Visualising the Structure of 18^{th} Century Operas: A Multidisciplinary Data Science Approach

Paula Muñoz-Lago, Nicola Usula, Emilia Parada-Cabaleiro, Álvaro Torrente
Instituto Complutense de Ciencias Musicales (ICCMU)
Universidad Complutense de Madrid, Spain
pmunoz@iccmu.es, nusula@iccmu.es

***Abstract*—Data visualisation is an effective strategy to communicate information. From a multidisciplinary perspective, applying this methodology (typical of computer science) to humanistic purposes (such as visualising musical, narrative and dramaturgical structure) has shown very promising results. Opera is a complex form of art, whose structure is strongly influenced by dramaturgical and musical aspects, both present in all operatic librettos. Although visualisation methods have been successfully applied in the understanding of music and narrative individually, a combined approach, aimed to jointly visualise both aspects together—by this enhancing opera comprehension—has not yet been developed. With this in mind, through a cooperative methodology of musicology and computer science disciplines, we carry out a data science project aimed to graphically represent the structure of 18^{th} century operas' libretto structure. The presented approach, based on XML librettos from the *Progetto Metastasio*, automatically generates a comprehensive graphical representation of the opera structure, based upon musical and dramaturgical information. The schemes developed in this work show all the elements relevant to the 18^{th} century opera structure, and graphically synthesise its complex shape, in order to encourage opera understanding across a large set of users, from general listeners to musicians and finally to musicologists.**

***Index Terms*—visualisation, opera, libretto, data science, digital humanities, dramaturgy, computational musicology**

I. Introduction

Data science is an interdisciplinary computer science field [1] whose goal is to process, with adequate speed, a sufficient quantity of diverse and reliable data, in order to obtain knowledge from it through algorithms [2]. Data visualisation is an essential step in data science [3], which aims to effectively communicate information from a vast amount of data through graphical means [4]. At the intersection between computer science and humanistic disciplines lay the digital humanities, a multidisciplinary field of research which aims to apply computational methods to answer humanities-oriented questions [5]. Nevertheless, the application of data science techniques in the realm of digital humanities present a strong bias towards data extraction and processing in detriment of data interpretation and analysis [6]. Indeed, although the visualisation of humanities-related data, e. g., narrative and music, would increase notably its understanding, this topic has been almost completely overlooked.

Structure is an intrinsic aspect of narrative and music [7]. In opera, structure is primarily determined by the interconnection between both, dramatic action and musical discourse [8]. Since in this genre dramatic and musical events are analog one to the other, this interconnection allows, e. g., to perform a musical-narratological analysis [9]. Opera structure is mainly defined by the libretto [10], i. e., the dramatic text divided in acts and scenes, which is set to music and finally sung in the opera. Beyond the dramatic framework, the 18^{th} century libretto structure is strongly influenced by musical aspects, as e. g., the alternation between fully musical pieces (i. e., arias, choirs, and, duets) and more dialogic passages, close to the recitation style (i. e., recitative). In order to encourage structure understanding, data visualisation techniques have been successfully utilised in graphically representing music [11] and narrative in general [12]. Still, the application of this data science approach on complex forms of art, such as opera, in which both (music and dramaturgy) converge, is still almost unexplored [13].

In this work we automatically generate a graphical visualisation of the dramatic and musical operas' structure, by processing 18^{th} century opera's librettos codified in XML. Our approach is based upon the paradigm of the agile methodology, chosen as the most adequate to face such a multidisciplinary project—performed between computer scientists and musicologists. The presented visualisations aim to become a supporting tool thought to encourage opera understanding in a varied audience—from the general public to the expert musicians and musicologists, but considering also all other opera-related professionals. For instance, this instrument will assist users in navigating through the plot during an opera performance or while analysing the music.

The rest of the manuscript laid out as following: in Section II we outline the state of the art in this field of research; in Sections III we describe the concept of libretto as well as the codified sources; in Section IV we present our methodology; in Section V we introduce the experimental set-up; in Section VI we outline the results and discussion; finally, in Section VII, we expose the conclusions and future work.

This paper is part of the Didone Project that has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme, Grant agreement No. 788986. We would like to thank Anna Laura Bellina from the *Progetto Metastasio* for letting us use their data

978-1-7281-9134-8/20/$31.00 ©2020 IEEE
DOI 10.1109/IV51561.2020.00091

II. State of the art

Data visualisation has progressively gained popularity in a variety of research areas, including: computer graphics, image processing, and human-computer interaction [14]. This lead to the development of methods aimed to represent information graphically [15]. In order to visualise time-series data, i. e., information distributed over a time-line (such as 18^{th} century opera librettos), a popular method is the sequence chart, which represents nominal time-dependent information chronologically organised, e. g., the spiral graph [16].

Time-dependent data visualization has been used in multiple fields, ranging from software development evolution [17], to crime analysis [18]. Within the realm of digital humanities, disciplines as narratology and musicology had benefit from the application of data science approaches [6]. In particular, since data visualisation effectively summarises large amount of data [12], text visualisation techniques, such as the *Text Visualization Browser*,[1] have increased over the last years [19]. Data visualization has also been applied in representing musical features, such as timbre [20], tonality [21], or texture [22], amongst other. However, the visualisation of both musical [11] and dramaturgical [13] information together, i. e., the two structural elements of opera librettos, is mostly underrepresented [13], being still an open research question.

III. Input Data

A. Opera Librettos

For the present work we took into consideration all the opera librettos by Pietro Trapassi [23], better known as Metastasio (Rome 1698 – Vienna 1782), i. e., a poet and playwright famous for his several dramatic works, which were set to music by a high number of composers. No other texts in the history of opera received as many musical settings as the ones by Metastasio, therefore his dramatic outcome is significant, both in terms of quality and quantity [23]. Metastasio's librettos respect the structure that started to be fixed at the end of the 17^{th} century: operas tended to be organized in three acts, divided in a short number of coherent scenographic settings, and structured in single scenes depending on the entrance or exit of a character from the stage [10].

In the 18^{th} century, the opera libretto was characterized by the alternation of two main elements: recitatives and arias, both present in almost all the scenes. The former, i. e., the recitatives, constitute the texture of the opera, both dialogues and monologues, and are described as "a type of vocal writing, normally for a single voice, with the intent of mimicking dramatic speech in song" [24]. On the opposite, the latter, i. e., the arias, are passages more musically developed–as it happens with the songs in the musicals—which function is usually related to a particular emotion expressed by a single character [25]. Since arias, duets, various groupings, and choirs, together with the passages purely instrumental, are the most "musical" moments in the operas, we will graphically identify their position and their function inside the libretto. By this, we aim to give the reader a better vision of the overall context, starting from a first scheme-draft published in 2014 by Nicola Usula [26].

[1] https://textvis.lnu.se

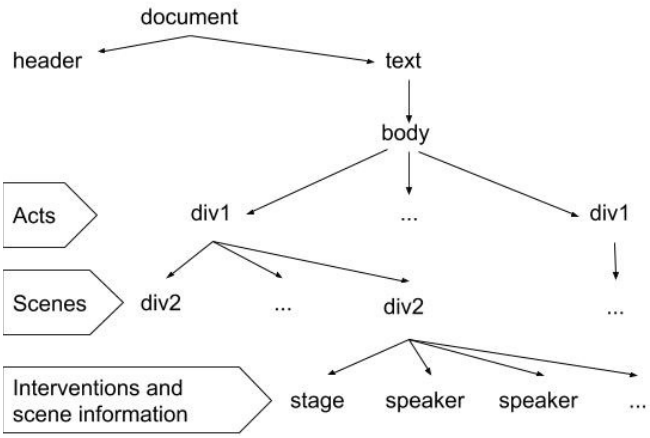

Fig. 1. XML tree structure based upon a libretto representation obtained from the *Progetto Metastasio*. The document's body contains three different sub-levels. A) div1 labels: Acts definition; B) div2 labels: Scenes contained in the Act corresponding to its parent node; C) Leaves: Information related to each speaker's intervention or scene information—note that 'speaker' in the XML libretto should be intended as singer in the opera.

B. XML Librettos

Each libretto, codified in XML format, presents a tree data structure. At the first level, a tree is composed by a root node which contains certain information and is linked to n subtrees with the same composition (i. e., every subtree has its own node which might be ramified in further levels); thus, establishing a parent-child relationship. Nodes that are not attached to other subtrees are called leaves. Figure 1 shows an ordered unranked node-labeled tree [27]—meaning that the children of a node are organised from left to right (i. e., ordered), and that the parent node does not determine the number of children (i. e., unranked). The root node at the first level, represented with the label 'document', has two children nodes: 'header' (i. e., a leave node) and 'text' (i. e., a node). 'Text' is at the same time subsequently ramified in another subtree, whose root node is 'body', which contains the libretto information. Each 'div1' label represents the root node of an act, i. e., the general tree's fourth level, while, labels such as stage or speaker are the leave nodes of the scene subtree i. e., the general tree's fifth level and whose nodes are represented with the label 'div2'.

IV. Methodology

Given the inherent challenges of approaching a multidisciplinary task [28], in order to maximise the success of an interaction between such different fields, i. e., musicology and computer science, we followed the Scrum's agile methodology [29], specially tailored to be adapted according our needs, team size, and requirements. Through the communication side-by-side with the customer (i. e., the musicologist), agile

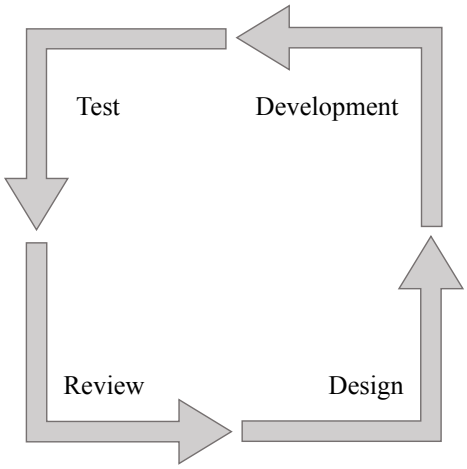

Fig. 2. Sprint's structure of the considered methodology. Each sprint is divided in four steps: design (the goal is established), development (the system is build), test (the system is evaluated), review (the output is revised). Given the cyclic interconnection between sprints, the first step (design) of a give sprint, is linked to the last step (review) of the previous one.

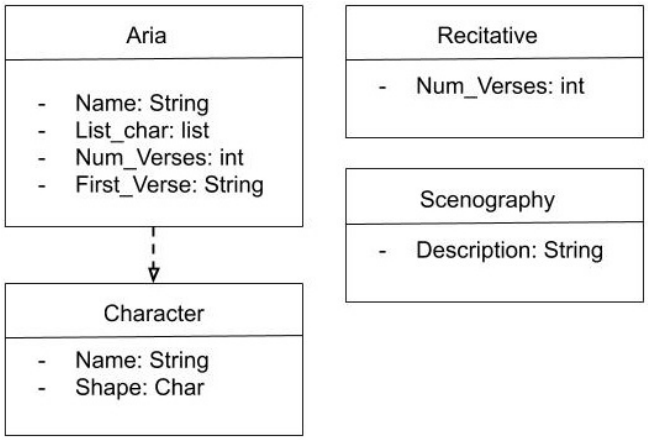

Fig. 3. UML (Unified Modeling Language) class diagram containing the four classes: Aria, Character, Recitative, and Scenography. The dotted arrow represents a relationship between Aria and Character—the second attribute of the aria (List_char) is a list of the characters singing the given aria.

methodologies increase efficiency by avoiding dispersion [30]. One of the most wide-used agile methodologies is the Scrum methodology [31], an iterative incremental approach based on two principles: (i) the *backlog*, i. e., the definition of the amount of work to be done, such as finding the most effective data structures or extracting the information; and (ii) the *sprints* [29], i. e., cycles of concrete steps in which the backlog is articulated. This incremental methodology, based on a continuous feedback, led the efforts to realistic goals and progressively improved the results' quality; thus, increasing motivation and making interactions within the multidisciplinary team smoother.

For the presented work, in a preliminary step, both, the backlog and the sprints, were collaboratively designed by the musicologist and the computer scientist. Each of the sprints was articulated in four steps: design (led by the musicologist), development and test (led by the computer scientist), and review (led by the musicologist); cf. Figure 2. A total of four iterations were considered, i. e., four repetitions of a sprint. For each iteration, a specific goal—which tried to push forward in each cycle the output of the previous iteration—was defined in conjunction between the musicologist and the computer scientist. The first sprint aimed to visualise libretto's structure by highlighting the alternation between arias and recitatives, as well as graphically indicating aria's length (represented with size, according to the number of poetic lines) and character (represented with colour). In the second sprint, colours were replaced by geometric figures, which might be used to highlight other cross features, such as the topic or the function of the aria; furthermore, concomitances between characters, i. e., duets, trios, or other groupings, were also indicated. In the third sprint, scenographic elements and recitatives' length were included. Finally, in the fourth sprint, graphical aspects were improved, e. g., different font type and indentation were introduced to distinguish between scenographic elements and the first aria verses.

V. Experimental set-up

Data extraction from the XML input files and its visualisation was performed via `python`[2] scripts. For the first task (i. e., data extraction and reading) the python library `ElementTree`[3] was used, while the second (i. e., visualization) was developed using `Matplotlib`.[4] Since structuring the data and ensuring its quality are essential steps to meaningfully extract knowledge [32], the data exploitation was preceded by a phase in which the information storage structure was defined (cf. Section V-A and Section V-B); parallel to the data exploitation, a cleaning phase was also performed (cf. Section V-C).

A. Low Level Data Structuring: Class Design

Since each low level element of a libretto, e. g., aria, recitative, character, or scenography, is a unique entity characterised by its own attributes, we propose an object oriented approach to preserve the differences between them.[5] This object oriented design is represented as an UML (Unified Modeling Language), an standard procedure in software design [33]. One of the most relevant UML representations are the class diagrams, which organises instances in classes and represents the relationships between them and their attributes.

In our UML class diagram (cf. Figure 3), the following four elements of a libretto were considered as classes: ARIA, CHARACTER, RECITATIVE, and SCENOGRAPHY. The class ARIA is constituted by four attributes: name, number of lines, first verse, and list of characters—several characters might

[2]https://www.python.org/

[3]https://docs.python.org/2/library/xml.etree.elementtree.html

[4]https://matplotlib.org/

[5]Note that scenes, acts, and opera are high level elements, thus, these were not considered within the object oriented design (cf. Section V-B).

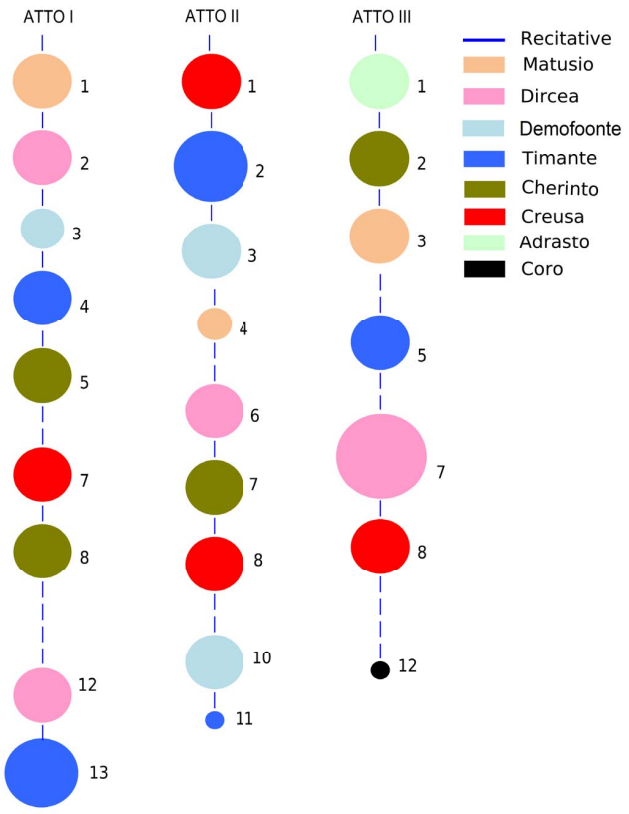

Fig. 4. First sprint result (visualisation of *Demofoonte* opera). Colours indicate the different characters; circles, i. e., arias, present a variable size according to the number of poetic lines; strokes, i. e., recitatives, have a fixed size; scenes' numbers (at the right) are also indicated.

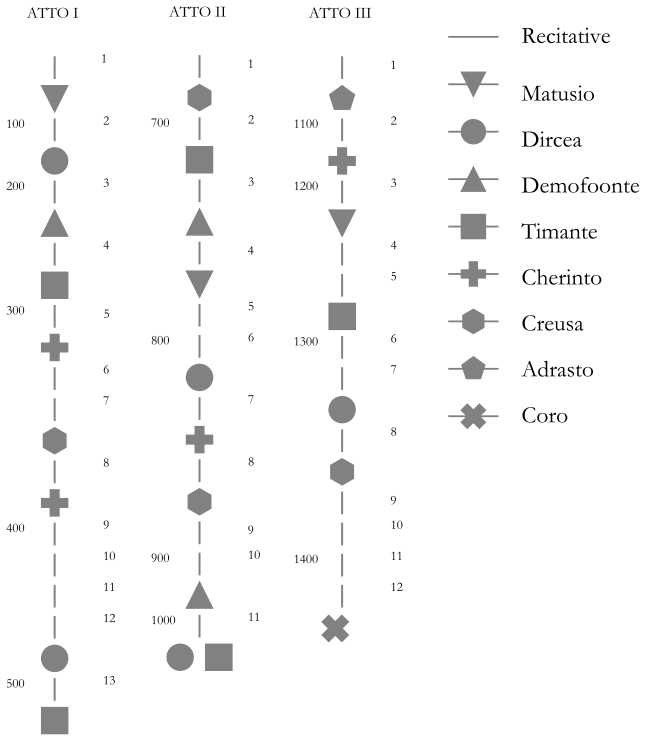

Fig. 5. Second sprint result (visualisation of *Demofoonte* opera). Figures indicate arias (a different shape is given for each character); strokes indicate recitatives; scenes' numbers (at the right) and poetic lines' numbers (at the left) are also given; concomitant singers (e. g., duet or trios) were indicated as well.

be involved, singing e. g., in duet or trio, thus, containing one or more instances of the class CHARACTER (cf. dotted arrow in Figure 3). The class CHARACTER contains two attributes: the name and the shape, i. e., the geometrical form that will be assigned in the visualisation—note that, in the first sprint, colour instead of shape was considered. The class RECITATIVE contains one attribute, i. e., the number of verses that encompass it; this was considered to highlight the musical differences with respect to the ARIA class. Finally, the class SCENOGRAPHY, integrated only in the last sprint, contains also one attribute, i. e., the stage description; this was considered to emphasise changes of scenography (i. e., mutations). The described classes and attributes, i. e., those considered in our UML model, constitute a sub-set of all the possible classes and attributes from the libretto's XML; this selection was performed by having in mind their relevance for the visualisation (cf. Section III-A).

B. High Level Data Structuring

Given the hierarchical relationships between opera, act, and scene, considered as a concatenation of events: an opera is a concatenation of acts, an act is a concatenation of scenes,[6] and a scene is a concatenation of low level elements; these, high level elements, were represented with data structures, such as dictionaries (opera, acts) or lists (scenes).

The opera is represented as a dictionary of dictionaries where act numbers are stored as keys and dictionaries as values; each of these dictionaries contains scene numbers as keys and lists of low level elements (present in that scene) as values. For instance, for an opera with three acts, the system would create a dictionary with three keys containing a dictionary in each of their values. Each of the three dictionaries would contain as many keys as the number of scenes, and for each of these the value would be a list containing scene's low level elements (i. e., aria, character, recitative, and scenography; cf. Section V-A).

C. Data Cleaning

In order to prevent system failure as well as to improve the quality of the data [34], data cleaning is an essential step aimed to avoid single-source issues, such as misspelling, redundancy, or data contradictions [34]. In this regards, the librettos' XMLs considered as input data presented a variety of 'labels' to indicate the *Chorus*, including, amongst other: *Coro*, *Parte*

[6]From the audience point of view mutations (i. e., a change of the scenography), might denote more representatively scene changes.

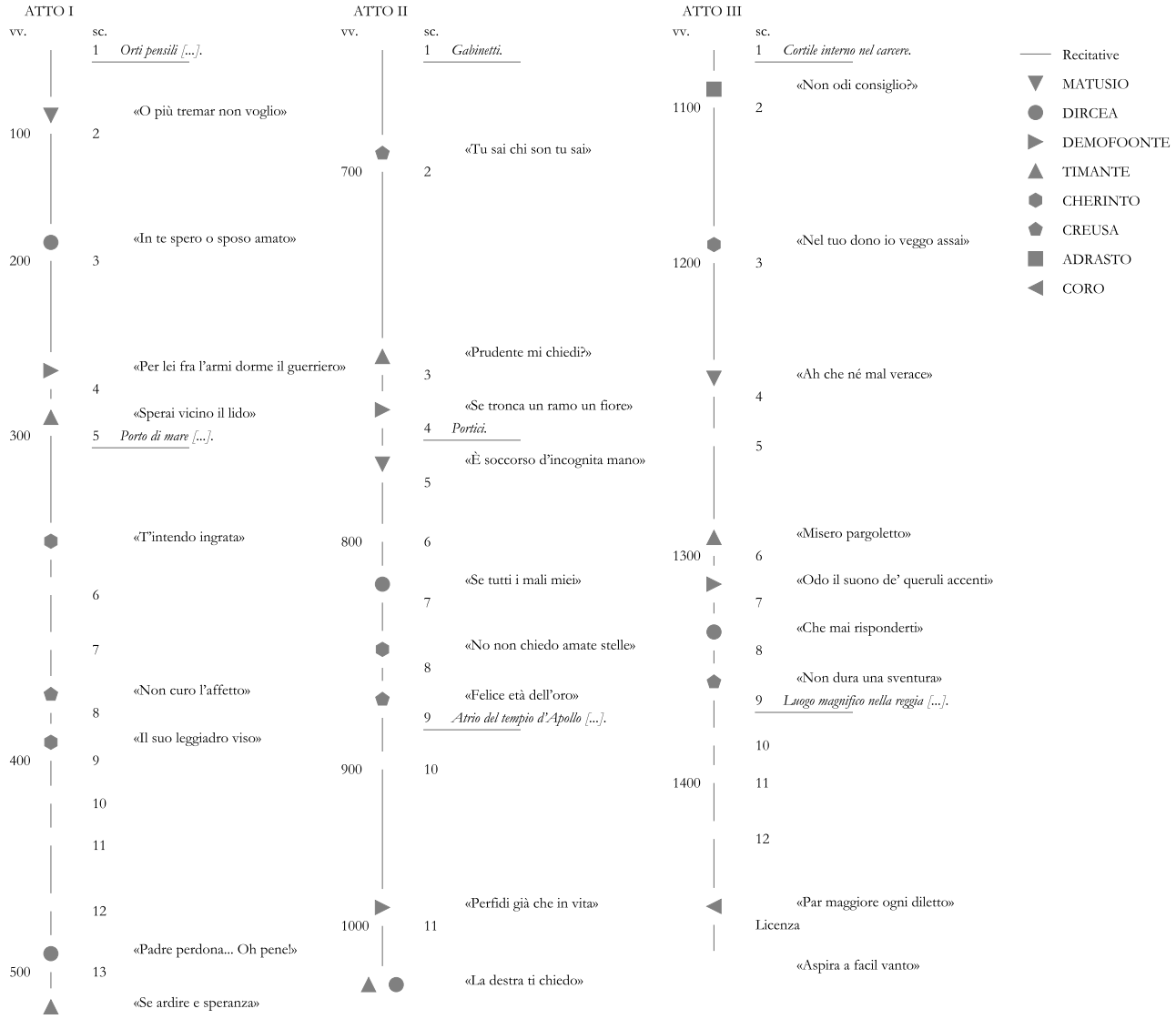

Fig. 6. Fourth sprint result (visualisation of *Demofoonte* opera). As in the previous sprint, figures indicate arias (a different shape is given for each character); strokes indicate recitatives (strokes' length according to the number of poetic lines); scenes' numbers and poetic lines' numbers (indicated at the right and at the left respectively); concomitant singers (e. g., duet or trios); and first verse of each aria (e. g., «O più tremar non voglio ») are also given. Additionally: scenographic annotations (given in italics); indentation; mutations (indicated by a horizontal line); and the *Licenza* (marked at the right) are indicated too.

del Coro, *Primo Coro*, or *Secondo Coro*. Similarly, other discrepancies in the naming convention were also found, such as the random use of articles for naming the characters (e. g., *La Gloria* and *Gloria* were interchangeably found referring to the same character). Having in mind the visualisation purposes established in the first sprint—according to which the *chorus* is considered an unique character—in order to avoid such a redundancies, all the different indications referred to this character were unified in the unique label *Chorus*. Furthermore, all the characters' names were uniformed to the form without article. Through the generalisation of the characters' names, we intend to avoid any ambiguity, thus, encouraging the structure understanding

VI. Results & Discussion

As mentioned in Section IV, for each sprint iteration a specific (incremental) goal—a graphical representation of the opera's libretto containing in every cycle more accurate and refined information—was set out. Each of the four visualisations was graphically designed in two dimensions, containing: in the x axis the acts and in the y axis the scenes and its elements. For doing this, our system automatically detected the number of acts of each opera (displayed in the x axis) and generated a vertical diagram (over the y axis), based on the acts' structure. In other words, the time line of the presented visual representations flows from the top to the bottom (i. e.,

going through scenes by displaying an alternation between arias and recitatives), and this is repeated for every column (i. e., going though the acts).

The first sprint's output is shown in Figure 4. Across the y axis the scenes' numbers, arias, and recitatives, as well as a legend, are indicated. In this sprint, the visualisation of the arias, particularly relevant from the musical point of view, was prioritised over the recitatives. Scenes' numbers, displayed on the right margin, are only associated to the arias, meaning that the exact position when the scene starts and ends is not yet indicated in this sprint. For the arias, represented with circles, both character (indicated with a specific colour) and length, i. e., number of poetic lines (indicated with the circle size) are displayed. For the recitatives, represented with fixed-length vertical lines (from now on we will refer to this as "strokes"), no indication of the character and length is given.

The second sprint's output is shown in Figure 5. Across the y axis the scenes' numbers, verses' numbers, arias, recitatives, and the legend, are indicated. Although this sprint is mostly centered in the representation of the arias too, unlike for the previous sprint, the scenes' numbers are regularly indicated in their starting position and not only according to arias, meaning that a recitative can be located in a concrete scene. Poetic lines' numbers are also regularly indicated (every 100 poetic lines) at the left margin. For the arias, the character's identification was changed from colours to shapes, since colours might be used to high-light other cross features. Arias' length (previously indicated with circle sizes) was not further marked, since incoherent—music length depends on a variety of factors, such as time signature, tempo, number of measures, or musical repetitions, which makes difficult an absolute definition of length. In addition, duets, trios, and other groupings of characters were also indicated (cf. act 2, scene 11, in Figure 5). Recitatives were indicated as in the previous sprint.

To improve the opera understanding in its context, and therefore simplifying the navigation through the opera, the third sprint's output includes, in the y axis, the scenographic notes and the first verse of each aria, at the right of the scenes' numbers. These elements were integrated in the structure already defined in the previous sprint, i. e., they were indicated along with the scenes' numbers, poetic line's numbers, arias (i. e., shapes), recitatives (i. e., strokes), and the legend. For the scenes' numbers and the poetic lines' numbers the indications 'sc.' and 'vv.' were respectively added at the top of the plot (between the act indication and the beginning of the sequence of recitatives and arias). Furthermore, recitatives length (measured according to the number of poetic lines) was also added with the stroke's length, i. e., the length of the strokes used to indicate each recitative is proportional to the recitative's number of poetic lines. In this sprint's output the final structure of the visualisation was totally defined, i. e., its appearance, unlike for minor changes, coincides with that one for the fourth sprint (cf. Figure 6).

Finally, the output of the last sprint was planned to refine the visual appearance of the plot: text size, font type, figures spacing, and the legend were optimised; indentation, mutations, and the *licenza* (i. e., closing encomiastic musical numbers, typical of the court productions [35]) were added (cf. Figure 6). Text size and font type were set to *Garamond* Python Matplotlib's size 21 and italics were used to represent the scenographic annotations. Figures spacing was optimised in order to avoid space-wasting due to the increment of information in the third spring. Figures present in the legend were indicated as in the graph, i. e., removing the default horizontal lines (cf. Figure 5). Indentation was included to enhance the differentiation between scenographic notes and the first arias' poetic lines. Mutations were indicated by adding a line under the the scenographic annotations. The *licenza*, was also marked in the right margin, aligned with the scene's numbers (cf. Figure 6).

VII. Conclusions

An effective collaboration between humanists and computer scientist is essential in digital humanities. We had shown that the use of the Scrum's agile methodology is an adequate way to efficiently handle multidisciplinary research questions, as those exposed (in particular) in the presented work, and (in general) in digital humanities. The developed system is thought to encourage opera understanding from a general perspective, i. e., considering both the musical and dramaturgical elements which relate to libretto organisation, therefore determining the opera structure. The opera's graphical visualisations, automatically generated with the presented tool, aims to assist not only musicologists in analytical process or music teachers on their lectures, but also musical amateurs and music lovers in easily following and understanding operas while enjoying them. Furthermore, the graphical visualisation might be also applied to any time-dependent sequence of events, e. g., it could be used as storytelling for musicals or movie soundtracks amongst others. Another benefit of the presented work is the opportunity to jointly contribute to already existing projects, which stimulates interest across disciplines and promotes further multidisciplinary collaborations.

One of the main limitations of our work is that it has been developed focusing on the *Progetto Metastasio* sources, thus, at its current state might not be fully compatible with other XML librettos. In this regards, one of our main goals in the near future is to optimise the system to robustly handle a variety of sources, not only in XML format but considering also other data extracting techniques, such as web scraping—practice very efficient in obtaining a large amount of sources which however should be carefully applied in order to preserve all related ownership rights. Another future goal is also to integrate the use of colours in our current black and white layout. Through the colours, in fact, the scheme will be able to highlight further features of the arias that we still did not take into consideration. The most important could be arias' topic (e. g., anger or love), arias' position (at the beginning, at the end, or in the middle of a scene), or arias' character (sung by one character alone or in company of other characters on the stage), amongst other functions that characterise arias.

References

[1] W. van der Aalst, *Process Mining: Data Science in Action*. Berlin, Germany: Springer, 2016.

[2] P. Bühlmann, P. Drineas *et al.*, *Handbook of Big Data*. Roca Raton, FL, USA: CRC Press, 2016.

[3] F. Provost and T. Fawcett, "Data science and its relationship to big data and data-driven decision making," *Big Data*, vol. 1, pp. 51–59, 2013.

[4] D. Keim, H. Qu, and K. Ma, "Big-data visualization," *IEEE Computer Graphics and Applications*, vol. 33, pp. 20–21, 2013.

[5] K. Fitzplatrick, "The humanities, done digitally," in *Debates in the Digital Humanities*, M. K. Gold, Ed. Minneapolis, MN, USA: University of Minnesota Press, 2012, pp. 12–15.

[6] E. Gardiner and R. G. Musto, *The Digital Humanities: A Primer for Students and Scholars*. New York, NY, USA: Cambridge University Press, 2015.

[7] E. Tarasti, "Music as a narrative art," in *Narrative Across Media: The Languages of Storytelling*, J. W. B. Marie-Laure Ryan, James Ruppert, Ed. Lincoln, USA: University of Nebraska Press, 2004, pp. 283–304.

[8] L. E. Zeiss, "The dramaturgy of opera," in *The Cambridge Companion to Opera Studies*, N. Till, Ed. New York, NY, USA: Cambridge University Press, 2012, pp. 179–201.

[9] C. Abbate, *Unsung Voices: Opera and Musical Narrative in the Nineteenth Century*. Princeton, NJ, USA: Princeton University Press, 1996.

[10] L. Bianconi, "Il libretto d'opera," in *Il Contributo Italiano alla Storia del Pensiero: Musica*, S. Cappelletto, Ed. Roma, Italy: Istituto della Enciclopedia Italiana, 2017, pp. 187–208.

[11] H.-H. Wu and J. P. Bello, "Audio-based music visualization for music structure analysis," in *Proceedings of Sound and Music Computing Conference*. Barcelona, Spain: SMC, 2010, pp. 1–6.

[12] E. Segel and J. Heer, "Narrative visualization: Telling stories with data," *IEEE Transactions on Visualization and Computer Graphics*, vol. 16, pp. 1139–1148, 2010.

[13] G. Scali and G. Howard, "XML coding of dramatic structure for visualization," in *Proceedings of Museums and the Web*. Washington DC, USA: MW, 2004.

[14] M. Hajirahimova and M. Ismayilova, "Big data visualization: Existing approaches and problems," *Problems of Information Technology*, vol. 9, pp. 72–83, 2018.

[15] N. Cao and W. Cui, *Introduction to Text Visualization*. Paris, France: Atlantis Publishing Corporation, 2016.

[16] M. Weber, M. Alexa, and W. Muller, "Visualizing time-series on spirals," in *Proceedings of Symposium on Information Visualization*. San Diego, CA, USA: IEEE, 2001, pp. 7–13.

[17] M. Ogawa and K.-L. Ma, "Software evolution storylines," in *Proceedings of International Symposium on Software Visualization*. Salt Lake City, UT, USA: ACM, 2010, pp. 35–42.

[18] N. Levine, "Crimestat: A spatial statistical program for the analysis of crime incidents," in *Encyclopedia of GIS*, S. Shekhar, H. Xiong, and X. Zhou, Eds. Cham, Switzerland: Springer International Publishing, 2004, pp. 381–388.

[19] K. Kucher and A. Kerren, "Text visualization techniques: Taxonomy, visual survey, and community insights," in *Proceedings of Pacific Visualization Symposium*. Hangzhou, China: IEEE, 2015, pp. 117–121.

[20] J. F. M. Cooper *et al.*, "Visualization in audio-based music information retrieval," *Computer Music Journal*, vol. 30, pp. 42–62, 2006.

[21] E. Gómez and J. Bonada, "Tonality visualization of polyphonic audio." in *Proceedings of International Computer Music Conference*. Barcelona, Spain: Citeseer, 2005, pp. 57–60.

[22] M. Giraud, F. Levé *et al.*, "Towards modeling texture in symbolic data," in *Proceedings of International Society for Music Information Retrieval Conference*. Taipei, Taiwan: ISMIR, 2014, pp. 59–64.

[23] D. Neville, "Metastasio [Trapassi], Pietro," *New Grove Music online*, accessed on February 26 2020. [Online]. Available: http://www.oxfordmusiconline.com

[24] D. E. Monson, J. Westrup, and J. Budden, "Recitative," *New Grove Music online*, accessed on February 26 2020. [Online]. Available: http://www.oxfordmusiconline.com

[25] J. Westrup, M. P. *et al.*, "Aria," *New Grove Music online*, accessed on February 26 2020. [Online]. Available: http://www.oxfordmusiconline.com

[26] N. Usula, "'Il carceriere di sé medesimo' di Alessandro Melani e Lodovico Adimari: dalla 'comedia' di Pedro Calderón de la Barca al 'Drama per Musica' di fine Seicento," PhD dissertation, University of Bologna, Department of the Arts, 2014. [Online]. Available: http://amsdottorato.unibo.it/6592/

[27] M. Lohrey, S. Maneth, and R. Mennicke, "XML tree structure compression using RePair," *Information Systems*, vol. 38, pp. 1150–1167, 2013.

[28] H. M. Cuevas, C. A. Bolstad *et al.*, "Benefits and challenges of multidisciplinary project teams: 'Lessons learned' for researchers and practitioners," *International Test and Evaluation Association Journal*, vol. 33, pp. 58–65, 2012.

[29] R. S. Kenneth, *Essential Scrum: A Practical Guide to the Most Popular Agile Process*. Upper Saddle River, NJ, USA: Addison-Wesley, 2012.

[30] M. Fowler, J. Highsmith *et al.*, "The agile manifesto," *Software Development*, vol. 9, pp. 28–35, 2001.

[31] G. S. Matharu, A. Mishra *et al.*, "Empirical study of agile software development methodologies: A comparative analysis," *ACM SIGSOFT Software Engineering Notes*, vol. 40, pp. 1–6, 2015.

[32] T. Wiktorski, *Data-intensive Systems: Principles and Fundamentals using Hadoop and Spark*. Cham, Switzerland: Springer International Publishing, 2019.

[33] D. Berardi, D. Calvanese, and G. De Giacomo, "Reasoning on uml class diagrams," *Artificial Intelligence*, vol. 168, pp. 70–118, 2005.

[34] E. Rahm and H. H. Do, "Data cleaning: Problems and current approaches," *IEEE Data Engineering Bulletin*, vol. 23, pp. 3–13, 2000.

[35] W. C. Holmes, "Licenza(i)," *New Grove Music online*, accessed on February 26 2020. [Online]. Available: http://www.oxfordmusiconline.com