Software Engineering 2019, 8(1): 1-10
DOI: 10.5923/j.se.20190801.01

Multimedia Software Engineering Methodology: A Systematic Discipline for Developing Integrated Multimedia and Software Products

Mohanad O. Al-Jabari*, Tariq KH Tamimi, Abdul-Aziz N. Ramadan

College of Information Technology, Hebron University, Hebron, Palestine

Abstract There are strong demands for integrating various digital multimedia products such as 2D and 3D images, models, and animations in the development of software applications. Many software engineering methodologies provide a structured discipline that explicitly specifies how to develop software applications in details. However, there is a lack of a methodology like software engineering methodologies that explicitly specifies how to develop multimedia products as well as how to integrate them with existing software engineering methodologies. This research proposes a multimedia software engineering methodology. Our methodology exploits the engineering discipline of the current software engineering methodologies and integrates the phases of multimedia production with them. This methodology might be considered as a generic process for multimedia products development which provides a guideline for undergraduate students on how to develop their own graduation projects. Moreover, specialists can adapt and use it according to their business needs.

Keywords Software Engineering Methodology, Software Development, Multimedia Production, Graphics Design

1. Introduction

1.1. Background

Recently, digital multimedia products such as 2D & 3D images, characters, models and animations have been widely used in many domains such as advertising, animated movies, publications editing and design, etc. Additionally, the development of software applications has widely used multimedia products in many domains such as e-marketing, training, entertainment and games, simulation and visualization, and interactive learning [13, 15, 17, 21].

In the field of computer science, software engineering is a well-defined engineering discipline which deals with all aspects of software production from the early stage of software requirements specification to maintaining the developed software after it has deployed [1, 7, 10]. On the other hand, graphics design is an art field which deals with all aspects of multimedia production. Practically, multimedia production goes through three common stages: pre-production, production, and post-production. However, there is a lack of researches that describe these stages explicitly and in structured discipline [2, 6, 15]. Moreover, the description of multimedia production stages doesn't be considered explicitly in the software development lifecycle. Indeed, there are many software engineering researches that address multimedia products, but they mostly focus on modeling aspects and how to extend the Unified Modeling Language (UML) for supporting the modeling of multimedia products as objects inside software applications [2-5] (see Section 2).

Therefore, there is a need for interdisciplinary engineering methodology that aims to satisfy two folds: describing the production stages of multimedia products explicitly. Integrating multimedia production stages with software development lifecycle.

1.2. Research Motivation and Goal

In 2011, Hebron University adopted a new program at the faculty of information technology called "Multimedia and Web Technologies". The program aims at providing graduates with knowledge and skills that help them to master multimedia products design and development, as well as mastering design and development of web applications.

One of the most valuable courses is "graduation project" in which students have to design and develop a complete multimedia product or multimedia web application. In this context, students face a challenge when they start analyzing, designing, and developing multimedia contents as a part of their project. Actually, there is no specific methodology that is used for developing multimedia products. Therefore, students attempt to adopt classical software engineering

* Corresponding author:
mohanadj@hebron.edu (Mohanad O. Al-Jabari)

methodologies such as waterfall, incremental, and agile methodologies [1, 7, 14]. However, these methodologies are not suitable for multimedia product development, as we mentioned above [2, 6].

The aim of this research is to propose a complete, clear multimedia engineering methodology that supports all kind of multimedia products based on software engineering principles. This research is organized as follows: Section (2) discusses the state of the art related to software and multimedia engineering. Section (3) describes the characteristics of multimedia products types, and Section (4) discusses the relations between software and multimedia development life cycle. Section (5) proposes an integrated multimedia software engineering methodology based on the current software engineering and multimedia development life cycle. Section (6) discusses a number of steps to evaluate the proposed methodology. Section (7) discuss a number of open issues related to the proposed methodology. Section (8) concludes our research and discusses the future works.

2. Related Works

In the field of graphic design, there are a few researches proposals that discuss the process of multimedia production. However, there are some resources that have described the process of multimedia production. For example, Steve R. in his book [15] described three stages of multimedia production: pre-production, production, and post-production. Similarly, there are other research proposals that mostly describe the process of multimedia production as three stages, as mentioned above [16-19]. In addition to the lack of researches in this field, most resources focus on the production of time-based multimedia products (See Section 3).

In the field of software engineering, many research proposals consider multimedia products within the software development lifecycle. These researches can be categorized into three types. The first type focus on extending the diagrams of the Unified Modeling Language (UML) to support the design, development, and integration of multimedia products with interactive multimedia-based software [2-5]. The second type focuses on how to use multimedia tools and computing, visual languages, and visualization to support software engineering processes [14, 31]. The third type focuses on how to apply software engineering principles to develop multimedia products and integrate these products with software applications.

Our research focuses on the third type. In this sense, many researchers stress on the needs of an engineering discipline that guide the developments of various types of multimedia products as well as the integration of multimedia production stages with software development life cycle. For example, a survey was conducted in [11], and it aims to identify the current practice of multimedia software development methods in Ireland. To this end, the researcher surveyed 1000 software development companies and 100 companies worked in the multimedia industry. The researchers concluded that there is no systematic methodology exists to multimedia systems development. Also, the researchers concluded that developers need new techniques that help them to capture multimedia requirements and integrate them within a systems development framework. Another survey was conducted in [6]. The authors aim to identify the most widely used development models/methodologies within the multimedia industry in Austria. The research concluded that various software engineering methodologies have used by multimedia developers. However, the majority of multimedia developers did not accept these methodologies and tend to implicitly customize these methodologies. Recently, a survey was conducted in [13] on a huge number of researches that address the game software development life cycle (GSDLC). Also, this survey aims to highlight areas that need further research consideration. The survey concludes that the production phase of the GSDLC has addressed by the largest number of researches, followed by the pre-production phase. However, the post-production phase has received much less research activity than the pre-production and production phases. The results of this study suggest that the game development software engineering process has many aspects that need further attention from researchers; that especially includes the post-production phase. Other surveys were conducted in [8, 9].

To the best of our knowledge, there are a few researches that propose a structured methodology for developing different types of multimedia products. In [20], the authors have proposed a development methodology for multimedia products. This methodology encompasses six phases and each phase encompasses three activities as guidelines for developing an interactive multimedia-based education and training resources. Also, another similar multimedia software engineering methodology has proposed in [1]. However, the proposed methodologies based on software engineering perspective and do not consider the development of multimedia products from the perspectives of multimedia producer. Therefore, the proposed phases and activities are very abstract and do not consider issues related to the integration between multimedia production steps and the software development lifecycle. To the best of our knowledge, there is no research proposal that introduces a multimedia software methodology that addresses the integration of multimedia production with the software development process, as we propose in this research.

3. Multimedia Products Categories

Multimedia products can be categorized into two categories: interactive and non-interactive products. Non-interactive products can also be categorized into static products (i.e., 2D and 3D products) such as posters, logo, brochure, 3D static models, etc., and time-based products (2D and 3D animation, multimedia advertisements, etc.)

[4, 5]. Interactive multimedia products are software applications that contain multimedia products (i.e., event-driven applications such as games, Multimedia-based web applications, and interactive-based multimedia learning materials). Figure (1) below illustrates the types of multimedia products.

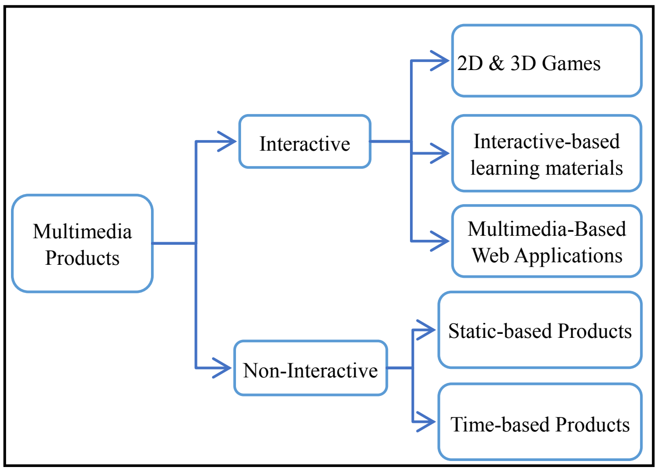

Figure 1. Types of Multimedia Products

Each of the aforementioned categories has a set of characteristics. These characteristics can be classified into three dimensions: *external view, a flow of actions, and roles of users*.

External view refers to the way that the user interacts with the product. In this sense, the users interact with non-interactive products in one way. More specifically, the users read/view static products in one frame, and they watch time-based products as multi-frames (i.e., a sequence of frames). On the other hand, users interact with interactive products in two ways. Indeed, the users usually make events and the product response to these events by making actions (ex, change its status, move to the next stage, etc.)

A flow of actions refers to the order in which product's frames are displayed to the users. In static products, there is no flow of actions, since the users read/view one frame. However, the flows of actions for time-based products are a sequence of frames (i.e., frame by frame). On the other hand, the flow of actions for the interactive products could be a sequence, selective, iterative, and event-driven [13, 15, 17, 18].

Table 1. Characteristics of Multimedia Products

Multimedia Products Types	Multimedia Characteristics		
	External View	Flow of action	Roles of Users
Static Products	One frame	No actions	Passive (Read/view)
Time-Based Products	Multi-frames	Sequence	Passive (Watch)
Interactive Products	Event-driven	Sequence, Selective, iterative, and event-driven	Active (Make Events)

Roles of users refer to what the users can do to interact with multimedia products. In non-interactive products, the users are passive where they just can read/watch products. However, users are active users with interactive products where they can make events. Table (1) summarizes types of multimedia products and their characteristics.

4. Software vs. Multimedia Development Life Cycle

In the software engineering domain, software development life cycle (SDLC) can be defined as a set of activities that leads to the production of software product [25, 27-29]. In the same sense, multimedia development life cycle (MDLC) can be defined as a set of activities that leads to the production of multimedia products [13, 15, 18, 22].

In general, the SDLC encompasses four fundamental activities: software requirement specification, design and implementation, software validation, and evolution. These activities mostly used for developing any type of software.

The MDLC for non-interactive multimedia products encompasses three fundamental activities: pre-production, production, and post-production. However, the development of interactive multimedia products integrates the aforementioned three activities with the SDLC activities, since the interactive multimedia products are software applications, as aforementioned.

This section aims to study the software development life cycle (SDLC) and the multimedia development life cycle (MDLC). Additionally, the similarities and differences between each activity included in both SDLC and MDLC will be identified. Section (5) will investigate how to adapt each phase in the SDLC to support the corresponding phase in the MDLC.

4.1. SDLC Phases

As aforementioned, there are four fundamental phases that are common to all kind of software development. These phases are illustrated in many researches and books [7, 24, 25, 26, 27, 29, 30] and can be summarized as follows:

1- Software Requirement Specification

The term requirement specification refers to the process of identifying the services that the customer requires from software and the constraints under which it operates and is developed. Requirements specification aims at answering "what" question (what customer expect? what developer is going to build? and what customer and developer are going to validate?). To do so, software analysts firstly study the software domain and identify all relevant stakeholders, eliciting each stakeholder's requirements, analyzing and validating these requirements. The output of requirement specification is a software specification document that consists of the idea of the software and a collection of functional and non-functional requirements.

2- Design and Implementation

Software design and implementation is a creative activity in which software components and their relationships are identified and implemented based on a customer's requirements and based on a set of software design rules and theories. The design and implementation process aims at answering "How" questions (how to translate customer requirement to a solution; how is the architecture of the system; how to input and how to output data; how the system look to users; how to realize the design as a program). To this end, software designers firstly select the design strategy (i.e., top down, bottom up or hybrid strategy) and design approach (e.g., Object-oriented Design approach). After that, the designers adopt a suitable software architecture which encompasses the major components that make up the software and their interactions (e.g., layered diagram, client-server diagram, etc.). Thirdly, the designers develop design diagrams (static and dynamic diagrams) to show objects and classes and relationships between these entities (i.e., class diagrams, subsystems diagrams, sequence diagrams, state-machine diagrams, etc.). Finally, the programmers develop a program by reuse existing programming code or create new programming code. The output of the design and implementation process is a software design document that encompasses all the aforementioned diagrams, and software.

3- Software Validation

Software validation (also called verification and validation (V&V)) is intended to show that a system both conforms to its specification and it meets the expectations of the system customer. Program testing, where the system is executed using simulated test data, is the principal validation technique. Validation may also involve checking processes such as inspections and reviews at each stage of the software process from user requirements definition to program development. Because of the predominance of testing, the majority of validation costs are incurred during and after implementation.

In software engineering, the term test refers to the process of checking whether the developed program does what it is intended to do and to discover program errors before it is put into use. The testing process aims at answering two questions: The first one is "does the software meets the requirements specification"; the second one is "does the software meets customer's requirements and expectations". To answer these questions, software developers verify that the building of software is right by conducting development tests (i.e., unit testing, module testing, and sub-system testing) and release test (full version test). After that, software developers and software users validate that the right software was built by conducting acceptance testing (i.e., alpha testing, and beta testing, etc.). The output of this process is verified and validated program.

4- Evolution

Software evolution refers to the process that starts by the development of the software and by any step that incrementally updates the software. Also, software evolution is very important because businesses have invested large amounts of money in their software and they consider software as business assets; so, to keep the value of these assets to the business, it must be changed and updated. The causes of software changes might come from changes in business environment and changes of user expectations which generate new requirements for the existing software, errors that are found in operation, changes in hardware and software platform, and to improve its performance or other non-functional characteristics. To do so, software engineering team firstly perform impact analysis that aims to minimize unexpected side effects from an intended change to the system. After that, if the proposed changes are accepted, a new release of the system is planned, implemented and validated, and a new version of the system is released. The output of the evolution process is a new release of the software.

4.2. MDLC Phases

As aforementioned, non-interactive products are categorized into static products and time-based products. Also, there are three fundamental activities that are common to all kind of non-interactive multimedia products. These activities are illustrated in many researches and books [13, 15, 16, 17, 23] and can be summarized as follows:

1- Pre-Production

Pre-production phase in multimedia development refers to the process of identifying the idea of the multimedia product and the constraints under it which designed and produced. Pre-production process aims at answering "what" question (what is the idea? What designer is going to design, what customer and designer are going to validate). To do so, the designer firstly identifies the idea of the multimedia product. Then, he analyzes the idea of using a set of techniques such as scenarios, storyboard, blueprints, sketches, etc. Finally, the product's idea is validated by stakeholders. The output of the pre-production phase is an abstract prototype that reflects and describe the idea of the multimedia product.

2- Production

Production is a creative activity in which multimedia product components are identified and developed based on a set of multimedia design rules and theories [12]. The production process aims at answering "How" questions (how to translate multimedia product's idea to multimedia components, how to design and develop these components, and how to integrate these components with each other's). In practice, each multimedia product categories have its own production activities.

For static products, the production process (also called static product design) consists of three activities. The first activity identifies the frame size (i.e., artboard) of the static product. In the second activity, the designer develops static product components such as logos, pictures, and text in poster using design elements such as points, lines, shapes, textures, colors, spaces, etc. Finally, the designer integrates all product components together based on design rules such as unity, balance, rhythm, etc.

For time-based products, the production process usually consists of six activities: the first activity identifies the frame size (i.e., aspect ratio) of the time-based product. In the second activity, the designer creates time-based product's components. In practice, the designer draws 2D/modeling 3D components such as 2D or 3D human character. Then, the designer textures these components (i.e., coloring, clothing, etc.). Thirdly, the designer integrates all product components together based on time-based design rules such as the safe frame, type of shots (i.e., close-up shot, medium shot, and long shot), etc. Fourthly, the designer animates the product components. In practice, the designer adds bones to the product components (Rigging) and then animates the components using these bones. In addition, the designer might add sounds to the product. Fifthly, the designer produces a complete scene. This includes preparing and tuning lights and cameras based on where and when the scene is taken (i.e., internal or external environment, day or night, etc.). (Note for digital video production.). Finally, the rendering activity is made, whereas the scene is rendered (i.e., processed) to produce the actual scene frames.

3- Post-Production

Post-production is a phase that is specific to time-based products. It refers to a set of activities that compose all actual scenes together based on a set of authentic aspects and verify that these scenes confirm to expectation. The post-production phase aims at finalizing and enhancing the final products. To do so, the designer firstly composes all scenes together (called compositing). This activity includes cutting some frames from a scene and adds video transitions between scenes. Secondly, the designer might use motion graphics techniques such as adding annotations to some scenes. Thirdly, the designer might use color correction techniques such as change color value and saturation. Finally, the designer exports the final product.

5. Multimedia Software Engineering Methodology

As we mentioned in section (3), multimedia products are categorized into interactive and non-interactive products. Additionally, each category can be seen from three perspectives: external view, a flow of actions, and roles of users.

Moreover, section (4) illustrates that there are common aspects between SDLC and MDLC phases. Indeed, the aims of software specification and pre-production phases are the same (i.e., answer "what" questions). Also, the aims of software design and development phase and production phase are the same (i.e., answer "how" questions). Thirdly, the aims of the software validation phase and post-production phase are relatively the same (i.e., validate that the product confirms customers'/stakeholders' expectations). Finally, the software evolution is dedicated to software and interactive multimedia products and aims to handle customers' change requests during product usage life cycle (i.e., after software deployment and publishing activity). Therefore, we advocate that we can integrate the phases of SDLC and MDLC based on the aims of phases. More specifically, we can integrate the software specification with pre-production phases together. Also, the design and implementation phase can be integrated with the production phase. Thirdly, the validation phase can be integrated with the post-production phase. Finally, the evolution phase is dedicated to software and interactive multimedia products, as there are no explicit evolution activities in MDLC.

In this context, the integration process can be done in two alternative ways. The first alternative is to start from SDLC as a base and integrate the MDLC activities inside the activities of the SDLC. The second alternative is to start from MDLC as a base and integrate the SDLC activities inside the activities of the MDLC.

We advocate that the first alternative is better for two reasons. First, there are many researches that discuss the activities of SDLC and there are many resources that describe these activities. In contrast, there are a few researches that discuss the MDLC activities, and there is no explicit methodology that discusses the MDLC activities, as aforementioned. Moreover, the MDLC activities focus on technical aspects to multimedia product development and don't consider explicitly other aspects such as project management and quality assurance [13, 20]. Second, multimedia products are usually part of interactive software applications. Therefore, our proposed methodology will be discussed as four phases: Requirement and Pre-Production, Design and Production, Validation and Post-Production, and Evolution. Each phase consists of a set of activities and each activity can be made using various tools and methods. Figure (2) illustrates the integration process and the remaining of this section demonstrates the integration process in more detail.

5.1. Requirements and Pre-Production

The aim of this phase is to answer what question (i.e., what customers expect, what developers are going to build, and what customers and developer are going to validate). To do so, the analysts have to perform the following activities:

Table 2. Requirements and Pre-Production Phase

Activities	Description
Product Idea Elicitation and Analysis	This activity aims to communicate with stakeholders to understand the application domain, elicit what services/ideas the product should provide, and what the constraints should be considered during the product development.
Requirements Validation	Requirements validation refers to the activity of checking that requirements actually define the product that the customer really wants. There are a number of common validation types that are used such as consistency, completeness, authentic, etc.
Project Management	Project Management is an activity that aims to manage and control product development within a specified time, at a minimum cost, and with high-quality features. Project management considers three commons aspects: feasibility study, project scheduling and planning, staff management, and risk management.

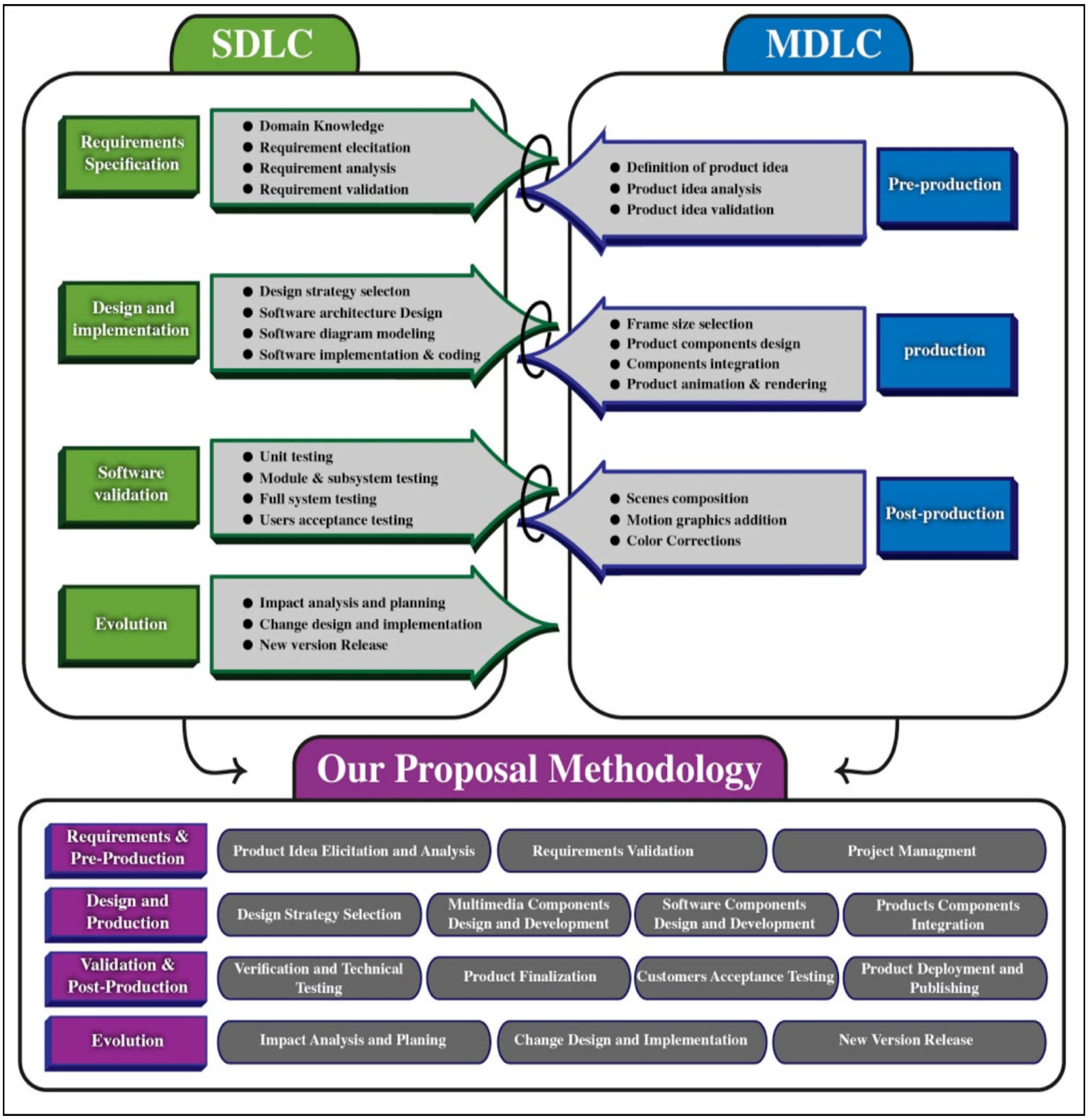
SDLC
MDLC
Requirements Specification
Domain Knowledge
Requirement elecitation
Requirement analysis
Requirement validation
Definition of product idea
Product idea analysis
Product idea validation
Pre-production
Design and implementation
Design strategy selecton
Software architecture Design
Software diagram modeling
Software implementation & coding
Frame size selection
Product components design
Components integration
Product animation & rendering
production
Software validation
Unit testing
Module & subsystem testing
Full system testing
Users acceptance testing
Scenes composition
Motion graphics addition
Color Corrections
Post-production
Evolution
Impact analysis and planning
Change design and implementation
New version Release
Our Proposal Methodology
Requirements & Pre-Production
Product Idea Elicitation and Analysis
Requirements Validation
Project Managment
Design and Production
Design Strategy Selection
Multimedia Components Design and Development
Software Components Design and Development
Products Components Integration
Validation & Post-Production
Verification and Technical Testing
Product Finalization
Customers Acceptance Testing
Product Deployment and Publishing
Evolution
Impact Analysis and Planing
Change Design and Implementation
New Version Release

Figure 2. Multimedia Software Engineering Methodology

5.2. Design and Production

The aim of this phase is to answer how question (i.e., how to transform product idea and specification to actual product based on a set of design and production rules. To do so, the designer/producer performs the following activities:

Table 3. Design and Production Phase

Activities	Description
Design Strategy Selection	This activity aims to identify how the product and its components will be developed. Design strategy might be top-down whereas a general overview of a product is formulated. After that, product's components are developed and added cumulatively. Other design strategies might be button-up, mixed, components reuse, etc.
Multimedia Components Design and Development	This is a creative activity that aims to design and develop multimedia components based on a set of multimedia design rules and theories. There are a number of common multimedia design stages such as components modeling, texturing, rigging, etc. [30]
Software Components Design and Development	This is a creative activity that aims to identify, design, and develop software components based on customers' requirements and based on a set of software design rules and theories [7]. There are a number of common design steps such as architectural design, static and dynamic design, interface design, etc.
Product's Components Integration	This activity aims to integrate and configure the developed components together so as the intended product is composited. There are a number of common steps for all products types such as unit and component integration. Other extra steps are dedicated to time-based and interactive-based multimedia products such as light and camera configuration, rendering, etc.

5.3. Validation and Post-Production

The aim of this phase is to finalize and enhance the final product and verify that it both conforms to its specification and expectations. To do so, the validator performs the following activities:

Table 4. Validation and Post-Production phase

Activities	Description
Verification and Technical Testing	This activity aims to show that the developed product conforms to its specifications. There is a set of technical testing that is performed such as unit and component testing, components integration testing, etc.
Product Finalization	This activity aims to composite, integrate, and configure all components together to produce the final product. There are a set of common steps to perform this activity such as compositing, motion graphics, color correction, exporting and packaging, etc.
Customer's Acceptance Testing	This activity aims to validate that the developed product meets the customers' expectations. There are a number of validation steps such as functionality testing, usability, and performance testing, look and feel testing, etc.
Product Deployment & Publishing	This activity aims to make the final product ready to use by its customers in its real environment. There are a number of steps to perform this activity such as stakeholder communication (i.e., interactions between producers and customers), product installation, configuration, publishing, etc.

5.4. Evolution

The aim of this activity is to consider customers' change requests after the product deployment and publishing activity is performed (i.e., during product usage in its real environment). In practice, the evolution process is clearly identified in the software engineering field, while it is not explicitly identified in the multimedia field. To do so, the team have to perform the following activities:

Table 5. Evolution Phase

Activities	Description
Impact Analysis and Planning	In this activity, the team analyze the impact of the requested change on the product's environment and decide to accept or reject the request. In practice, change request might be rejected for several reasons such as inapplicable change, the cost of change is very high, the risk of change is very high, etc. If the change request is accepted, the team manager set up a plan for conducting change. The plan considers the scheduling and management of the other change activities. (i.e., this activity is similar to the project management activity that is described in the requirement and pre-production phase)
Change Design and Implementation	This activity is similar to the activities of the design and production phase that are described above to consider the requested change.
New Version Release	This activity is similar to the activities of validation and post-product phase that are described above.

6. Methodology Evaluation

In order to justify the validity of our proposed methodology, a detailed evaluation plan is considered and partially undertaken. This can be divided into three evaluation steps. In the first evaluation step, our methodology is compared and evaluated with the ISO/IEC/IEEE 12207:2017 [32]. Theoretically, the aforementioned standard addresses the software development life cycle and describe it as four processes: agreement, organizational project enabling, technical management and technical processes. The agreement, technical management and part of technical processes (i.e., requirement definition) are covered in our proposed methodology by the requirement and preproduction phase. Also, most of the technical process is covered by two phases in the proposed methodology: design and production, and validation and post-production phases. Finally, the organizational project enabling process is covered by the evolution phase.

In the second evaluation step, our methodology has been peer reviewed by a number of experts that works in developing multimedia products as well as software applications in Palestine. This review show that our methodology considers the weaknesses of both SDLC and MDLC.

Table 6. Shows the weaknesses that are considered in our methodology

SDLC weakness	MDLC weaknesses
• Do not consider the art discipline (i.e., design rules and principles) related to multimedia production • The implementation and integration activities of multimedia production with software application are not explicitly considered.	• Do not consider technical management aspects (e.g., project management, risk management, quality assurance, etc.) • Documentation of multimedia production is not explicitly considered and not part of process development deliverables. • The evolution and maintenance of multimedia products is not considered.

Additionally, this review has identified a number of changes that have to be made on the proposed methodology. For example, each phase need to be clearly identify the methods and tools that will be used to accomplish the aforementioned activities. In addition, the tools and methods that are used to accomplish activities might be differ based on the type of the product to be developed. For example, the product idea elicitation and analysis activity for multimedia product is usually made using storyboard, blueprint, and sketches. On the other hand, the product idea elicitation and analysis activity for software product is usually made using interview, observation, and use case description.

In the third evaluation step, plan is identified to piloting our methodology among different types of multidisciplinary software and multimedia students' projects at Hebron University. Also, we plan to distribute our methodology to a number of multimedia and software specialists that work in this field. At the end of this pilot study, feedback from students as well as specialists will be collected and analysed and then our methodology will be updated accordingly.

7. Discussion and Further Aspects

In Section (5), we discuss our proposed multimedia software engineering methodology. The aim to this methodology is to provide a systematic discipline for developing software as well as multimedia products. Our methodology integrates the phases of MDLC with SDLC phases based on their aims.

The arrangement of activities of the integrated phases might be seen as simple as arranging Lego items to produce a game. However, different aspects should be considered in the arrangement of activities.

The *similarity* is one aspect whereas two activities from the integrated phases are relatively the same. For example, software specification consists of an activity called "study software domain" and the pre-production consists of an activity called "identify the multimedia product idea". These two activities are integrated together (i.e., Product Idea elicitation and analysis) and they relatively have the same purpose. However, the tools and method for performing the integrated activities might be different. For example, sketches are used to illustrate the idea of static-based multimedia products, while storyboard is used in time-based and interactive multimedia products.

Dependability is another aspect whereas the dependability between different activities should be identified and considered. For example, the multimedia designer must develop multimedia components before the programmer starts writing program code. In practice, dependability can be classified as sequential, parallel, and interleaved (i.e., two activities could be made in sequential, parallel, or interleaved fashion). For example, project management activity is usually performed in parallel with all other activities.

Terminology issue is the third aspect, whereas multimedia and software are different fields and people who working in these fields have a different meaning for the same concepts. For example, design in multimedia field refers to the use of design elements in developing a multimedia product based on multimedia design principles. In contrast, the term design in software field refers to a set of diagrams that describe the software structure and components. In the same sense, the terms scenario, modeling, materials, production, etc. have a different meaning in multimedia and software fields.

Finally, the *output* of each phase is another aspect and it depends on product type. For example, in SDLC the output of software specification is a requirement specification document that describes the functional and non-functional requirements. On the other hand, the output of the pre-production phase in the MDLC is a document that describes the product idea (i.e., logo, character specification, etc.).

8. Conclusions and Future Works

The use of multimedia products in software development becomes essential, especially after the emerge of interactive software applications such as Web applications, mobile applications, and interactive multimedia games. This creates a need for a new methodology that support the development of interactive multimedia products.

In this paper, we have proposed a multimedia software engineering methodology that exploits the engineering discipline of the SDLC and integrates the phases of SDLC with MDLC. This methodology might be considered as a generic process for multimedia products development which guides undergraduate students how to develop their own graduation projects. Moreover, specialists can adapt and use them according to their product types and business needs.

As future works, the evaluation steps that are discussed in Section (6) and aspects that are discussed in Section (7) have to be addressed. Also, there is a need for investigating which process model is suitable for applying our proposed methodology (i.e., waterfall model, rational unified process model, agile model, etc.) [26, 27].

REFERENCES

[1] Y. Zhang. The Research on Multimedia Software Engineering Based on Software Life Cycle. In proceeding in the 9th International Conference on Computer Science and Education (ICCSE), Canada, pp. 241-244, 2014.

[2] A. PleuB. Modeling the User Interface of Multimedia Applications. In proceeding in model-driven Engineering Languages and Systems (MoDELS), Lecture Notes in Computer Science, vol 3713. Springer, Berlin, Heidelberg, pp 676-690, 2005.

[3] S-K. Chang. Multimedia Software Engineering. In proceeding in the International Series in Software Engineering, Springer; 2000.

[4] G. Engels, S. Sauer. Object-oriented Modeling of Multimedia Applications. In proceeding in the Handbook of Software Engineering and Knowledge Engineering, vol. 2, pp. 21-53, 2002.

[5] S. Sauer and G. Engels. UML-based Behavior specification of Interactive Multimedia Applications. In Proceeding in HCC '01 Proceedings of the IEEE 2001 Symposia on Human-Centric Computing Languages and Environments, Italy, pp 248 – 255, 2001.

[6] M. szota and K. Ellis. Methodologies for Developing Multimedia Systems: A Survey. In Proceeding in the Emerging Trends and Challenges in Information Technology Management book, Volume 1 & Volume 2, 2006.

[7] I. Sommerville. Software Engineering textbook. 9th edition, 2011.

[8] M. Lang, C. Barry. Techniques and Methodologies for Multimedia Systems Development: A Survey of Industrial Practice. In Russo, N. L. et al. (eds), Realigning Research and Practice in Information Systems Development, Proceedings of IFIP WG 8.2 Conference, USA, pp. 77-86, 2001.

[9] M. Amor, L. Fuentes, M. Pinto. A Survey of Multimedia Software Engineering. Journal of Universal Computer Science, vol. 10, no. 4, pp. 473- 498, 2004.

[10] S-K Chang. A Framework for Multimedia Software Engineering. Book chapter Multimedia Software Engineering, in the proceeding of the International Series in Software Engineering, Springer; pp. 1 – 10, 2000.

[11] C. Barry and M. Lang. A survey of multimedia and Web development techniques and methodology usage. In IEEE MultiMedia, vol. 8, no. 2, pp. 52-60, 2001.

[12] A. W. White. The elements of graphics Design, 2nd Edition, Allworth Press, 2011, ISBN: 9781581157628.

[13] S. Aleem, L. F. Capretz, F. Ahmed. Game development software engineering process life cycle: a systematic review. International Journal of Software Engineering Research and Development, Vol. 4 Issue 6, pp. 1 – 30, 2016.

[14] F. G. Ribeiro, C. E. Pereira, A. Rettberg, M. S. Soares. Model-based requirements specification of real-time systems with UML, SysML, and MARTE. Journal of Software and Systems Modeling (SoSyM), Vol. 17, issue 1, p.p. 343-361, 2018.

[15] R. Steve. Pre-Production Planning for Video, Film, and Multimedia. Focal Press Publisher, USA, p.p 14 – 21, 1996. ISBN 0-240-80271-3.

[16] G. Engels, S. Sauer, B. Neu. Integrating software engineering and user-centered design for multimedia software developments, Human-Centric Computing Languages and Environments. Proceedings in IEEE Symposium, pp. 254-256, 2003.

[17] B. G. Dan, B. Curless, D. Salesin, M. S. Steven. Schematic Storyboarding for Video Visualization and Editing. Journal of ACM Transactions on Graphics, Volume 25, Issue 3, pp. 862-871, 2006.

[18] R. Ramadan, Y. Widyani. Game development lifecycle guidelines, Proceeding in International Conference on Advanced Computer Science and Information Systems (ICACSIS), Bali, pp. 95-100, 2013.

[19] L. Min, C. Jones, S. Hemstreet. Interactive Multimedia Design and Production Processes. Journal of Research on Computing in Education. Volume 30, Number 3, pp. 254-280. 1998 ISSN 0888-6504.

[20] C. Sherwood, T. Rout. A Structured Methodology for Multimedia Product and Systems Development. In proceeding in proceedings of the 15th Annual Conference of the Australasian Society for Computers in Learning in Tertiary Education (Ascilite ’98), Wollongong, pp. 617-625, 1998.

[21] A. F. Barbara, J. M. Sutton. A Model for Developing Multimedia Learning Projects (MERLOT). Journal of Online Learning and Teaching, Vol. 6, No. 2, pp. 491 - 507, 2010.

[22] M. Hirakawa. Do software engineers like multimedia? In *Proceedings of IEEE International Conference on Multimedia Computing and Systems*, Italy, pp. 85-90 vol.1, 1999.

[23] R. E. Beasley. Interactive Multimedia Development: Pre-Design Analyses. Journal of Educational Technology Systems, vol. 27 issue 1, pp. 23-42, 1999.

[24] K. S. Babu, R. Maruthi. Lifecycle for Game Development to Ensure Enhanced Productivity. International Journal of Innovative Research in Computer and Communication Engineering. Vol. 1, Issue 8, pp. 1490 – 1503, 2013.

[25] N. M. Munassar, A. Govardhan. A Comparison between Five Models of Software Engineering. International Journal of Computer Science Issues, Vol. 7, Issue 5, pp. 94 - 101, 2010, ISSN: 1694-0814.

[26] A. Y. Egwoh, O. F. Nonyelum. A Software System Development Life Cycle Model for Improved Students’ Communication and Collaboration. International Journal of Computer Science & Engineering Survey (IJCSES), Vol. 8, No. 4, pp. 20 - 41, 2017.

[27] PK. Ragunath, S. Velmourougan, P. Davachelvan, S. Kayalvizhi, R. Ravimohan. Evolving A New Model (SDLC Model-2010) For Software Development Life Cycle (SDLC). International Journal of Computer Science and Network Security (IJCSNS), VOL. 10 No. 1, pp. 112 – 119, 2010.

[28] C. Laura, H. A Rodriguez-Martinez, R. Duran-Limon, G. Mendoza, M. Jaime. Identifying Common Activities in the Graphical User Interface Development Process and their integration into the Software-System Development Life Cycle. Journal of Computer Science and Information Systems. Vol. 12, No. 1, pp. 323–348, 2015.

[29] A. Dennis, B. H. Wixom, D. Tegarden. Systems Analysis and Design: An Object-Oriented Approach with UML, 5th edition, Wiley Publishing, 2015, ISBN:1118804678 9781118804674.

[30] T. Vaughan. Multimedia-making-it-work. 8th edition, McGraw-Hill publishing, 2011, ISBN: 978-0-07-174850-6.

[31] S. King. Tool support for systems emergence: A multimedia CASE tool. Journal of Information and Software Technology. Vol. 39, Issue 5, p.p 323-330, 1997.

[32] ISO/IEC/IEEE 12207, Systems and software engineering - Software life cycle processes, Geneva, Switzerland: International Organization for Standardization, 2017, DOI: 10.1109/IEEESTD.2017.8100771.