Chapter

Reduction of Kernel Mechanism Processing Time in Multimedia Authoring Tool

Marvin Chandra Wijaya

Abstract

Multimedia authoring tools is a tool for designing multimedia documents based on multimedia presentation scenarios designed by the user. Visual media (images, video, animation, text, etc.) and non-visual media (sound, music, etc.) are combined into a multimedia document. A multimedia programming language represents multimedia documents. The core process in multimedia authoring tools is called the kernel mechanism, which consists of two mandatory processes: modeling and verification. Verification carried out consists of time-computation and spatiotemporal verification. The time needed for the verification process on a multimedia authoring tool is highly dependent on the object or media. The relationship between processing time and the amount of media is exponential, so if a lot of media is processed, it will take a very long time. This is the main obstacle of a multimedia authoring tool at this time. The incrementality algorithm can be applied to the verification process in order to reduce processing time to verify multimedia documents. Based on the experimental results, the processing time can be reduced by up to 75%.

Keywords: multimedia authoring tools, multimedia authoring attributes, multimedia authoring modeling, incrementality, verification

1. Introduction

Multimedia authoring tools are tools that function to combine various kinds of media into a multimedia document. The multimedia document is formed from a combination of several media according to the temporal and spatial layout [1–3]. Multimedia authoring tools must have good performance to help users produce proper multimedia documents [4]. The resulting document is represented using a multimedia language, which currently uses several multimedia languages. A multimedia authoring tool performs a kernel function in merging various independent media into a multimedia document. In general, there are three main parts in a multimedia authoring tool: the process of entering temporal data and spatial layouts, kernel mechanisms, and multimedia documents. The three parts are executed sequentially, as shown in **Figure 1**.

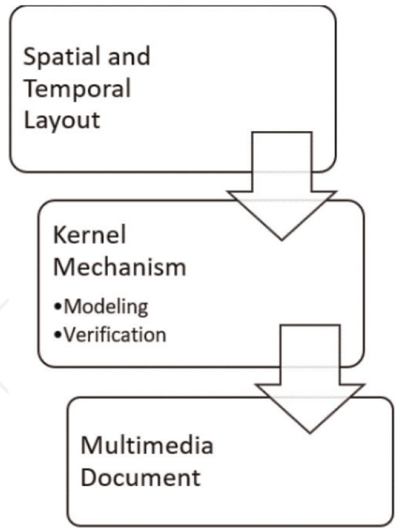

Figure 1.
Three steps in multimedia authoring tools [5].

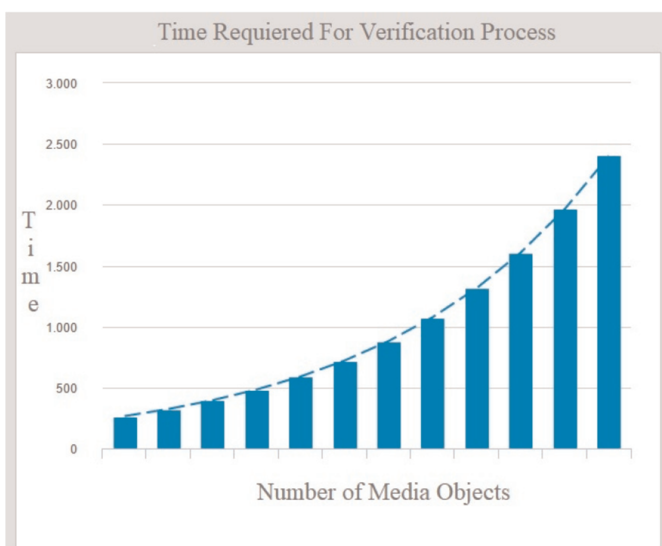

Figure 2.
Relation between kernel processing time with the number of media objects [6].

One of the problems in the process of producing multimedia documents using multimedia authoring tools is the time required for the kernel process. The kernel processing time depends on the number of media inside the multimedia document, as shown in **Figure 2** [6].

The processing time occurs because many processes occur in the kernel mechanism. Therefore, making the kernel mechanism process more efficient in terms of the runtime process is challenging.

2. Multimedia authoring attributes

A multimedia authoring tool has several attributes that inform that the multimedia authoring tool is functioning properly [7]. The attributes are service attribute, edit attribute, performance attribute, and formal model, as shown in **Figure 3**.

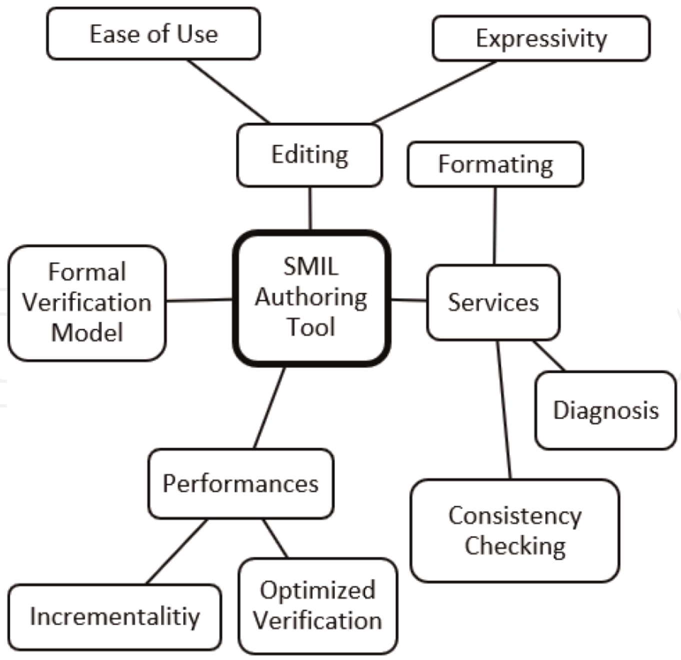

Figure 3.
Multimedia authoring attributes [8].

Attribute editing is a feature provided for the user in designing multimedia documents. There are two factors that become parameters in editing attributes: ease of use and expression. The editing process is the first part of the kernel mechanism in a multimedia authoring tool. Multimedia authoring tools have editing attributes tools including the editing process, user interface, syntax errors, error messages box, information box, and others. The more facilities in the multimedia authoring tool, the better the editing attribute value will be. Ease of use is a factor that underlies the author's ease in designing multimedia documents [9]. One of the ease-of-use factors in the multimedia authoring tool is the user interface for the authoring proses, which makes it easy to use [10]. **Figure 4** is an example of a multimedia authoring tool that uses WYSIWYG as a user interface.

The services provided by a multimedia authoring tool can be measured indirectly by investigating the following:

- Formulate and find questions or goals.
- Search for data
- Conduct a preliminary study
- Data processing
- Data output.

Figure 5 shows the hierarchy of document multimedia.

Performance attribute is a direct or indirect assessment of the performance results of the multimedia authoring tool. There are two parameters in the performance attribute: incrementality and optimized verification. Incrementality is the ability to process data by considering changes in input data that occur [11]. Software with the

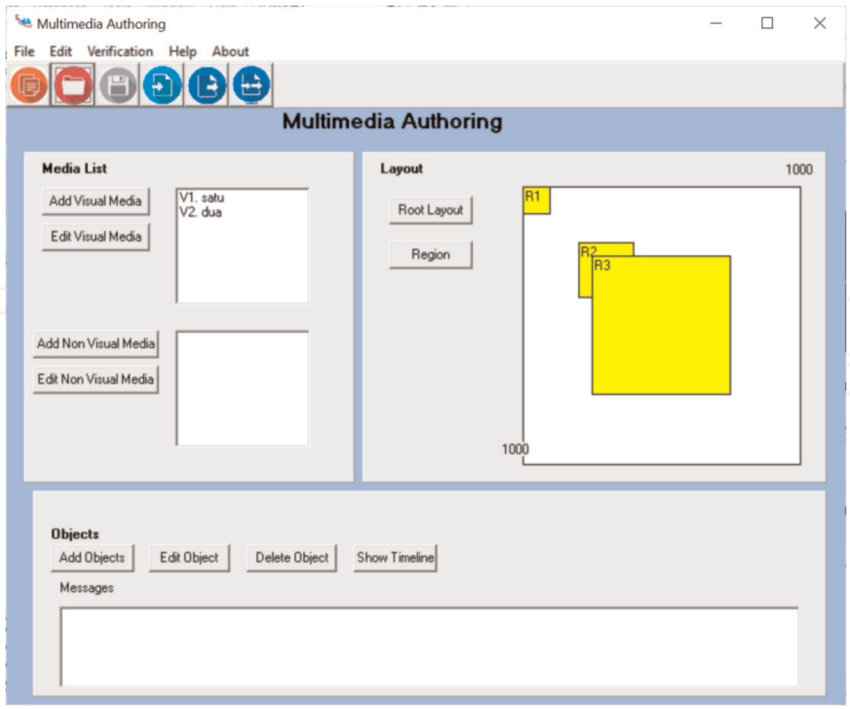

Figure 4.
Example of WYSIWYG interface on a multimedia authoring tool [5].

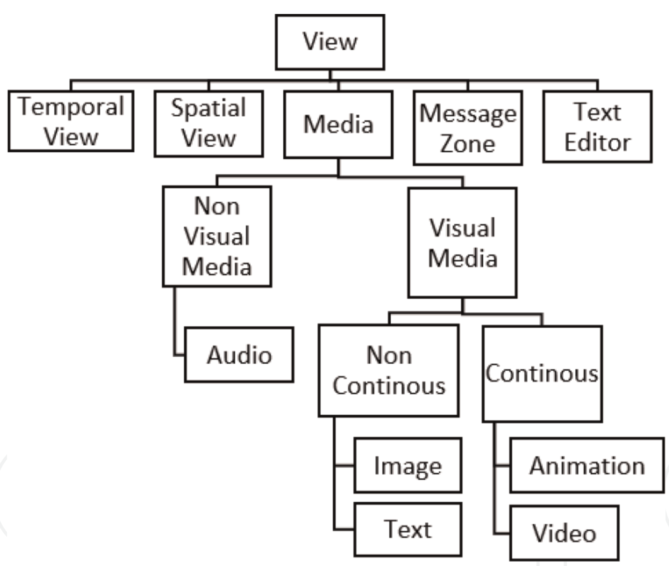

Figure 5.
Hierarchy of document multimedia [8].

incrementality concept is considering the process of recalculating based on the changed input data.

Verification is important in the multimedia authoring tool to maintain the integrity of multimedia documents. Several strategies to verify multimedia documents include analysis of editing process, time verification, spatial verification, and other validation.

Formal verification is a process to check a system that has the correct functioning of the system. Formal verification is useful to check a system in software in producing multimedia documents [12, 13].

3. Multimedia programming languages

There are several multimedia programming languages, such as synchronized multimedia integration language (SMIL), nested context language (NCL), hypertext markup language 5 (HTML 5), and others. Two programming languages that are quite widely used are SMIL and NCL.

Multimedia documents can be represented using a markup language, called SMIL (Synchronized Multimedia Integration Language) [14]. SMIL is a recommendation from the World Wide Web Consortium (www.w3c.org) for multimedia programming languages and part of XML [15, 16]. SMIL is a markup language similar to HTML and a XML variant. SMIL can control multimedia presentations, for example regarding on-screen layout and timing. The SMIL structure is defined by the module structure, consisting of <smil>, <head>, and < body> elements. The SMIL structure is as follows:

```
<smil>
     <head>
     ...meta information, layout, title, transistion effect.
     </head>
     <body>
     ... media, group, linking, animation.
     </body>
</smil>
```

Nested context language (also called as NCL) is an XML-based declarative language that represents multimedia documents. NCL has a modular structure that combines modules into a language profile [17, 18]. NCL has the ability to create multimedia presentations by creating application content on various existing devices. NCL as well as SMIL can also determine the temporal and spatial space for the media to be played. NCL has very strict separation rules between content and structure. This causes NCL not to define the media content itself, so NCL must define the glue that holds the media together and arranges them in a spatial and temporal form to become a complete multimedia document [19]. Structure of nested context language (NCL) is as follows:

```
<?xml?>
<ncl>
<head>
        <region Base>
. . . .
</region Base>
<descriptor Base>
. . . .
</descriptor Base>
<connector Base>
. . . .
</connector Base>
</head>
<body>
      ..............
</body>
</ncl>
```

4. Kernel mechanism

There are two essential processes in the kernel mechanism: the verification and modeling processes [20]. The verification process is time-consuming in a multimedia authoring tool process. Therefore, modifying the verification process and using an incrementality algorithm is necessary to improve its performance [5]. Modifications in the kernel mechanism are shown in **Figure 6**.

The verification process in the kernel mechanism is added with an incrementality algorithm to streamline its performance. The incrementality process will reduce processing time in the kernel mechanism of a multimedia authoring tool [18].

Incrementality of time calculation verification is combined with spatiotemporal verification process. This can make the verification process more efficient. The time needed in the verification process will be better. The verification process is running as usual in its original state, and there has been no change whatsoever. The incremental algorithm will be applied when there is a change or modification process. The verification process occurs for each object grouped by region in a multimedia presentation. If an error is found, the system will inform the user. Eq. (1) is used to find conflicts that occur in a region.

$$\forall Media_a, \forall Media_b, a < m, b < m, \exists End(media_a) < Begin(media_b) \quad (1)$$

Where:
$Media_a$ = Media Object a

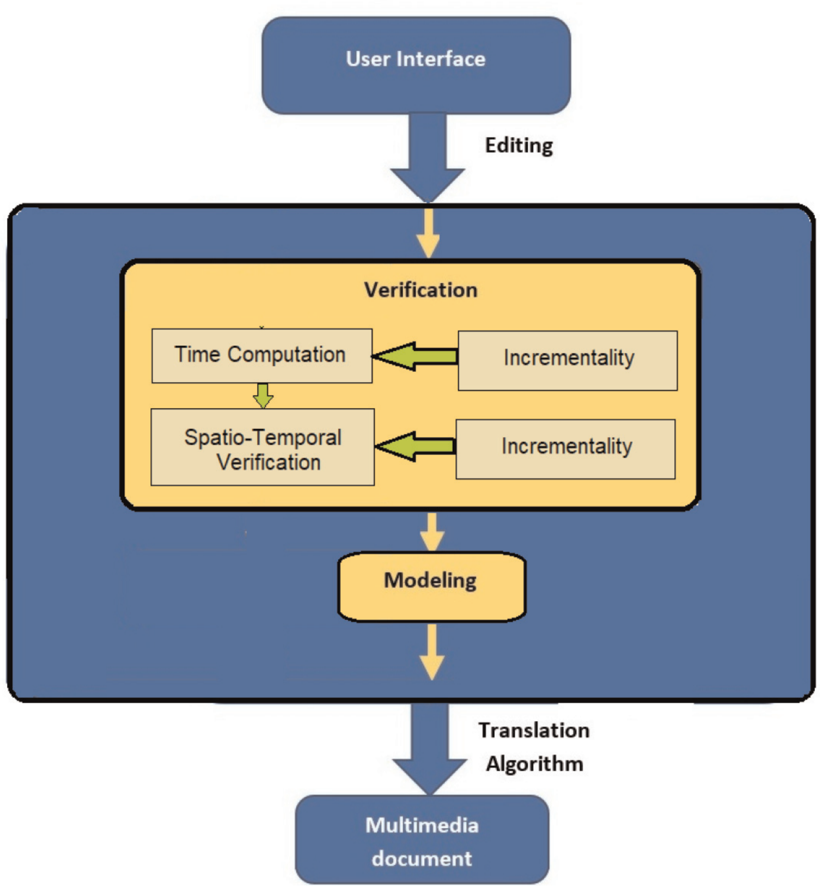

Figure 6.
Modified kernel mechanism.

DOI: http://dx.doi.org/10.5772/intechopen.110037

$Media_b$ = Media Object b
m = Number of media objects
Begin = Begin time
End = End time.

The incrementality algorithm of the time computation verification process is as follows:

---

1. Prepare each object for each area
2. Convert the timeline form from two dimensions to three dimensions.
3. The process of setting the media based on time
4. For each area {
5. If there is a change in a media {
6. For each medium {
7. IF Media_end(i) < Media_begin (i) then give message
8. End for
9. end if

Spatialtemporal verification is used to verify objects in different overlapping regions at the same time interval. Multimedia documents will be checked first to determine whether each region is overlapping: spatial and whole overlapping. Any possible overlap in each pair of regions will be included in a list. When the list is complete, all media on the list are checked for broadcast times. The verification process is applied to two overlapping media in different regions. In the incrementality of the spatiotemporal verification algorithm, verification will be applied to objects that the user has changed.

Boolean Eq. (2) is used to find two overlapping regions.

$$((Lb > La)\&\&(Lb < La + Wa)\|(Lb + Wb > La)\&\&(Lb + Wb < La + Wa)\&\&$$

$$((Tb > Ta)\&\&(Tb < Ta + Ha))\|(Ta + Ha > Ta)\&\&(Tb + Hb < Ta + Ha)) \quad (2)$$

Where:
La = left position of media a.
Lb = left position of media b.
Ta = top position of media a.
Tb = top position of media b.
Wa = width of object a.
Wb = width of object b.
Ha = height of object a.
Hb = height of object b.

The incrementality algorithm of the spatiotemporal verification process is as follows:

```
1. Prepare the substrate
2. Convert the timeline from two dimensions to three dimensions
3. Organize media by time
4. For regions
5. Process the overlapping areas
6. Ends For
7. For overlapping areas
8. If there is a change in the media
9. For main region{
10. For the next region{
11. If the start of the main region < the start of the second region then
    submit a message
12. End For
13. End For
14. End If
15. End For
```

The algorithm was tested using the SMIL language to determine its success. A multimedia presentation scenario consisted of four regional layouts and eight video media. The eight video media are displayed in a predetermined region, because the number of media is more than the number of regions, and a combined parallel and sequential scenario is designed. The scenario is designed so that all media can be displayed correctly and can be tested using the designed incrementality algorithm. The following is a multimedia document in the form of the SMIL language which is used for testing the incrementality algorithm [5].

```
<smil>.
<head>.
    <layout>.
    <root-layout   height=“110”   width=“110” />.
    <region   width=“480”   height=“480”   top=“10”   left=“10”
    id=“area1” />.
```

```
<region width="480" height="480" top="10" left="510"
id="area2" />.
<region width="480" height="480" top="510" left="10"
id="area3" />.
<region width="480" height="480" top="510" left="510"
id="area4" />.
</layout>.
</head>.
<body>.
<seq>.
<video region="area1" src="movie5.vid" id="id1" dur="160"/>.
<par dur="160 s">.
<video region="area2" src="movie6.vid" id="id2" dur="160"/>.
<video region="area3" src="movie7.vid" id="id3" dur="160"/>.
</par>.
<par dur="160 s">.
<video region="area2" src="movie8.vid" id="id4" dur="160"/>.
<seq dur="160 s">.
<video region="area3" src="movid9.vid" id="id5" dur="80"/>.
<video region="area4" src="movie10.vid" id="id6" dur="80"/>.
</seq>.
</par>.
<par dur="160 s">.
<video region="area2" src="movie11.vid" id="id7" dur="160"/>.
<video region="area3" src="movie12.vid" id="id8" dur="160"/>.
<video region="area3" src="movie13.vid" id="id9" dur="160"/>.
</par>.
</seq>.
</body>.
</smil>.
```

When the duration of the “movie5.vid” is changed from 160 to 180, the following process is

- Process: Dur1(par) = 160

  Dur2(par) = 180

- Process 2: D1 (par) = D2 (par)

- Process 3:"par” is invariant

Where:
D1 = initial duration
D2 = modification duration
The verification process will get a value of D = 160 which is regulated in the coding program, but a different value is obtained after the verification process. This causes the algorithm to declare an error. The author will also receive information on the error message in the message box on the user interface.

Number of Media	Number of Region	Number media per region	Time Required (with Incrementality)	Time Required (without Incrementality)
36	12	3	6 ms	31 ms
36	3	12	9 ms	69 ms
72	12	6	20 ms	129 ms
72	6	12	19 ms	137 ms
96	12	8	33 ms	180 ms
96	8	12	30 ms	174 ms
120	12	10	47 ms	238 ms
120	10	12	45 ms	229 ms
144	12	12	55 ms	297 ms

Table 1.
Process time comparison.

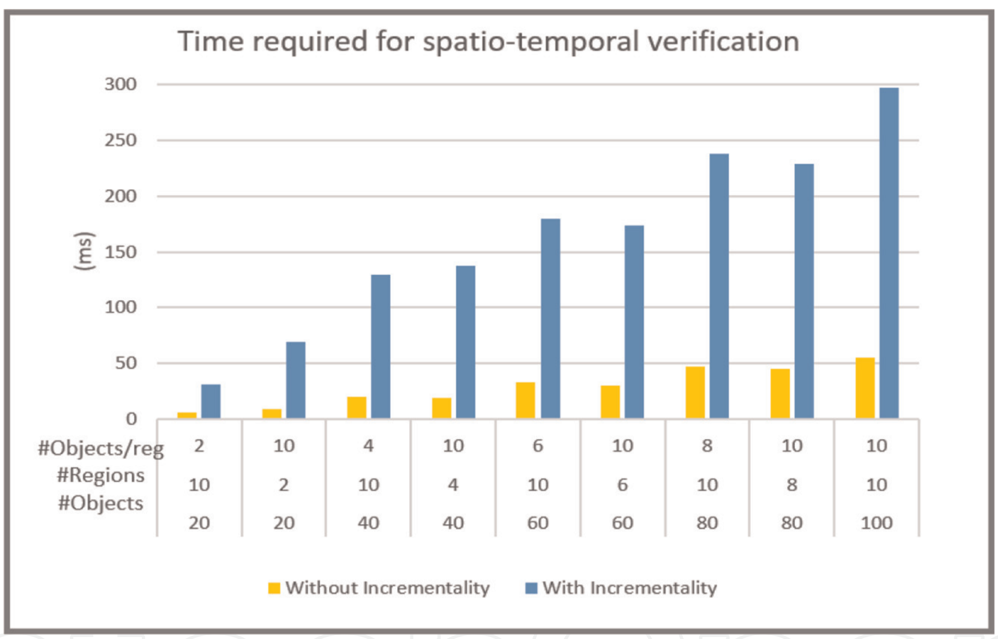

Figure 7.
Process time comparison graph.

Testing the incrementality algorithm in the verification process is done by comparing the time of the verification process. The comparison being tested is an algorithm that uses incrementality and without incrementality. **Table 1** and **Figure 7** show a comparison of the time needed to calculate the time using and without the incrementality algorithm in the verification process. Based on the experimental results, it can be seen that the incrementality algorithm has succeeded in reducing the processing time by almost 75%.

5. Conclusions

Multimedia authoring tools combine and process visual media (images, videos, animations, text, and others) and non-visual media (sound, music, and others) into a multimedia document. Currently, there are several languages for multimedia programming, such as SMIL, NCL, HTML 5, and others. The core process in the multimedia authoring tool is called the kernel mechanism, which consists of two mandatory processes, namely modeling and verification. There have been many verifications and modeling developments in multimedia authoring tools. Multimedia authoring tool has several attributes such as editing, formatting, services, formal verification model, and performance. All attributes must be met by a multimedia authoring tool to produce efficient multimedia documents, errors, and others.

Based on the experimental results, it was found that adding media to a document would result in a longer verification process, both time computation and spatiotemporal verification. The incrementality algorithm can reduce processing time to verify multimedia documents. Based on the experimental results, the processing time can be reduced by up to 75%.

Acknowledgements

I also thank Departement of Computer Engineering, Faculty of Engineering, Maranatha Christian University, Indonesia, for the support in writing this book chapter.

Conflict of interest

The authors declare no conflict of interest.

Author details

Marvin Chandra Wijaya
Maranatha Christian University, Bandung, Indonesia

*Address all correspondence to: marvin.cw@eng.maranatha.edu

References

[1] De Mattos DP, Muchaluat-Saade DC. Steve: A hypermedia authoring tool based on the simple interactive multimedia model. Proceedings of the ACM Symposium on Document Engineering. 2018;**14**(4):1-24. DOI: 10.1145/3209280.3209521

[2] Belhadad Y, Refoufi A, Roose P. Spatial reasoning about multimedia document for a profile based adaptation: Combining distances, directions and topologies. Multimedia Tools and Applications. 2018;**77**(23):30437-30474. DOI: 10.1007/s11042-018-6080-8

[3] Huang B, Yi S, Chan WT. Spatio-temporal information integration in XML. Future Generation Computer Systems. 2004;**20**(7):1157-1170. DOI: 10.1016/j.future.2003.11.005

[4] Picinin D, Farines JM, Santos CAS, Koliver C. A design-oriented method to build correct hypermedia documents. Multimedia Tools and Applications. 2018;**77**(16):21003-21032. DOI: 10.1007/s11042-017-5325-2

[5] Wijaya MC, Maksom Z, Abdullah MHL. Speedup multimedia authoring process: Incrementality of multimedia authoring verification. Journal of Engineering Science and Technology Review. 2022;**15**(3):170-176. DOI: 10.25103/jestr.153.19

[6] Wijaya MC, Maksom Z, Abdullah MHL. Novel framework for translation algorithms in multimedia authoring tools. IAENG International Journal of Computer Science. 2022; **49**(3):628-636 [Online]. Available: http://www.iaeng.org/IJCS/issues_v49/issue_3/IJCS_49_3_02.pdf

[7] Bouyakoub S, Belkhir A. SMIL builder: An incremental authoring tool for SMIL documents. ACM Transactions on Multimedia Computing, Communications, and Applications. 2011;**7**(1):1-30. DOI: 10.1145/1870121.1870123

[8] Wijaya MC, Maksom Z, Abdullah MHL. A brief of review: Multimedia authoring tool attributes. Ingénierie des Systèmes d'Information. 2021;**26**(1):1-11. DOI: 10.18280/isi.260101

[9] Pulungan FF, Sudiharto DW, Brotoharsono T. Pattern lock and the app based on context, ease of use aspect in comparison. Indones. Journal of Electrical and Engineering Informatics. 2019;**7**(1):58-66. DOI: 10.11591/ijeei.v7i1.936

[10] Yang CC, Yang YZ. SMILAuthor: An authoring system for SMIL-based multimedia presentations. Multimedia Tools and Applications. 2003;**21**(3): 243-260. DOI: 10.1023/A:1025770817293

[11] Carlsson M. Monads for incremental computing. ACM SIGPLAN Notices. 2002;**37**(9):26-35. DOI: 10.1145/583852.581482

[12] Belkhir A, Bouyakoub-Smail S. Formal design of smil documents. In: Proceeding of Third International Conference on Web Information Systems and Technologies (Webist 2007); 3-6 March 2007; Spain. 2007. pp. 396-399. DOI: 10.5220/0001271403960399

[13] Mekahlia FZ, Ghomari A, Yazid S, Djenouri D. Temporal and spatial coherence verification in SMIL documents with Hoare logic and disjunctive constraints: A hybrid formal method. Journal of Integrated Design

and Process Science. 2017;**20**(3):39-70. DOI: 10.3233/jid-2016-0020

[14] Meixner B. Hypervideos and interactive multimedia presentations. ACM Computing Surveys. 2017;**50**(1): 1-34. DOI: 10.1145/3038925

[15] Antonacci MJ, Muchaluat-Saade DC, Rodrigues RF, Soares LFG. Improving the expressiveness of XML-based hypermedia authoring languages. Multimedia Modeling: Modeling Multimedia Information and Systems, MMM. 2000;**2020**:71-88

[16] Hoschka P. The application/smil and application/smil+xml media types. RFC. 2006;**4536**:1-8. DOI: 10.17487/RFC4536

[17] Yovine S, Olivero A, Córdoba L. An approach for the verification of the temporal consistency of NCL applications. In: Proceeding of Simposio Brasileiro de Sistemas Multimídia e Web (WebMedia'10); 5-8 October 2010; Brazil. 2010. pp. 179-184

[18] Júnior DP, Farines JM, Koliver C. An approach to verify live NCL applications. In: Proceedings of the 2012 Brazilian Symposium on Multimedia and the Web (WebMedia'12); 15-18 October 2012; Brazil. 2012. pp. 223-231. DOI: 10.1145/2382636.2382685

[19] Ratamero EM. Tutorial sobre a linguagem de programação NCL (Nested Context Language). Universidade Federal Fluminense; 2007. p. 11

[20] Bouyakoub S, Belkhir A. A Spatio-temporal authoring tool for multimedia SMIL documents. International Journal of Multimedia and Technology. 2012; **2**(3):83-88