Voting Adapter*: the start and stop of a voting and intermediate updates.
- *Delegate on Seat Adapter*: data describing on which seats a person is sitting.

[3] http://wiki.dbpedia.org/
[4] http://musicontology.com/
[5] http://xmlns.com/foaf/spec
[6] rdfs.org/sioc/ns

- *Agenda Adapter*: the start and stop of new agenda items.
- *Keyword Adapter*: detected keywords during the broadcast.
- *Track Adapter*: the start and stop of songs.
- *Commercial Adapter*: the start and stop of commercials.

3.3 Services

Each *Service* reasons on the integrated data by using SWRL-rules and generates a sequence of shots, based on some precondition. In the following example, rule (1) creates a *Sequence* when the microphone of the DJ is active. Rule (2) generates *Shots* to be shown in the *Sequence*. The *Shot* can show the main guest and can only be added if there is such a guest. The separation of the two types of rules allows multiple combinations in each *Service*, where multiple rules of each type can be active. When an event arrives at one of the *Services*, it uses the reasoner to create and retrieve all the instances of the type *Sequence*.

```
(1) Microphone(?m),capability(?m,DJ),unitState(?m, On) -> Sequence(Sequence_dj)
(2) Track(?t),capability(?g,MainGuest) -> member(Sequence_dj,Shot1),show(Shot1,?g)
```

The created *Services* used in both cases are elaborated below:

- The *Select Speaker Service* receives microphone activity data and can create *Sequences* when there is microphone activity for the DJ, the main guest, the chairman or some arbitrary person that is speaking. *Shots* can be created to visualize, e.g., the DJ, the main guest or the person speaking.
- The *Decide Camera Service* controls the cameras. It captures the possible *Sequences* of camera *Shots* and determines what the best suited cameras and camera positions are to show a given person in the received *Shots*.
- The *Decide Overlay Service* provides the video stream with additional information based on the activities in the studio or the conference room. The type of overlay is selected through reasoning in the *SelectSpeakerService* or one of the case-specific services elaborated below.

The *Services* created for the conference use case are:

- The *Voting Service* collects all the voting information and decides who should be shown and what overlays should be selected when a voting starts or stops.
- The *Agenda Service* selects who to show when a new agenda items starts or stops. It is able to define overlays such as item titles, linked documents, etc.

The *Services* created for the broadcast use case are:

- The *Song Service* captures information about the playing songs. It can decide to control the cameras and overlays upon the start or end of a song.
- The *Commercial Service* contains all the information regarding the played commercials and provides similar functionality as the *Song Service*.
- The *Keyword Service* receives a spoken keyword as input and determines what overlay should be shown upon detecting the keyword.

3.4 Implementation

The platform is created in OSGi[7], which allows it to be extended with additional *Adapters* or *Services* on the fly. The ontologies are internally represented using the OWL API[8]. The Pellet reasoner is used in the *Services* to reason with the created SWRL-rules.

[7] www.osgi.org

[8] owlapi.sourceforge.net

3.5 Reasoning Manipulation

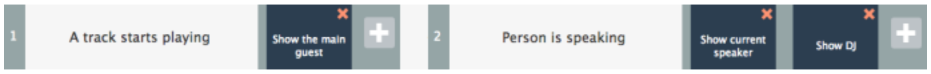

Figure 2. User interface for the adaptation of the rules by non-technical users.

To allow control over the automated video composition, a visual *Rule Adapter* is provided that allows end users to adapt the reasoning decisions in the *Services*.

The rules in each *Service* can be adapted to manipulate the automated process. The manipulation of the rules has been abstracted, eliminating the need for the end user to have specific knowledge regarding rules or the ontology. Each *Service* provides high-level generic rules which can be made more specific. The provided rules are based on the possible events during the show or conference. As shown in Figure 2, a possible rule might be the fact that a track starts playing and multiple predefined actions can be chosen for that fact. The example below shows a high-level rule with the possible event as the antecedent in (2) and the predefined actions as consequences in (5) and (7).

```
(1) {"description": "A track starts playing",
(2)  "antecedent": {
(3)    "rule": ["Track(?t), q:isActive(?t,true) -> Sequence(Sequence_Song)"]},
(4)  "consequences": [
(5)    {"subRule":"Track(?t), capability(?guest, MainGuest)
                -> member(Sequence_Song, Shot1), show(Shot1, ?guest)",
(6)     "description":"Show the main guest"},
(7)    {"subRule":"Track(?t), capability(?dj, DJ)
                -> member(Sequence_Song, Shot2), show(Shot2, ?dj)",
(8)     "description":"Show the DJ"}]}
```

The descriptions in (1), (6) and (8) show how the rules are mapped to readable sentences, alleviating the non-technical user from the technical details. The antecedent and the consequences contain a (sub)rule, which are valid SWRL-rules. When the end user selects one (or more) of the predefined consequences, the antecedent rule (3) and the selected subrules (5) or (7) are added to the ontology, allowing the reasoner to incorporate the users preferences. Note that these rules can be manipulated anytime.

4 Conclusion

We have shown the possibility to provide intelligent video overlays and camera manipulation in real-time media production through the use of Linked Data and semantic reasoning. The end user has been provided with a tool to manipulate the outcome of the reasoning process, which selects the overlays, in order to have more control over the automated decisions.

References

1. Famaey, J., et al: An ontology-driven semantic bus for autonomic communication elements. In: Lecture Notes in Comput. Sci. vol. 6473, pp. 37–50 (2010)
2. Kobilarov, G., et al: Media meets semantic web - How the bbc uses dbpedia and linked data to make connections. Lecture Notes in Computer Science 5554 LNCS, 723–737 (2009)
3. Ongenae, F., et al: Ambient-aware continuous care through semantic context dissemination. BMC medical informatics and decision making 14(1), 97 (2014)
4. Redondo-Garcia, L.J., et al: Television Meets the Web: a Multimedia Hypervideo Experience. Proceedings of the Doctoral Consortium co-located with ISWC 2013 pp. 48–55 (2013)