



UNIVERSITY OF CAPE TOWN

DEPARTMENT OF COMPUTER SCIENCE



CS/IT Honours Final Paper 2019

Title: Salsational: An application for dance sequence generation

Author: Alka Baijnath

Project Abbreviation: DeDance

Supervisor(s): Associate Professor Maria Keet

Category	Min	Max	Chosen
Requirement Analysis and Design	0	20	13
Theoretical Analysis	0	25	9
Experiment Design and Execution	0	20	6
System Development and Implementation	0	20	12
Results, Findings and Conclusion	10	20	10
Aim Formulation and Background Work	10	15	10
Quality of Paper Writing and Presentation	10		10
Quality of Deliverables	10		10
<u>Overall General Project Evaluation</u> (<i>this section allowed only with motivation letter from supervisor</i>)	0	10	
Total marks		80	

Salsational: An application for dance sequence generation

Alka Baijnath
alkabaj@gmail.com
University of Cape Town
Cape Town, South Africa

ABSTRACT

Evolution Dance Company is Cape Town's first community-based Salsa Dance Company with a primary focus on the development of Salsa dance. In order to accomplish their mission, a software tool was proposed to supplement 'in-class' learning for beginner dancers, by providing functionality for dance students to create and practice dance sequences and for dance teachers to plan dance lessons. A dance notation is used to document four-dimensional dance choreography into two-dimensional space. Thus, in order to utilize dance moves in the application the ideology of an existing dance notation was computerized to compose a dictionary of dance moves. This was designed to enable a uniform storage mechanism for a subset of Salsa moves. The intention of the dance dictionary was to access the stored dance moves and create dance sequences. In order to generate dance sequences, a control mechanism was required to apply constraints to dance sequences to dictate what moves could be performed in succession. The construction of a context-free grammar (CFG) was instrumental in implementing the aforementioned functionality and was designed to interpret the constraints that a beginner dancer would experience during a dance sequence. This CFG was implemented in the application to actualize the sequence generation component. A user interface was designed to allow the users to access the application's capabilities. Both the interface and sequence generation component were tested with end-users of the application. During testing, the usability of the application was assessed in order to discover whether such an application would enhance the learning experience of students. Ultimately, the application met all its requirements and is agreeably conducive to the learning process of dance. It was concluded that this new approach to dance education is not only valuable, but has promising prospects for the future.

CCS CONCEPTS

• **Theory of computation** → **Grammars and context-free languages**; • **Software and its engineering** → **Designing software**.

KEYWORDS

parser, grammar specification, context-free grammar, dance notation, Salsa

1 INTRODUCTION

Dance is a performance art and is considered to be an ancient, cultural heritage. In addition, dance is not only a form of expression but provides the means to improve one's mental capacity [13]. Consequently, the preservation of dance is of the utmost importance. Oral communication is prone to error [12] and humans are circumscribed by memory, concerning the intricacies of past and present dance art forms. Therefore, it is imperative to archive various performance arts for future generations in order to facilitate the survival of dance. Due to the lack of comprehensive dance documentation

[19], dance novices are likely to record a video of a dance lesson with the intention of using this video to practice choreography. This approach often proves to be futile because the video quality may be substandard and there is no formal clarification of moves into clear steps and positions [40].

Historically, technology has been used as an instrument to actualize innovation in choreographic support tools [17]. From the limited amount of tools available, few of them support interactive or generative components [17]. The current technological platforms that are available to dance students favour live dance performances as a means to teach dance, which suffer from the aforementioned disadvantages [40]. Hence, there are still opportunities for superior solutions.

The Evolution Dance Company (EDC) [1] is Cape Town's first community based Salsa Dance company. It is an organisation that provides a friendly and professional environment for people to immerse themselves in the enthrallment of dance. It provides a creative outlet for the appreciation of this artform. The primary focus is on the development of dance, with the vision of making Salsa accessible to all the people of Cape Town. Therefore, the formulation of this project is in pursuance of this establishment's aspirations.

We proposed the development of a tool to aid and improve the learning process of Salsa dance, with no intent of replacing the traditional 'in-class' learning, but rather to supplement it. Our aim is to develop a user-friendly, software tool, in the form of a desktop application, for teachers to plan lessons that will provide a different approach to the way Salsa dance is taught. Furthermore, this tool will allow novice dancers to develop their skills by enabling them to analyze specific dance moves graphically, create dance sequences and learn dance sequences defined by teachers. Thus, our goal is to aid the dance community. The research questions we aim to answer are:

- How effectively can the application computerize a dance syllabus?
- How effectively can our software tool formalise a sequence of dance steps?
- How can the use of an application impact the dance skills of a novice Salsa dancer?

The remainder of the paper first introduces background information and relevant papers relating to the problem domain. The application's requirements and design decisions follow. Thereafter, theoretical analysis of a context-free grammar, to define a dance language, is conducted. The development and implementation of the system is then explained. The evaluation methods used to assess the project's components are analyzed to produce results before the conclusions, that were attained, are presented along with areas for future work.

2 BACKGROUND AND RELATED WORK

This section will present information necessary to understand the concepts that are referred to in this paper and an overview of research related to the project.

2.1 Introduction to Dance

Dance is a performance art and is composed of premeditated bodily movement [36]. This movement can be characterized as selected sequences of active steps. The acknowledgement of movement as a dance form is a shared affiliation in a particular culture by both performers and observers [36].

2.1.1 Salsa Dance. For this project, we opted for the Salsa dance style. Salsa is a syncretic dance form with origins from Cuba [6]. It is usually a partnered dance. The ‘partners’ are namely the leader and the follower. The leader is the director and indicates the succeeding move to the follower [4]. The leader is responsible for physically guiding the follower through the dance sequence. Salsa music is written with four beats per measure. The basic salsa dance step uses eight beats, therefore it is danced over two bars of music [4, 8]. The rhythmic composition of the Salsa basic step is defined as ‘quick-quick-slow’ [8] which is performed on beats 1-2-3, 5-6-7 of the 8 beats. Salsa as a dance style consists of many variations. The variation we chose is the Salsa On1. On1 salsa timing is known as L.A style salsa. This style is distinguished by its garish, agile moves and the ‘slow’ portion of the Salsa rhythm. The Salsa On1 timing can be further explained according to the following guidelines: “the lead breaks forward on 1 and back on 5”; “follower breaks backwards on 1 and forwards on 5”; “the slow counts are immediately after the break steps”[6]. Break steps are a change of direction in the salsa basic step.

2.1.2 Dance Notation. A dance notation, similar to musical notes, is a symbolic form of representing the movements of the dancers using various graphical symbols such as lines, circles, rectangles, squares, bars etc. “The primary use of a dance notation is the documentation, analysis and reconstruction of choreography”[22]. Dance notations entail expressing four-dimensional movement into two-dimensional space [18] and provides a means to conduct theoretical analysis on choreography [34].

2.2 Types and Applications of Dance Notations

There have been previous attempts to construct comprehensible dance notations to model dance choreography. Feuillet’s Notation was one of the earliest discovered types of dance notation. This notation is able to document foot positions [32] however, his notation is limited by its inability to represent movement for the upper part of the body [32]. Labanotation, a dance notation, was invented in 1928 [14]. It has been described as complicated and only easily understood by those who study it [14]. This dance notation is used for human movement but has not been optimised for a partnered dance. These notations focus on the more classical dance styles and do not allow users to experiment with different sequences or clearly see the movement of different parts of the body. Renesse and Ecke [39] created a ‘Space of Salsa Dance’ notation using mathematical equations in the form of a text-based diagram. This method is only implemented for arm movement and lacks notation for feet movement which is a fundamental part of Salsa Dance. In 2002, the “Salsa Dictionary” was created [7] as a way to learn Salsa turn patterns displayed in a table-based notation. The basic elements of

the system are hand holds, feet movement, directions and positions. This method however is not defined in XML or any other structured serialisation. In the past 17 years, there has been no uptake to the use of this notation. The aforementioned notations are paper-based notations, which can be easily misinterpreted [16] and difficult to conceptualize [15].

LabanWriter [27], LabanEditor[27] and DanceLaban[27, 43] are extensions of Labanotation. LabanWriter was created as a Labanotation editor. However, its usability is limited as it treats the symbols strictly as 2D objects and it does not perform grammar checks, thus we cannot ensure the accuracy of the elements. LabanDancer [27, 43] does not have a feature to prepare Labanotation scores. Preparation of scores is useful in enabling dance movements to be accurately interpreted. MovementXML [20] was then created as a tool that extends LabanWriter. MovementXML ensures that the score will always be correct. It has a structured nature making it possible to search for a pattern in the score. Despite this, the efforts were neither aimed at describing gestural interactions nor for Salsa.

2.3 E-Learning in Dance

In 2018, the WebDANCE project [24] was the first of its kind to be developed as an e-learning dance tool. This attests to the fact that there has not been much development in the approach to teaching dance and that there are no established tools to do so. The aim of the WebDANCE project was to develop a 3D platform in the interest of being able to visualize dance movements in a virtual environment [24]. Consequently, this would exploit the concept of e-learning by providing a more intuitive approach to learning.

Following the launch of WebDANCE, a project called OpenDANCE [30] was created. This project used the experience and results obtained by WebDANCE. An improvement that was made was providing functionality for users to input dance content online, giving rise to dance lessons [30]. This was done to enhance the interactive component between the web-learning environment and the user [30]. The OpenDANCE project finds application within the industry of dance education but may not be easily adapted for teaching Salsa, due to the absence of Salsa specific information and expertise.

Learn Salsa[2], *Salsa Anywhere*[3] and *Salsa Dancing*[5] are among the top rated applications with the aim of teaching its users the Salsa dance style. All three applications do not implement heuristic principles in furtherance of a good user interface. *Learn Salsa* and *Salsa Dancing* resemble organized databases. They supply users with links to YouTube videos and do not offer any original content. *Salsa Dancing* also contains text-based information, but this information is ill-formatted. Although *Salsa Anywhere* offers original content, users are forced to download them in order to gain access to the content. Furthermore, there is limited access to videos as some require in-app purchases. Nevertheless, the videos can be broken down according to the eight beat rhythmic composition of Salsa. The videos can also be separated into the different dance moves performed. All three applications rely on videos to educate their users to help them improve their skills. This approach to learning often proves ineffective as discussed earlier in Section 1.

2.4 Introduction to Compilers

The following information is central to the components of the project. They give a broad sense of the technicalities and concepts that were considered during the development process.

2.4.1 Context Free Grammar. A context-free grammar (CFG) consisting of a finite set of grammar rules is a quadruple (N, T, P, S) where

- N is a set of non-terminal symbols
- T is a set of terminals where $N \cap T = \text{NULL}$.
- P is a set of rules, $P: N \rightarrow (N \cup T)^*$, i.e., the left-hand side of the production rule P does not have any right context or left context.
- S is the start symbol

2.4.2 Parser. A parser is a compiler or interpreter element that separates information into smaller components for simple interpretation into another dialect. Parsing involves three steps:

- (1) **Lexical Analysis:** A stream of string characters are given as input to a lexical analyzer. These characters are separated into distinct elements to give rise to meaningful expression. Tokens are the output of this stage.
- (2) **Syntactic Analysis:** Analysis of the aforementioned tokens are carried out using a context-free grammar. The ordering of the tokens are checked against the grammar's algorithmic procedures to see if the tokens form a valid expression.
- (3) **Semantic Parsing:** The effect and ramifications of the approved expression are resolved and the necessary actions are performed.

2.4.3 BNF Notation. Backus-Naur Form (BNF) is a notation strategy to formalize a context-free grammar. BNF can also be utilized to define the syntactic meaning of a programming language. Extended BNF (EBNF) is a collection of metasyntax notations. EBNF makes a formal description of a programming language.

2.5 Parser Development

There have been previous works of generating original parsers and parser generators. Sabo et. al [33] took a unique approach to creating a parser generator prototype. In their work, a parser is created from a collection of annotated classes [33]. This is in contrast to the traditional approach of specifying a syntax or grammar in BNF notation. Although their efforts were aimed at investigating new approaches to generating parsers, they utilized JavaCC, an open-source parser generator written in the Java programming language, as a fundamental parsing technology. The difference between JavaCC and their prototype is that the prototypes's output can be produced for both top-down or bottom-up parsers whereas JavaCC uses primarily a top-down approach. Sabo et. al concluded that their approach to creating a parser generator may prove to be more efficient than traditional methods. Aggarwal et. al [42] also focused on improving efficiency, by constructing a context free grammar to automate password guesses. Their grammar specification proved insightful owing to a systematic and comprehensive explanation for each production in their context-free grammar. In comparison to Sabo et. al, Aggarwal et. al [42] did not utilize any existing parsing tools. Another approach to grammar specification was proposed by Milani et. al [22] who sought to generate a context-free grammar for mathematical expressions. Unlike Aggarwal et. al, they used an algorithmic approach to inspire their grammar specification and hence, were able to support their design decisions in the generation of their grammar.

3 REQUIREMENTS GATHERING AND DESIGN

This section outlines the strategy adopted during the design phase of this application. The inspiration for this project stems from the problems faced by the dance community. An example is illustrated by Tanya Karen [23], who wrote a blog-post about documenting her Salsa dance lessons in notebooks and utilizing this written version to recall dance choreography.

Existing paper-based notations are complicated for students to record and model dance choreography. Current applications, for teaching the Salsa dance style, rely on videos to capture dance moves. There are tools that implement paper-based dance notations to teach dance, however no tools of this nature exist solely for the purpose of teaching Salsa dance style.

3.1 The Requirements

Our client is Angus Prince from EDC, discussed in Section 1. The requirements for this software tool arose from a combination of our client's aspirations and our ideas to eradicate the aforementioned problems. From the client's knowledge, there is no application that possesses a mutual approach to dance education. As a direct result, we attended a beginners' Salsa dance lesson, taking an ethnographic approach to understand our users. This firsthand experience allowed us to consider what a user would desire or need our application to possess that would enrich their experience. Furthermore, we were able to familiarize ourselves with basic steps and Salsa concepts. In addition to this method, we conducted a user experiment with our client, during a second meeting we had with him. He gave us insight into what he required from this application. To prevent scope creep, we prioritized and selected the application's functionality. This ensured that our scope was focused and concise which optimized the software development process. The resulting requirements, pertaining to this report, are provided below.

- A dance student should be able to:
 - View information on dance moves
 - Have access to a beginner's dance syllabus
 - Create a valid dance sequences for a pair
 - Enhance their dance skills
- A dance teacher should be able to:
 - Define dance sequences for students

The objective of the design process was to gain a conceptual view of the system in accordance with the client's needs, to anticipate the expected behavior of the final product and how to make this system efficient and effective. A co-design technique was employed. This commenced during our initial meeting with our client and persisted with electronic communication. We regularly updated the client on our progress and subsequently received constructive feedback which was applied to the system. We adopted a structured design [41] method with a top-down approach [38] which involves the decomposition of the system as one entity. Structure design [41] can be understood as solution design and aims to conceptualize the problem into distinct, well-organized components. The fundamental components chosen to deliver the requirements are:

- A dictionary of dance moves
- A user interface
- A control mechanism for sequence generation
- A sequence generator

3.2 Dance Dictionary

Our client’s expertise provided us with a beginner dance syllabus which will be represented by the dictionary. A dictionary was required to promote a well-organized and uniform system to store dance moves. From Section 2, dance notations are used to document dance choreography in a logical manner. Hence, using a dance notation was appropriate for the requirements. There were two options for designing a dance notation for the application, namely constructing an original notation or leveraging an existing one. We assessed the implications of creating our own notation. We had to consider the limitations of our Salsa knowledge and the time it would take to create and refine an original version. We opted to use an existing notation, specifically the Salsa Dictionary [7] referred to in Section 2. The Salsa Dictionary’s target market is beginner dancers. They employ a salsa method which decomposes moves into handholds, directions, actions and positions, which provides an intuitive separation of elements. The directions are based on the notion of a line of dance, the dancers being either parallel or perpendicular to this line, and on the directions of the dancers in relation to each other. The Salsa dictionary includes variations of the aforementioned elements, a written explanation of each element, a visual example and corresponding symbol, illustrated in Appendix F. Furthermore, this notation defines moves for a pair, a leader and a follower, which coincides with our solution. A move is illustrated by a matrix of *salsa lines* and *salsa elements*, illustrated by Figure 1. The columns represent four beats in salsa choreography. This notation was modified and extended to design the dictionary for our application. There is ambiguity in the representation of the symbols of a normal open handhold and a normal closed handhold, as both use N [7]. Furthermore, the same symbol, an asterisk, is used to indicate whether the hands are held up or down on either the left or right side, by placing it on either side of the handhold symbol e.g. $*N$ denotes that both the left and right hands are held. To compensate for the ambiguous nature of the notation, we adjusted the symbolic representation of the notation, but preserved its fundamental principles. The revised notation uses a $*$ to signify if the hands are held up and a $+$ is used to signify if the hands are held down. The symbol NC is designated for normal closed hands. The terms ‘man’ and ‘lady’ has been changed to follower and leader. The directions utilize arrows to symbolize the respective positions of the dancers. Some variations of the directions occupy two lines. This was altered to occupy one line, by using a ‘|’ to indicate the change in line, to offer uniformity and simplicity in the symbolic representation e.g. $\left\{ \begin{matrix} M \\ L \end{matrix} \right\} = M|L$. In addition, more suitable names have been assigned to each dance move. The dance syllabus, composed of the symbolic representation and updated names for the dance moves, can be found in Appendix A. The structure of a dance move is illustrated by Figure 2.

HandHold		
Direction		
Man		
Common Action		
Lady		

Figure 1: A matrix of elements to define a Salsa move

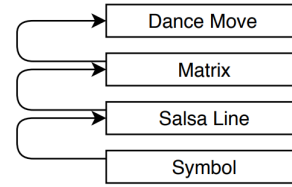


Figure 2: The structural composition of a Salsa move

3.3 User Interface

The design of the interface was influenced by the required functionality and the intention to enhance the learning experience of dance students. When designing a solution, we addressed questions such as what are the barriers that are hindering dancers’ access to efficient dance tools and what problems are experienced in the dance community.

3.3.1 Design Techniques. We opted to follow the User-Centred Design process (UCD), to achieve user satisfaction. Our intended users are students and teachers of Salsa. Our attendance at a beginners’ Salsa class and regular communication with our client fueled this process. During our meetings with our client, we collaboratively explored design ideas and discussed the implications of the design. In addition, we gathered information from dance students to discover their design concerns.

Personae were established to inform the ideation practices. An example of a persona, was an elder female, who recently started Salsa dancing. This technique encouraged us to be mindful of our users.

Paper prototypes were developed in rapid iterations. Each iterative step was an advancement using a self-assessment technique. The iterative process of developing the interface focused on improvements to the usability and the possible user satisfaction. Prototyping allowed us to exhaust all design options. UCT students, who were enrolled in an Human Computer Interaction (HCI) course, were consulted to review a refined paper prototype. Design flaws were discovered, and placement of elements and the flow were critiqued. This feedback was used to revise the design and improve it based on what was observed. This ensured that a refined design was created before software development began.

3.3.2 Design Principles. User experience (UX) design targets the users and helps to achieve the overall goal. From Section 2, one may deduce that the existing applications for Salsa dance education lacks in user experience, therefore we placed importance on UX design to have a competitive edge. We were inspired by Susan Weinschenk’s UX psychology facts [11] about the human mind, which can be directly applied to design decisions conducive to an intuitive user interface. Her approach encourages focus on the user throughout the design process, which correlates to the UCD technique. Her concept of a visual system was adopted, which guides the layout of an interface. Another principle affecting the layout of the interface is Fitt’s Law [29]. This law also influenced the size specifications of the interface elements. Another fact is that people make mental models. A mental model correlates to a person’s cognitive process concerning how something operates in the real world. Hence, the design was influenced by activities done by dancers. Lastly, the user’s attention is essential, which has aesthetic implications.

Other usability heuristics adopted was to promote recognition rather than recall [9], by partnering functions with a unique and complementary graphic. Furthermore, we wanted to provide the user with helpful information on how to use the application. We hoped that the UI design would be intuitive and therefore promote minimal use of the available support. Another important principle is the visibility of system status [9]. The design decision to offer the user informative feedback on the system’s current state embodied this concept. We also had to consider the possibility of color blind users. Thus, wherever colour is used to convey information, there is a complementary, secondary cue for users who are unable to view the colours presented. An additional design goal was to make the interface easily navigable, which is a benchmark for great UX design. Hence, access to a menu that meticulously divides the application’s capabilities was designed to be available to the user at all times.

3.4 Sequence Generator

The core functionality of this application comprises the analysis and generation of dance sequences. This component aids dance instructors in planning lessons. It also allows students to create dance sequences to improve their dance abilities. This component utilizes the dance moves from the dance dictionary: sequences are made by adding dance moves. In advanced Salsa dancing, any dance move can follow any other dance move with the aid of an *intermediate step*. This intermediate step advances the dancers to a position where they are able to perform the following step. This approach to sequence creation is complex. It requires the end state of the preceding move, the start state of the following move and a method to choose an appropriate intermediate step. An exhaustive list of intermediate steps would be required to circumvent cases where there is no available intermediate step for a particular sequence. Furthermore, the focus of this application is teaching introductory Salsa.

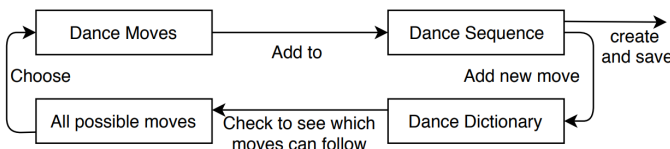


Figure 3: The Process of Validation

Consequently, an alternative approach was adopted. We opted to design a control mechanism that determines which dance moves may be performed in succession. This control mechanism, thus governs the validity of a sequence. This limits the structural composition of sequences, but this approach is more suitable for beginner dancers. The initial design for this mechanism included feedback after the sequence was created, informing the user whether the sequence is valid not. This design has the following shortcomings: time would be wasted creating a sequence that could possibly be invalid and the user may repeat an invalid sequence structure, therefore the process would be tedious and inefficient. A more efficient design was developed to process the sequence in real-time. This design enforces and updates restrictions of which moves can be added to the sequence, therefore the user is allowed to only select moves that can follow the preceding one. This process of validation is illustrated by Figure 3.

An option to design this functionality was to utilize Bean Validation [10]. This framework would require the design of annotations

and validator classes. Although this framework is widely adopted for designing constraints, it was not an optimal method for this problem domain. It would require explicitly stating every constraint for every move. A more appropriate approach was to design a parser to check the validity of dance sequences. A parser would be able to analyse the sequence against a set of rules to detect its validity. A grammar specification was required to design and control the constraints of dance moves.

4 THEORETICAL ANALYSIS OF GRAMMAR SPECIFICATION

The analysis of dance sequences enforces constraints of dance moves to determine the sequence’s validity. This equates to the notion of whether or not a dance sequence is possible to perform. This functionality was implemented by means of a context-free grammar, whose precise structure investigates if it compiles with the defined rules thereby facilitating error detection.

4.1 Constraints and The Dance Language

The limitations of which dance moves are possible to do in succession is dependent on the state of the dancers at the end of a dance move. Our client offered his expertise that informed us that variations of handholds are the most affecting element on what moves can be performed. From a technical perspective, the position of the dancers in relation to the line of dance and each other also has implications on which moves may be possible. Thus, these elemental properties administer the constraints. The notion of a dancer’s state required application to the unfamiliar problem of computerizing a dance sequence. In conclusion, the language for generating a dance sequence, to be implemented by the CFG, can be defined according to the following two elemental properties:

- The type of handhold for a sequence must persist unless otherwise altered during a dance move.
- The positions of the dancers in relation to one another dictate how the next dance move should start.

Once the requirements for the grammar were established, development of the grammar commenced to apply the constraints.

4.2 Grammar Development

The CFG was developed in iterations until the desired outcome was reached. Each iterative step comprised the design of production rules to enforce the constraints of the two elemental properties, namely the handholds and directions. This was followed with paper-based testing.

4.2.1 Iteration 1. The initial approach involved creating unique terminals to represent the constraints of the elemental properties. The resulting complexity of an original symbolic representation was unnecessary and therefore, we opted to use the symbolic representation offered by the Salsa Dictionary [9].

From Figure 2, the elemental properties affecting the grammar are *Handhold* and *Direction*. The variations of handholds that were included were:

- *normal open holds*, denoted by *n*, where dancers hold left to right and/or right to left either up, denoted by *+*, or down, denoted by ***.

- *crossed holds*, denoted by c , where dancers hold left to left and/or right to right either up, denoted by $+$, or down, denoted by $*$.

If the hands are held up or down on either the left or right side, the $+$ or $*$ will be placed on the respective sides of the n or c e.g. $+n^*$ denotes that the dancers' handhold is a normal open hold where the left hand of the leader and right hand of follower is held up while the right hand of the leader and left hand of the follower are held down. The $+$ or the $*$ is omitted if the dancers' hands are not held. Since it takes one beat, in the rhythmic composition of Salsa, to either leave or hold a hand, variations of the hand positions may follow one another. Conversely, variations of handhold types may not follow one another, considering it traditionally takes four beats to shift between handholds types. The variations of directions that were included is when both dancers, the follower, denoted by f , and the leader, denoted by l , are facing the line of dance:

- Facing each other, denoted by either $f \rightarrow \leftarrow l$ or $l \rightarrow \leftarrow f$
- Facing the same direction with the follower in front, denoted by either $f \leftarrow l$ or $l \rightarrow f$
- Facing the same direction with the leader in front, denoted by either $f \rightarrow l$ or $l \leftarrow f$
- Facing opposite directions, denoted by either $f \leftarrow l$ or $l \leftarrow f$

As a result the terminals for the following grammar specifications are: $T\{n; c; +; *; f; l; \leftarrow; \rightarrow\}$ where n = normal open hold, c = crossed hold, $+$ = up hold, $*$ = down hold, f = follower, l = leader and combinations of the arrows determine the directions of the dancers. An initial string, " $*n^*n^*n^*nf \rightarrow \leftarrow lf \rightarrow \leftarrow lf \rightarrow \leftarrow lf \rightarrow \leftarrow l$ " was chosen to serve as regulatory test case for a dance sequence. The string comprises four dance moves where the follower and leader are facing each other and have a normal open handhold occasionally leaving either the right or left handhold. The input string would be in the form Move 1: <handhold element> <direction element>; Move 2... The dance sequence representation was a sequence learnt during the introductory Salsa lesson we attended. A \times symbolizes the grammar's failed attempt to generate the test case.

4.2.2 Iteration 2. $G_2:: P_1: S \rightarrow BE \quad P_2: B \rightarrow BJKJ|e$
 $P_3: E \rightarrow EGHG|e \quad P_4: G \rightarrow f|l \quad P_5: H \rightarrow \rightarrow|\leftarrow$
 $P_6: K \rightarrow c|n \quad P_7: J \rightarrow +|*|e$

Expansion: $S \Rightarrow BE \Rightarrow BJKJE \Rightarrow BJKJKJE \Rightarrow BJKJKJKJE \Rightarrow BJKJKJKJE \Rightarrow BJKJKJKJKJE \Rightarrow JKJKJKJKJE \Rightarrow *cJKJKJE \times$

The aim of this grammar was to validate an unlimited amount of moves. This grammar experienced several shortcomings. During the expansion it was possible to access both a crossed and open hold, thus not conforming to the test input string. In addition, it was possible to have more handhold elements than direction elements making the format incorrect. It was possible to have zero handholds which does not correspond to an accurate representation of a dance move. Thus, the desired outcome was not attained and the constraints were not represented correctly.

4.2.3 Iteration 3. $G_3:: P_1: S \rightarrow A \quad P_2: A \rightarrow BC$
 $P_3: B \rightarrow ADED|DED \quad P_4: C \rightarrow CGFFG \quad P_5: D \rightarrow +|*|e$
 $P_6: E \rightarrow c|n \quad P_7: F \rightarrow \rightarrow|\leftarrow \quad P_8: G \rightarrow f|l$

Expansion: $S \Rightarrow A \Rightarrow BC \Rightarrow AEDC \Rightarrow AEDDEDC \Rightarrow AEDDEDC \Rightarrow DEDEDEDEDC \Rightarrow *c \times$

This grammar sought to rectify the format issue experienced by the grammar before. It was successful in this attempt, but assorted combinations of handhold types and directions were possible, thus

disregarding all the constraints. Furthermore, during the expansion it suffered the same shortcoming as the grammar in Section 4.2.2.

4.2.4 Iteration 4. $G_4:: P_1: S \rightarrow A|H \quad P_2: A \rightarrow BC$

$P_3: B \rightarrow ADED|DED \quad P_4: C \rightarrow CfF|ClFf|ff|l|Ff$

$P_5: D \rightarrow +|*|e \quad P_6: E \rightarrow c \quad P_7: F \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \leftarrow \leftarrow \leftarrow \leftarrow \leftarrow \leftarrow \leftarrow$

$P_8: H \rightarrow IC \quad P_9: I \rightarrow HDLD|DLD \quad P_{10}: L \rightarrow n$

Expansion: $S \Rightarrow H \Rightarrow IC \Rightarrow HDLC \Rightarrow HDLDDLDC \Rightarrow HDLDDLDC \Rightarrow DLDDLDDLDC \Rightarrow *n^*n^*nC \Rightarrow *n^*n^*n^*C|Ff \times$

This grammar intended to fix the recurring problem experienced by the two previous grammars, by isolating the handhold type. Thus, during the expansion the handhold property of the dance sequence was accounted for, however the direction constraint was not represented correctly.

4.2.5 Iteration 5. During this iteration, the design decision to check the dance sequence at every step of adding a new move was employed. Thus, the grammar was designed to account for only two moves. The new test case was a subset of the previous test case: " $*n^*n^*nf \rightarrow \leftarrow lf \rightarrow \leftarrow l$ "

$G_5:: P_1: S \rightarrow YYA|YYB|ZZA|ZZB \quad P_2: Z \rightarrow XnX \quad P_3: Y \rightarrow XcX$

$P_4: X \rightarrow +|*|e \quad P_5: A \rightarrow CKDCDK|CLDCLD|CODCOD|CPDCPD$

$P_6: B \rightarrow DKCDKC|DLCDLC|DOCDOC|DPCDPC$

$P_7: C \rightarrow f \quad P_8: D \rightarrow l \quad P_9: K \rightarrow MM \quad P_{10}: L \rightarrow MN$

$P_{11}: O \rightarrow NM \quad P_{12}: P \rightarrow NN \quad P_{13}: M \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \leftarrow$

Expansion: $S \Rightarrow ZZA \Rightarrow XnXZA \Rightarrow *n^*ZA \Rightarrow *n^*XnXA \Rightarrow *n^*n^*CLDCLD \Rightarrow *n^*n^*fLDCLD \Rightarrow *n^*n^*fMNICLD \Rightarrow *n^*n^*f \rightarrow \leftarrow ICLD \Rightarrow *n^*n^*f \rightarrow \leftarrow lf \rightarrow \leftarrow l \Rightarrow$ test case reached \checkmark

The above grammar is context-free because it is of the form

$A \rightarrow \beta$ and $A \in V_N$ and $\beta \in V_T$ where V_N represents nonterminals and V_T represents terminals.

In order to ensure the grammar's competency, several test cases were performed to ensure if implemented the constraints and satisfied the dance language. The test cases were designed to explore all possible combinations of terminals, thus providing a thorough analysis of the CFG. The test cases are evidence to a successful design of the grammar.

Test Case	String Input	Expected Outcome	Actual Outcome
1	" $n^*nf \rightarrow \rightarrow lf \rightarrow \rightarrow l$ "	\checkmark	\checkmark
2	" $n^*cf \rightarrow \rightarrow lf \rightarrow \rightarrow l$ "	\times	\times
3	" $ccf \leftarrow \rightarrow lf \leftarrow \rightarrow l$ "	\checkmark	\checkmark
4	" $ccf \leftarrow \rightarrow ll \leftarrow \rightarrow f$ "	\times	\times
5	" $n^*n+f \rightarrow \leftarrow lf \rightarrow \leftarrow l$ "	\checkmark	\checkmark
6	" $*n^*nf \leftarrow \leftarrow lf \rightarrow \leftarrow l$ "	\times	\times
7	" $n+n+l \leftarrow \leftarrow fl \leftarrow \leftarrow f$ "	\checkmark	\checkmark
8	" $nc+f \rightarrow \rightarrow ll \rightarrow \rightarrow f$ "	\times	\times

Table 1: Testing the context-free grammar

5 SYSTEM DEVELOPMENT AND IMPLEMENTATION

This section outlines the development strategies and the implementation process for the system creation. The system was built in accordance with the requirements specification. This project was developed using the iterative waterfall method. The iterative approach to the classical waterfall method has feedback paths from every phase to its preceding phase. To elaborate, once we received

the requirements for the software project we conducted analysis and designed the respective components. This was followed by the implementation of the components which were tested to see if the expected behaviour was accomplished. The testing and monitoring of the component's outputs allowed us to discover the underlying problems and contemplate possible enhancements. A feasibility demonstration was performed in the initial iteration. The purpose of the demonstration was to evaluate the project's feasibility and to provide us with feedback to continue before development commenced. Activities were planned for each respective system components. Each sub-component was built parallel to each other and was later integrated into the system. Implementation commenced with choosing an appropriate programming language. The chosen programming language was Java because it automates memory management[35], the build process is quicker and runtime error detection is the system's responsibility[35]. In addition, Java generates bytecode after the compilation of a program which is platform independent and thus, portable. Java supports Object Oriented Programming (OOP). This style of programming is easy to conceptualize and ensures that the system is flexible, extensible and modular. Java also has a rich API and supports many additional libraries that offer high-level services [35]. Furthermore, the decision for the programming language was based on the possibility of easier integration with the components of the application discussed in this paper.

Our team had regular weekly meetings in order to brief each other on the progress of our respective components and dwell on integration strategies to ensure that we all shared a common end goal. During the project life-cycle, we communicated with our project supervisor through email and met for regular meetings to discuss progress and strategies to move forward. This was advantageous to us as we received critical help and criticism on the development thus far. These meetings were a reminder of the project requirements which ensured we stayed on track. In addition, we communicated with our client via e-mail and had a WhatsApp group for constant communication and feedback, which supplemented the five meetings we had.

Thorough testing and maintenance, described later, was done throughout the project life-cycle to ensure the components were always in a releasable state. The requirements helped to design the testing framework. Constructive and thorough documentation was recorded in the project code. This was done to ensure that, in an open source environment, all users would understand the code and its functionality. A GitHub repository was created to store the remote code and commits were made throughout the project life-cycle. This application is able to run on any operating system, but requires external installations of Java[35], JavaFX[31], JFoenix and JavaCC[26].

5.1 Dance Dictionary

A subset of steps to define the different moves associated with the Salsa On1 dance style were implemented. The Salsa steps were implemented using the design of the dance notation discussed in Section 4. and an OOP model. The notation is primarily for the backend functionality, therefore each move contains a human-recognisable dance term for users as opposed to complicated symbols to represent a dance schema. The class structure is illustrated by Figure 5.

An Element is a symbolic representation of one of the four components which comprise a dance move. This will either define the handhold, direction, position action for a Salsa move. The Step class defines movement for four beats in the rhythmic composition of a Salsa move. The addition of this class is due to the variations that may occur in Salsa moves. A Salsa move may comprise of more than one four-beat movement, therefore, a single move can be made of one or more steps. The reason to substantiate the use of an array in the Step class is that each matrix must contain five entries for the handhold, direction, leader, common action and follower. An ArrayList was used in both the Move and DanceDictionary class. This data structure was chosen because it is possible to know the maximum capacity for the ArrayList which ensures optimal performance [21]. Furthermore, there are no intensive functions that would jeopardize the performance of the application by not utilizing a more computationally efficient data structure. The data for dance moves are stored in a YAML file. The file specification is included in Appendix B. YAML is a "human-readable data-serialization language" [37]. It allows the data to be stored in a formatted way. The loadMoves() method uses the YAML file to initialize the dictionary, load the moves and create move objects according to the class structure above. The attributes separate the information for a dance move into coherent components. These components are used in the user interface to display information in a way that is useful to users. During the implementation phase we had a meeting with our client, during which he provided clarification of the dance moves we had selected to utilize.

5.2 Sequence Generator

The context-free grammar, which is theoretically analysed in Section 4, imposes constraints on which moves can be performed in a sequence. Thus, the context-free grammar had to be implemented to form the parser.

5.2.1 Frameworks. The framework employed for the parser creation was JavaCC [26]. JavaCC is a lexer and parser generation tool for LL(k) grammars. JavaCC also supports other capabilities associated with tree building and debugging. A parser generation tool that processes a grammar specification. The grammar specification of Section 4.2.5 is converted into a Java program that can observe matches to the specified grammar. The lexical and syntactic description of the grammar must be specified in a JJ file. This JJ file is then compiled using *javacc* to generate the lexer and a recursive descent parser. The JavaCC parser generator tool produced java code which made it easy to integrate with our other components, as the programming language chosen was Java.

5.2.2 Grammar Implementation. The grammar had to be specified in EBNF form during implementation. In addition to this, JavaCC cannot compile the parser successfully if choicepoints in a production exist [4]. Choicepoints occur when the productions contain more than one option for expansion. Thus, the implications were that two nonterminals, for different choicepoints, in the same production on the right hand side may not begin with the same terminal/nonterminal. This affected productions 2,3,5,6 and 9-12 from the grammar specified in Section 4.2.5. Consequently, necessary alterations to the productions' structure were made, while still maintaining the fundamental interpretation. This was tested using the test cases in Section 4.2.5 which is presented later. The grammar's tokens had to be explicitly listed. The productions resembled

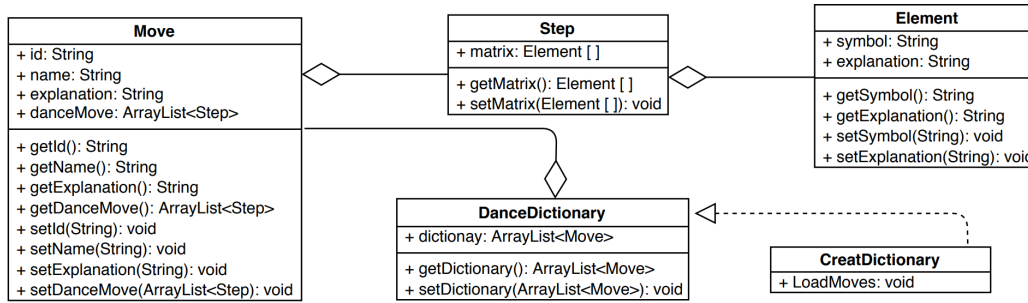


Figure 4: UML Diagram for the Dance Dictionary

the structure of a function. The implemented grammar can be found in Appendix C. The code was created to do a syntax check only. It was not necessary to develop an interpreter because the desired functionality was to check the input and determine the validity of the syntax. Hence, syntactic analysis is carried out. During this process, the parser receives a string as input. This input symbolizes the dance moves that have been chosen to create a dance sequence. The parser will inspect this string in relation to the rules specified by the grammar during sequence generation.

5.2.3 *Sequence Generation.* During sequence generation, moves are added sequentially. During each sequential step each dance move from the dictionary is checked to determine whether it may follow the last move added to the sequence. This process involves extracting the symbols necessary for the syntactic analysis. The end states of the handholds and directions of the last move is combined with the start states of the handholds and directions of every move to create an input string. This is done in iterations. During each iteration, the input string, which is the symbolic representation for two moves, is passed to the parser. Due to the restrictions of the implemented grammar, the input string is formatted before it is given as input. If the input string is valid, the dance move is added to a group of valid dance moves which are made available to the user. This iterative approach was implemented to make the process optimal for the user. A graphical representation of this process is included in Appendix D.

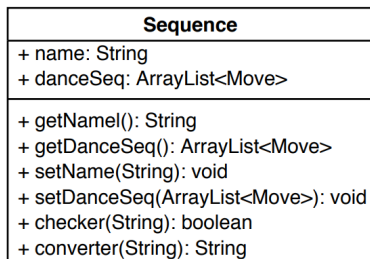


Figure 5: Class structure for Sequence Generation

From Figure 5, the checker method is called in every iteration. This method invokes the parser’s main method. Thus, if the syntax is valid the checker method returns true to indicate that the dance move being checked should be made available to the user. The converter method converts the string before it is given as input to the parser. Once a sequence is stored it is written to a YAML file. There exists one YAML file for each user. The YAML file contains a sequence name and the respective moves which can be identified

by their ID and filepath. The file specifications can be found in the Appendix B.

5.2.4 *Testing.* For testing purposes, the complementary input string from the test cases in Section 4.2.5 were supplied to the parser. The parser processed the string and provided output in the form of a message specifying if the string has the correct syntax or not. Below are the converted test cases to determine if the implementation of the context-free grammar was correct. The output was either "The syntax is not good" implying an error or "The syntax is good" implying a valid string. From Table 2, it is apparent that the parser behaves in the expected manner.

Test Case	String Input	Expected Outcome	Actual Outcome
1	"n*nfl((fl(("	✓	✓
2	"n*cfl((fl(("	✗	✗
3	"ccfl)(fl)("	✓	✓
4	"ccfl)(lf)("	✗	✗
5	"n*n+fl()fl()"	✓	✓
6	"*n*nfl)fl()"	✗	✗
7	"n+n+lf))lf)'"	✓	✓
8	"nc+fl((lf(("	✗	✗

Table 2: Testing the grammar implementation

5.3 User Interface

During implementation, UCT students, who had been previously enrolled in the HCI course offered by the Computer Science Department at UCT, were consulted to analyse the user experience of the interface. They were supplied with a list of five tasks to perform, included in Appendix D, and expressed their view on the usability. This process allowed us to receive constructive feedback and implement improvements. We received suggestions pertaining to the size of interface elements and to provide easier navigation for the user.

5.3.1 *Frameworks.* The platforms used to create the interface was JavaFX [31] and SceneBuilder. The choice of platform was influenced by an advanced aesthetic experience that is achievable using these frameworks. It also allows for fast UI development with Scene Builder to ensure that less time is spent on trivial activities, hence it simplifies development. It also has a rich toolkit and is friendly with the MVC pattern [28]. Additionally, one may use CSS [25] with JavaFX which enhances the application’s appearance and formatting is easier. CSS contains ‘style rules’. These rules are enforced

by the program and applied to the respective interface elements. Furthermore, applications made using JavaFX are compatible with many devices, like mobile phones, TVs, desktop computers and tablets. This means that this application can be launched to any of the above devices.

5.3.2 Functionality. The login process provides a personalized experience for users and differentiates dance instructors and dance students. Users may navigate to the homepage. The following features are implemented in a concise menu: access information on Salsa; view information on beginner dance moves; create and save dance sequences; view, edit or delete stored sequences. This menu is always available to the user for easy navigation. The interface for viewing information on dance moves, arranges the moves into a simple list. A user may click on a move in the list-view to display information pertaining to that move. The sequence generation interface displays a list of dance moves and an empty list for the sequence. These are clearly defined. When a user begins the sequence creation process, invalid moves are disabled to signal to the user that they may not be used. Feedback is supplied to the user when actions are made using alert dialogs. There is a consistent theme and aesthetic appeal. The user interface is simple and intuitive and encompasses good UX design. Images of the UI can be found in Appendix E.

5.4 Ethical, Professional and Legal Issues

In order to conduct evaluation testing to discuss the outcomes of our design and implementation, we received ethical clearance from the Faculty of Science Research Ethics Committee. We will also comply with the ethical principles from the Belmont Report [44] relating to the participants.

As developers, we hold the intellectual property rights of this project that has been developed in the university environment. We will open-source our project in order to promote external improvements and encourage innovative ideas to the problem. Furthermore, the outputs of this project may be released to the community and, as such, there is a professional responsibility to ensure that the output is of a high standard.

6 EVALUATION METHODS AND RESULTS

6.1 Grammar Implementation Testing

6.1.1 Aim. To determine the robustness of the constraint system in identifying incorrect configurations of dance sequences.

6.1.2 Participants. Two dance teachers from EDC, who possess the expertise suitable for evaluating this component. They were recruited based on our ongoing communication with EDC.

6.1.3 Procedure. This process was conducted at the EDC studio. The teachers were introduced to the problem domain, the sequence generation component and aim of this process. Full consent was obtained from the participants and they were informed that they could stop participation at any point during the evaluation. Each participant was handed a form containing a list of the implemented dance moves identified by a number and four empty tables. They were asked to create two valid and two invalid dance sequences in the provided tables using the unique numbers assigned to dance moves. They were also requested to provide a reason for the sequences validity or invalidity. They were able to ask questions

for clarification. Approximately fifteen minutes were occupied for this process. The eight formulated sequences served as test cases to be tested in the application to determine the accuracy of the implemented grammar.

6.2 Results of Grammar Implementation

Below is a table representing the dance moves that were used by the dance teachers to create their valid and invalid sequences. Some dance moves have variations dependent on the directional element. Below this, is a table displaying the test cases that were captured during testing.

ID	Dance Move	Dancer Orientation
1.1	Basic with open hands, lead front	f → ← l
1.2	Basic with open hands, lead front	l → ← f
2.1	Basic with open hands, lead back	f → ← l
2.2	Basic with open hands, lead back	l → ← f
3.1	Basic with crossed hands, lead front	f → ← l
3.2	Basic with crossed hands, lead front	l → ← f
4.1	Basic with crossed hands, lead back	f → ← l
4.2	Basic with crossed hands, lead back	l → ← f
5.1	Basic with handhold change, leader back	f → ← l
6.1	Cross Body lead	f → ← l
7.2	Double turn, right hands held	l → ← f
8.2	Spot turn, left hand to right hand	l → ← f
9.2	Lead comb	l → ← f
10.1	CBL right turn, right hands held	f → ← l

Table 3: Dance moves utilized during testing

Valid Sequences				Invalid Sequences			
VS1	VS2	VS3	VS4	IS1	IS2	IS3	IS4
1.1	1.1	3.2	1.2	1.1	3.2	1.1	1.2
2.1	5.1	4.2	2.2	4.1	4.2	2.1	2.1
1.1	3.1	3.2	1.2	6.1	3.2	1.2	1.2
6.1	4.1	7.2	9.2	3.1	8.2	10.1	2.2

Table 4: Invalid and valid dance sequence test cases

These test cases were used to generate sequences in the application. Sequences VS1 - VS4 (valid sequences) were successfully generated. The teachers' reasoning to why these sequences were valid were that the type of handholds in a particular sequence persisted through the entire sequence, unless a move was utilized to alter the handhold type in the case of sequence VS2. In addition, the orientation of the dancers remained constant which indicates the validity of a dance sequence. Thus, the implemented CFG from Section 4.2.5 successfully interpreted the constraints according to the expertise of the dance teachers involved in the testing.

The reason for IS1 (invalid sequence) not being valid is that the handhold property does not remain constant, therefore the implemented CFG was able to distinguish the different types of handholds. IS2 was not possible considering that the last move, which was a spot turn with a normal hold, followed a move which utilized a crossed hold. Both IS3 and IS4 are invalid due to the inconsistent orientation of the dancers' directions. Usually, one dance move is required to alter the direction of the dancers and no such move is present in these cases. From the test cases utilized, the application conformed to the intended behaviour, thus the implementation of the CFG met the requirements of the problem domain.

6.3 Usability Test

6.3.1 *Aim.* To determine the application's usability for dance students and dance teachers.

6.3.2 *Participants.* Four dance students and two teachers from EDC. They were recruited with aid from our client and received muffins to convey our gratitude for their participation.

6.3.3 *Procedure.* This usability test was conducted at the EDT space. Before the test commenced, the participants were introduced to the problem domain, what the aim of the test was and the procedural structure. Full consent was obtained from the participants after the introduction and they were informed that they could stop participation at any point during the evaluation. The participants were prompted to gather in a group formation to conduct the usability test in a workshop style. A laptop was placed before them and the application was loaded. We conducted a simple demonstration of the application and explained the desired functionality. The participants had a chance to individually navigate the application themselves.

This guide was followed by a constructive discussion between the participants, during which their discussed common and unique opinions. During the discussion, the participants were prompted to stay on track and be honest in their communication. Questions were allowed during the procedure with regards to the intended usability or clarification of the application. After the discussion, each participant was handed a questionnaire which can be found in Appendix F. The questionnaire consisted of questions concerning the main factors affecting the usability of the application. The deductions of this experiment were based on participants' questionnaires and points considered during the discussion.

6.4 System Usability and User Satisfaction

The experimental method used to conduct the usability testing proved insightful. It was advantageous to evaluate the application with the selected participant group because we were able to receive direct feedback that was both constructive and complimentary. The workshop approach to the testing was chosen due to the participants' time restrictions. The workshop style, although lacking the personal experience of individual testing, was an effective method. The request that the participants individually navigate the application was beneficial to the process and allowed them to gain perspective into when and how they would utilize this application. The participants were able to express their views, listen to others and discuss amongst themselves. The discussion ran smoothly, with little need for us to interject. Although, participants were prompted to share their views, it was possible for others to influence their opinions. Another drawback was that the discussion would occasionally focus on future aspirations of the application and had to be directed back to the aim of the investigation. As a precautionary measure, the participants completed the questionnaires individually and unattended to receive their honest opinions.

The overall feedback on the application was positive, with all the criticism being constructive and leading to enhancements to the final design of the interface. Participants valued the simplicity and consistency, with praise being given to the final UI theme and design. The navigation, a property that was meticulously deliberated, was agreeably well-implemented. All the interface's elements had good affordances, since there was no confusion on how to interact

with them. The participants' described the interface as easy to use and understandable. The introductory information that the application provides to the user was appreciated. The participants were able to recognize the beginner dance moves that were implemented and thought the layout of information was clear and concise. In addition, the shared opinion was that the sequence generation process was efficient and easy, but that it could be enhanced with a horizontal view of the moves distributed over the Salsa rhythmic composition of eight beats. This feature was not added to the application because it would be most helpful with a visual representation of the sequence, to determine and view what step occurs at each respective beat. A suggestion to have a clearer division to the sequences created by the teacher and the sequences created by the user was proposed and implemented. A point was raised, by a teacher, to limit the amount of moves that can be added to a dance sequence to four. The reason for this is that beginner dance sequences are usually composed of four dance moves. A decision was made to provide the user with a warning message once four moves were added to a sequence, rather than restrict the user's control over the application. Other suggestions that were raised were to add background music and add a feature to allow users to write notes for a sequence. These ideas were implemented to enhance the user experience. Overall, five out of six participants confirmed that they would utilize this application to either plan a lesson or practice their dance skills.

7 CONCLUSIONS AND FUTURE WORK

In this paper, we presented a design solution for an application to provide a dynamic approach to teach Salsa. A user centered design approach was adopted and a final product was evaluated with dance students and dance teachers. Ultimately, the application met all the requirements. The implementation of a dance dictionary attained a uniform and coherent storage mechanism for Salsa dance moves. An iterative approach to design a context free grammar was used, which was implemented as a mechanism to validate dance sequences. The iterative process was insightful as context-free grammars for this problem domain are novel. The option to design the grammar to iteratively check two moves, instead of an unlimited amount, proved to be a better user experience. The parser was thoroughly tested, using personal and external test cases, to ensure that nearly all sequence options were analysed. The grammar's aim, to control the sequence generation process, was accomplished, but whether or not the context-free grammar had the optimum grammar specification cannot be confirmed. Unfortunately, the grammar was limited to a subset of moves of this dance style and hence, is specified to the problem domain and not extensible. Given the end-users' positive feedback and enthusiasm from the usability test, we can deduce that because there is a scrupulous representation of dance moves and efficient sequence generation method, this application may be a useful and impactful tool for the dance community.

It would be advantageous to track the progress of a learner's dance skills over a chosen duration whilst they are using the Salsational application. This can be compared to the progress of a learner who is using videos to develop their skills. Hence, the overall effect of the application, on a user's learning experience, can be closely controlled to deem accurate results. In addition, a feature to allow teachers to define their own moves can initiate a personal dance syllabus.

Further research can be conducted into the grammar specification for the parser. Currently, the grammar is designated for the Salsa Dictionary notation. A generic grammar may be developed to allow different dance notations to be utilized by a parser for dance sequence generation, given their constraints.

The application could have a network server implemented, which would allow teachers to upload predefined dance sequences, to be practiced by their students, as well as additional resources. This feature would also allow teachers to notify their students when lessons and events are scheduled for. It also offers a more personalized approach to dance education. This application can be deployed to dance schools throughout Cape Town, to facilitate resource sharing, uniformity and a cooperative dance community.

With the advancement of technology and its increasing use, it has become apparent how it can be used to enhance the daily activities in many industries, such as education. The utilization of technologies has given rise to momentous effects for both students and teachers. These effects have induced the development of various online platforms to motivate and support learners. This notion can be extended to other fields of education besides the traditional educational system, like dance. Therefore, the continual advancement of the Salsational application is urged. We hope that this application may be the foundation for a new generation of e-learning to exploit technology to change educational practices, the way students learn and to empower them at each stage of dance education.

ACKNOWLEDGMENTS

I would like to thank my project partners, Jordy Chetty and Micara Marajh, for their dedication, advice and co-operation throughout the project duration. Without their guidance and good-natured amiability I would have not been able to complete the project. My sincere thanks and appreciation are extended to my project supervisor, Professor Maria Keet, for her constant support, valuable input and encouragement. I would also like to extend this appreciation to the second reader, Professor Deshen Moodley, for his time and feedback. Thanks goes to Tracey Cable, a Salsa dance teacher at EDC, for participating in my evaluation. Finally, I would like to express my gratitude towards our external expert, Angus Prince, for being so gracious and eager to participate in our study and collaborate with us and for allowing us to use the Evolution Dance Company space and to gain access to his students for the evaluation.

REFERENCES

[1] [n. d.]. Evolution Dance Company. <https://www.evolutiondance.co.za/>. Accessed: 2019-08-30.

[2] [n. d.]. Learn Salsa. <https://play.google.com/store/apps/details?id=top.appslaborator.learn.salsa&hl=en/>. Accessed: 2019-08-18.

[3] [n. d.]. Salsa Anywhere. <http://salsa-anywhere.com/>. Accessed: 2019-08-17.

[4] [n. d.]. Salsa Dance: Origin, History Steps. <https://study.com/academy/lesson/salsa-dance-origin-history-steps.html>. Accessed: 2019-08-11.

[5] [n. d.]. Salsa Dancing. <https://pocket-salsa.soft112.com/>. Accessed: 2019-08-19.

[6] [n. d.]. SalsaGente: Cuban Style Salsa. <https://www.salsagente.com/history-of-salsa-music-dance/>. Accessed: 2019-08-15.

[7] [n. d.]. SalsasGood. http://www.salsaisgood.com/dictionary/dictionary_Comb.htm. Accessed: 2019-08-30.

[8] 2012. Incognito Dance. <https://www.incognitodance.com/what-is-salsa/>. Accessed: 2019-08-11.

[9] Beth Adelson, Susan Dumais, and Judith Olson (Eds.). 1994. *CHI '94: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, New York, NY, USA. 608940.

[10] AR Al-Ali and M AL-Rousan. 2004-05. Java-based home automation system. *IEEE transactions on consumer electronics* 50, 2 (2004-05), 498,504.

[11] Thomas Bohm. 2014. 100 things every designer needs to know about people by Susan Weinschenk. *Information Design Journal* 21, 1 (2014), 67–71. <https://doi.org/10.1075/idj.21.1.08boh>

[12] Huang Chiu-Ping. 2010. Exploring factors affecting the user of oral communication. *LongHua Technology University Journal* 30 (2010), 85–104.

[13] Trevarthen Colwyn and N. Malloch Stephen. 2000. The Dance of Wellbeing: Defining the Musical Therapeutic Effect. *Nordisk Tidsskrift for Musikterapi* 9, 2 (2000), 73–88.

[14] M Franko. 2005. Labanotation for Design of Movement-based Interaction. *In Proceedings of the Second Australian Conference on Interactive Entertainment 5* (2005), 113–120.

[15] M Franko. 2005. Writing for the body: Notation, Reconstruction and Reinvention in Dance. *Common Knowledge* 28 (2005), 301–309.

[16] M Franko. 2007. Modeling the Dance Video Annotations. *International conference of Digital Information Management 1* (2007).

[17] M Franko. 2011. Writing for the body: Notation, Reconstruction and Reinvention in Dance. *Common Knowledge* 17,2 (2011), 321–334.

[18] A.H Guest. 1990. Dance Notation. *Perspecta* 26 (1990), 203.

[19] Hao Guo, Miao Zhenjiang, Feiyue ZHu, Gang Zhang, and Song Li. 2014. Automatic Labanotation Generation Based on Human Motion Capture Data. *In Pattern Recognition* (2014), 426–435.

[20] Jonathan Hatol. 2006. MovementXML: A representation of semantics of human movement based on Labanotation. *Ph.D. Dissertation, School of Interactive Arts and Technology* (2006).

[21] Michael T. Helmick. 2007. Interface-based Programming Assignments and Automatic Grading of Java Programs. *SIGCSE Bull.* 39, 3 (June 2007), 63–67. <https://doi.org/10.1145/1269900.1268805>

[22] Sahereh Hosseinpour, Mir Mohammad Reza Milani, and Huseyin Pehlivan. 2018. A Step-by-Step Solution Methodology for Mathematical Expressions. *Symmetry* 10,7 (2018), 285.

[23] Tanya Karen. [n. d.]. Could taking notes improve your dance move recall and creativity? <http://socialdancecommunity.com/could-taking-notes-improve-your-dance-move-recall-creativity/>. Accessed: 2019-08-20.

[24] Vicky Karkou, Sophia Bakogianni, and Evageline Kavkli. 2008. Traditional dance, pedagogy and technology: an overview of the WebDANCE project. *Research in Dance Education* 9,2 (2008), 163–186.

[25] Matthias Keller and Martin Nussbaumer. 2009. Cascading Style Sheets: A Novel Approach Towards Productive Styling with Today's Standards. *In Proceedings of the 18th International Conference on World Wide Web (WWW '09)*. ACM, 1161–1162. <https://doi.org/10.1145/1526709.1526907>

[26] V Kodaganallur. 2004-07. Incorporating language processing into Java applications: a JavaCC tutorial. *IEEE software*. 21, 4 (2004-07), 70,77.

[27] K Kojima, K Hachimura, and M Nakamura. 2002. LabanEditor: Graphical editor for dance notation. *In Proceedings of International Workshop on Robot and Human Interactive Communication 11* (2002), 59–64.

[28] E. V. Kortright. 1997. Modeling and simulation with UML and Java. *In Proceedings of 1997 SCS Simulation Multiconference*. 43–48. <https://doi.org/10.1109/SIMSYM.1997.586477>

[29] I Scott MacKenzie. 1992-03. Fitts' Law as a Research and Design Tool in Human-Computer Interaction. *Human-computer interaction* 7, 1 (1992-03), 91,139.

[30] N Magnemat-Thalman, D Protopsaltou, and E Kavakli. 2007. Learning How to Dance using a Web 3D Platform. *Lecture notes in Computer Science advances in Web Based learning* (2007), 1–12.

[31] Simon Morris. 2009. *JavaFX in Action* (1st ed.). Manning Publications Co., Greenwich, CT, USA.

[32] Ken Pierce. 2002. Choreographic Structure in Dances by Feuillet. *In Proceedings of the Twenty-Fifth Annual Conference of the Society of Dance History Scholars* (2002), 96–106.

[33] J. Porub dn, M. Forg a d, and M. Sabo. 2009. Annotation based parser generator. *In 2009 International Multiconference on Computer Science and Information Technology*. 707–714. <https://doi.org/10.1109/IMCSIT.2009.5352763>

[34] K.E Raheb and Y Ioannidis. 2012. A Labanotation Based Ontology for Representing Dance Movement. *Gesture and Sign Language in Human-Computer Interaction and Embodied Communication Lecture Notes in Computer Science* (2012), 106–117.

[35] Kirk Reinholtz. 2000. Java Will Be Faster Than C++. *SIGPLAN Not.* 35, 2 (Feb. 2000), 25–28. <https://doi.org/10.1145/345105.352548>

[36] Meredith Ritter and Kathryn Graff. 1996. Effects of dance/movement therapy: A meta-analysis. *The Arts in Psychotherapy* 23,3 (1996), 249–260.

[37] Vivek Sinha, Frederic Doucet, Chuck Siska, Rajesh Gupta, Stan Liao, and Abhijit Ghosh. 2000. YAML: A Tool for Hardware Design Visualization and Capture. *In Proceedings of the 13th International Symposium on System Synthesis (ISSS '00)*. IEEE Computer Society, 9–14. <http://dl.acm.org/citation.cfm?id=501790.501793>

[38] M. Thomas and F. McGarry. 1994. Top-down vs. bottom-up process improvement. *IEEE Software* 11, 4 (July 1994), 12–13. <https://doi.org/10.1109/52.300121>

[39] Christine von Renesse and Volker Ecke. 2011. Mathematics and Salsa Dancing. *Journal of Mathematics and the Arts* 5,1 (2011), 17–28.

[40] E.C Warburton. 2000. The Dance on Paper: The effect of notation-use on learning and development in dance. *Research in Dance Education* 1,2 (2000), 193–213.

[41] A. I. Wasserman, P. A. Pircher, and R. J. Muller. 1990. The object-oriented structured design notation for software design representation. *Computer* 23, 3 (March 1990), 50–63. <https://doi.org/10.1109/2.50272>

[42] M. Weir, S. Aggarwal, B. d. Medeiros, and B. Glodek. 2009. Password Cracking Using Probabilistic Context-Free Grammars. In *2009 30th IEEE Symposium on Security and Privacy*. 391–405. <https://doi.org/10.1109/SP.2009.8>

[43] Lars Wilke, Thomas Clavert W., Ronda Ryman, and Illene Fox. 2005. From dance notation to human animation: The LabanDance Project. *Journal of Visualization and Computer Animation* 16 (2005), 201–211.

[44] Deborah Zucker. 2013. The Belmont Report. (2013), 1–3. <https://doi.org/10.1002/0471667196.ess7160>

A DANCE SYLLABUS

The Salsa Dictionary’s representation, dance moves and variations are presented below. The new names for the dance moves are given and the old moves can be accessed from the Salsa Dictionary. All moves are presented in the format according to the matrix in Figure 2.


Term	Definition	Symbol	Thumbnail Click to enlarge
Normal closed hold	Dancers are facing each other; the man positions his right hand on the lady’s left shoulder blade and holds her right with his left with his arm bent. The lady places her left hand on the man’s right shoulder.	N	

Figure 6: Salsa Dictionary’s representation of an element

A.1 Basic Step

c
f → ← l
Basic back

Table 5: Variation 1

c
f → ← l
Basic back

Table 6: Variation 2

c
l → ← f
Basic front

Table 7: Variation 3

c
l → ← f
Basic back

Table 8: Variation 4

A.1.1 Basic Step with crossed hold.

n
f → ← l
Basic back

Table 9: Variation 1

n
f → ← l
Basic back

Table 10: Variation 2

n
l → ← f
Basic front

Table 11: Variation 3

n
l → ← f
Basic back

Table 12: Variation 4

A.1.2 Basic Step with normal open hold.

n	*c*
f → ← l	f → ← l
XHands	

Table 13: Variation 1

n	*c*
f → ← l	f → ← l
XHands	

Table 14: Variation 2

c	*n*
l → ← f	l → ← f
XHands	

Table 15: Variation 3

n	*c*
l → ← f	l → ← f
XHands	

Table 16: Variation 4

A.1.3 Basic Step with change in hand hold.

A.2 Crossed body lead

n	*n*
f → ← l	l → ← f
XBL	

Table 17: Variation 1

n	*n*
l → ← f	f → ← l
XBL	

Table 18: Variation 2

c	*c*
l → ← f	f → ← f
XBL	

Table 19: Variation 3

c	*c*
f → ← l	l → ← f
XBL	

Table 20: Variation 4

A.2.1 Standard cross body lead.

n	*n*
f → ← l	l → ← f
XBL	
l@	

Table 21: Variation 1

n	*n*
l → ← f	f → ← l
XBL	
@l	

Table 22: Variation 2

c	*c*
l → ← f	f → ← f
XBL	
l@	

Table 23: Variation 3

c	*c*
f → ← l	l → ← f
XBL	
@l	

Table 24: Variation 4

A.2.2 Cross body leads with turns.

A.3 Spot turn

c
f→ ←l
1@

Table 25: Variation 1

c
f→ ←l
@1

Table 26: Variation 2

c
l→ ←f
1@

Table 27: Variation 3

c
l→ ←f
@1

Table 28: Variation 4

c
f→ ←l
E*

Table 33: Variation 1

c
f→ ←l
*E

Table 34: Variation 2

c
f→ ←l
E*

Table 35: Variation 3

c
f→ ←l
*E

Table 36: Variation 4

A.3.1 Turns with crossed hands.

c
l→ ←f
*E

Table 37: Variation 5

c
l→ ←f
E*

Table 38: Variation 6

n
f→ ←l
1@

Table 29: Variation 1

n
f→ ←l
@1

Table 30: Variation 2

c
l→ ←f
*E

Table 39: Variation 7

c
l→ ←f
E*

Table 40: Variation 8

n
l→ ←f
1@

Table 31: Variation 3

n
l→ ←f
@1

Table 32: Variation 4

A.3.2 Turns with normal open hands.

A.4 Comb

A.4.1 Comb with crossed hands.

n
f → ← l
E*

Table 41: Variation 1

n
f → ← l
*E

Table 42: Variation 2

n
f → ← l
E*

Table 43: Variation 3

n
f → ← l
*E

Table 44: Variation 4

n
l → ← f
*E

Table 45: Variation 5

n
l → ← f
E*

Table 46: Variation 6

n
l → ← f
*E

Table 47: Variation 7

n
l → ← f
E*

Table 48: Variation 8

A.4.2 Comb with normal open hands.

B FILE SPECIFICATIONS

B.1 Dance Move File

This file is loaded when the application starts in order to instantiate the dance move objects. The YAML file for the dance moves are specified as follows:

- id: the identification of the move type
- name: the name of the dance move
- explanation: a description about the technicalities of the move
- danceMove: an arraylist of step objects, which are an arraylist of element objects
 - symbol: a symbol denoting an element property of a dance move object
 - explanation: a description of the elemental property

B.2 Sequence File

The sequence file stores instances of sequences in the following format:

- name: the name of the sequence
- dance sequence: an arraylist of dance move objects as specified above

C IMPLEMENTED GRAMMAR

Below is a segment of code from the .jj file where the CFG was implemented.

```

TOKEN: { "(" | ")" | "|" | ")" | "(" | "+" | "*" | "<FOLLOWER: ([\"f\"])+> |
<LEADER: ([\"l\"])+> | <CROSSED: ([\"c\"])+> | <OPEN: ([\"n\"])+> }
void S(): (Y()Y()Z()Z()) (A()B()) <EOF>
void Z(): <OPEN> X()
void Y(): <CROSSED> X()
void X(): "+" | "*"
void A(): C() (K()C()K())L()C()L()O()C()O()P()C()P()
void B(): D() (K()D()K())L()D()L()O()D()O()P()D()P()
void C(): <FOLLOWER> <LEADER>
void D(): <LEADER> <FOLLOWER>
void K(): "("
void L(): ")"
void O(): ")"
void P(): ")"

```

D SEQUENCE GENERATION COMPONENT

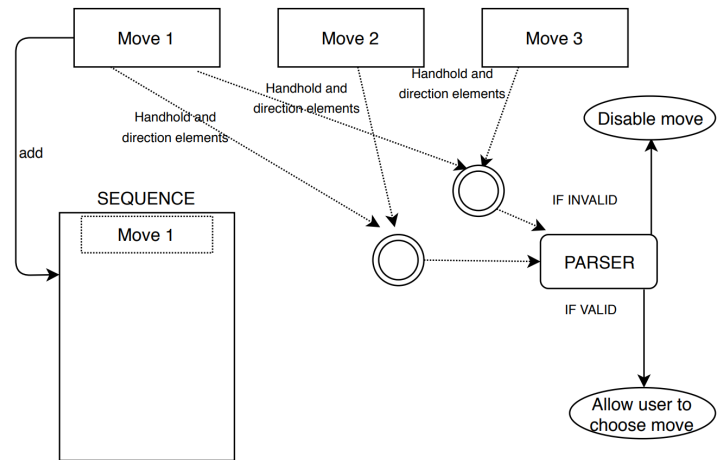


Figure 7: Process for Sequence Generation

E USER INTERFACE

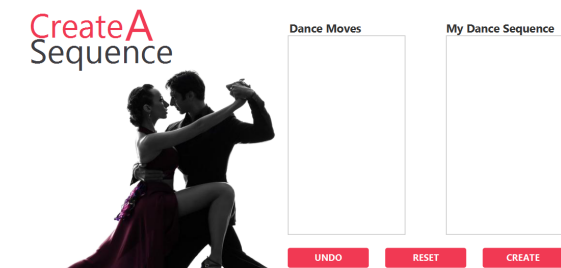


Figure 8: Interface for creating a sequence

Welcome to Salsational

Username: _____
Password: _____
Name: _____
Surname: _____
E-mail: _____

SIGN UP [Already a member?](#)



Figure 9: Interface for creating a new user

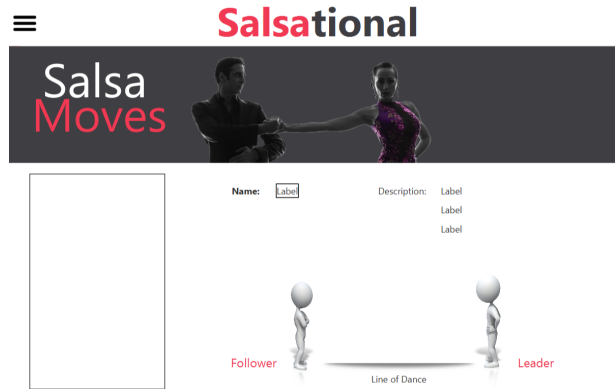


Figure 10: Dashboard interface with the view moves options displayed