MOVEMENTXML: A REPRESENTATION OF SEMANTICS OF HUMAN MOVEMENT BASED ON LABANOTATION

by

Jonathan Hatol
BS Computer Science, Ateneo de Manila University, 1997

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE
in the School of
Interactive Arts and Technology

© Jonathan Hatol 2006
SIMON FRASER UNIVERSITY
Summer 2006

All rights reserved. This work may not be reproduced in whole or in part, by photocopy or other means, without the permission of the author.
APPROVAL

Name: Jonathan Hatala
Degree: Master of Science
Title of thesis: MovementXML: A Representation of Semantics of Human Movement Based on Labanotation

Examining Committee:

________________________________________
Dr. Marek Hatala, Chair

________________________________________
Dr. Vive Kumar, Senior Supervisor

________________________________________
Dr. Tom Calvert, Supervisor

________________________________________
Dr. Susan Kozel, External Examiner

Date Approved: May 26, 2006
DECLARATION OF
PARTIAL COPYRIGHT LICENCE

The author, whose copyright is declared on the title page of this work, has granted to Simon Fraser University the right to lend this thesis, project or extended essay to users of the Simon Fraser University Library, and to make partial or single copies only for such users or in response to a request from the library of any other university, or other educational institution, on its own behalf or for one of its users.

The author has further granted permission to Simon Fraser University to keep or make a digital copy for use in its circulating collection, and, without changing the content, to translate the thesis/project or extended essays, if technically possible, to any medium or format for the purpose of preservation of the digital work.

The author has further agreed that permission for multiple copying of this work for scholarly purposes may be granted by either the author or the Dean of Graduate Studies.

It is understood that copying or publication of this work for financial gain shall not be allowed without the author’s written permission.

Permission for public performance, or limited permission for private scholarly use, of any multimedia materials forming part of this work, may have been granted by the author. This information may be found on the separately catalogued multimedia material and in the signed Partial Copyright Licence.

The original Partial Copyright Licence attesting to these terms, and signed by this author, may be found in the original bound copy of this work, retained in the Simon Fraser University Archive.

Simon Fraser University Library
Burnaby, BC, Canada
Abstract

Most of us are familiar with music notations. Dance notations, on the other hand, have not gained widespread acceptance. This can be attributed to the fact that dance notations, and human movement notations in general, are inherently complex. This complexity has also hindered researches in the field. Up to now, there is no de facto standard for dance score interchange. Labanotation is the most prevalent among the different human movement notations. Most technological advances so far have revolved around displaying and editing Labanotation scores using the computer. Applications such as the LabanWriter store the dance scores in a graphical format that is neither suitable for interpretation nor analysis. LabanXML, the latest advancement in the representation of Labanotation, falls short of describing the more complex aspects of Labanotation. This study uses an object-oriented approach to come up with an XML-based representation that captures the semantics of Labanotation.
To my beloved Amma.
“Pater Vincent, haben Sie Mut zur Lücke.”

(“My dear Father Vincent, do have the courage to leave some gaps.”)

— Pope Benedict XVI
Acknowledgments

In developing an academic undertaking that attempts to encapsulate the semantics of human movement, I owe a debt of gratitude to several people who made this thesis possible.

First. To my family for their unwavering support, and especially to my father who constantly prodded me to complete my thesis within the best possible time.

Second. To my grandparents who passed on their insights and inspiration over the years. I express an equally deep sorrow for the passing away of my grandmother who is unable to share my success in the completion of this humble work.

Third. To Emmy Jorda and my sister Jeannette, who generously shared their time to proof-read these pages. To Sheila Tan, for spending countless days and nights helping me understand Labanotation.

Fourth. To Rhonda Ryman, who in no small way imparted her expertise in Labanotation, and whose intelligent observations significantly aided the revision this thesis.

I appreciate my Examining Committee, for the invaluable role which they played, in seeing that this thesis finds fruition.

My Senior Supervisor, Vive Kumar, for extending the kind of sustaining patience that allowed me to meet the varied challenges of the whole process.

My Supervisor, Tom Calvert, for leading me to the doors of Simon Fraser University. Mr. Calvert largely influenced my choice of topic for this research.

My External Examiner, Susan Kozel, who espoused suggestions during my defense which paved the way for important revisions.

To the Chair, Marek Hatala, whose keen suggestions allowed the inclusion of worthy points in the revision of this thesis.

I am humbled by the outpouring of support. Indeed, it has been said: If the only prayer in one’s life is a mere ‘thank you,’ then, it would suffice.
# Contents

Approval ii

Abstract iii

Dedication iv

Quotation v

Acknowledgments vi

Contents vii

List of Tables xii

List of Figures xiii

List of Programs xv

## 1 Introduction 1

1.1 Background ...................................... 1

1.2 Statement of the Problem ............................ 2

1.2.1 The Subproblems .................................. 2

1.3 The Hypotheses .................................... 2

1.4 Limitations of the Study ............................. 2

1.5 Definitions of Terms ................................ 3

1.6 Assumptions .................................... 3

1.7 The Importance of the Study ......................... 3

vii
2 Related Literature

2.1 Labanotation ........................................ 5

2.1.1 Other Movement Notations .................... 5

2.2 LabanWriter ........................................ 6

2.2.1 LabanReader ................................ 6

2.3 LabanEditor ........................................ 7

2.4 LabanXML .......................................... 7

2.5 MusicXML ......................................... 7

2.6 Smoliar's Representation ....................... 8

2.7 LabanDancer ...................................... 8

2.8 XML ........................................... 8

2.9 Semantic Representation ......................... 9

3 The Representation .................................. 10

3.1 Score ............................................... 10

3.1.1 Authors ....................................... 11

3.1.2 Categories ..................................... 12

3.1.3 Score Components .............................. 12

3.2 Stage ................................................ 12

3.2.1 Stage Objects .................................. 13

3.2.2 Performers .................................... 14

3.2.3 Stage Props ................................... 15

3.3 Defined Movements ................................. 15

3.3.1 Body Wave .................................... 15

3.3.2 Define Movement ............................... 16

3.4 Staff .............................................. 16

3.4.1 Staff Segment .................................. 17

3.4.2 Measure ....................................... 18

3.4.3 Base Measure .................................. 18

3.4.4 Repeat ......................................... 19

3.5 Channels ........................................... 19

3.5.1 Support Channel ............................... 20

3.5.2 Gesture Channel ............................... 20
3.10.2 Body Part Range ........................................... 38
3.10.3 Body Part Surface ......................................... 38
3.10.4 Phrase ....................................................... 39
3.11 State Channel Elements ...................................... 39
3.11.1 Cross of Axes ............................................. 39
3.11.2 Focal Point ............................................... 40
3.12 Location Channel Elements .................................. 41
3.12.1 Location .................................................... 41
3.12.2 Orientation ............................................... 42
3.12.3 Path ........................................................ 42

4 The Tinikling Project ......................................... 44

5 Analysis .......................................................... 46
5.1 MovementXML vs. Existing Formats .......................... 46
5.2 A Semantic Representation of Human Movement .......... 48
5.3 Tinikling ........................................................ 49
5.4 MovementXML vs. Labanotation ............................. 49
5.4.1 Channels vs. Columns ...................................... 50
5.4.2 Complex Channel Advantage ............................. 51
5.4.3 Intermediate Movements ................................. 51
5.4.4 Simultaneous Movements ................................. 52
5.4.5 Jumps ......................................................... 52
5.4.6 Center of Gravity .......................................... 52
5.4.7 Inclusion Bow .............................................. 52
5.4.8 Addition Bow .............................................. 52
5.4.9 3rd Dimension ............................................. 53

6 Conclusion and Recommendations ......................... 54
6.1 Future recommendations ..................................... 54
6.1.1 Extending MovementXML ................................. 54
6.1.2 Finish Tinikling Editor ................................... 54
6.1.3 Dance Ontology ............................................ 55
A MovementXML Schema 56
B Examples 71
Bibliography 74
Index 76
List of Tables

3.1 The stage object types ........................................ 14
3.2 The **beat-duration** values of the different note types ............ 18
3.3 The common direction elements and their equivalent values. ........ 30
3.4 The different level elements and their equivalent values. ........... 30
3.5 Degree values and their meaning. ................................ 30
3.6 The common body part elements and their equivalent **body-part** representations. ........................................ 37

5.1 MovementXML feature comparison ............................... 46
List of Figures

3.1 A sample score with properties ........................................... 11
3.2 A stage 30 feet high and 20 feet deep. ................................ 13
3.3 Stage Object Hierarchy ...................................................... 13
3.4 Defining a male performer “A” ............................................. 15
3.5 Defining a chair and a table. .............................................. 16
3.6 The staff structure .............................................................. 17
3.7 Staff segment class hierarchy .............................................. 17
3.8 A 3/4 measure showing the different channels ......................... 19
3.9 A movement that involves the left hand. ................................ 22
3.10 The right arm moving forward-high towards a final spot. ........... 23
3.11 An aerial turn. ................................................................. 24
3.12 A 360 degree turn in the air ............................................... 26
3.13 Position right foot to the front of the body. ......................... 28
3.14 Position right foot in front of the left foot. ......................... 28
3.15 The head turns clockwise towards the focal point. ................ 31
3.16 Move the left foot forward 1 meter .................................. 32
3.17 The right index finger pointing to performer B. ..................... 34
3.18 Both hands approach each other. ...................................... 35
3.19 The full heel touching the floor. ....................................... 36
3.20 The right knee is supported by the bench. ......................... 36
3.21 A body part range from head to toe. ................................ 38
3.22 The palm of the right hand. ............................................. 39
3.23 Sets the focal point to a stage prop named “Chair1” ............... 41
3.24 The right index finger pointing to “Chair1” previously defined in Fig. 3.23. 41
3.25 Set the location to the center of the stage floor. ...................... 41
3.26 The left side of the body faces the focal point. ....................... 42
3.27 Make a very big full circle moving forward. ......................... 43

5.1 Modifying the direction of a movement. ............................... 50

B.1 Repeat example resulting in taking 8 steps to complete a circle. .... 72
B.2 Repeat example resulting in taking 4 steps to complete a circle; 2 circles are completed. .................................................. 73
List of Programs

2.1 LabanDancer [22] .................................................. 9
4.1 Tinikling Dance Editor ............................................. 45
6.1 A simple dance ontology using Protégé. ....................... 55
Chapter 1

Introduction

1.1 Background

The most popular way of recording human movement electronically is through key framed 3D animation. One might wonder why we even need to bother with movement notations. The truth is, while key framed 3D animations can record more precise movements using Euler angles or quaternions, they are only good for playback. This type of representation is impractical for converting to diagrams on paper. In any comprehensive system, information must be abstracted and put into symbols [15]. These symbols can provide more meaning than Euler angles or quaternions.

Labanotation is the most prevalent among the different notations. Other notations such as Benesh are well suited for certain types of dance only. Labanotation is general and flexible enough to represent any type of movement which makes it a good choice as a basis for a movement representation.

There are already several Labanotation representations available. However, most of them store their representations in a graphical format. These formats are sufficient for saving the graphical layout of the score but these formats have very little use for other types of applications such as playing back the score as a 3D animation. LabanXML tries to solve this problem by storing the semantics of the movements themselves. But it falls short of describing the more complex details of the movements. It has elements and attributes that describe the objects on the page rather than what those objects mean.
CHAPTER 1. INTRODUCTION

1.2 Statement of the Problem

This researcher proposes to analyze and identify the semantics behind the different Labanotation symbols and come up with a highly structured object-oriented semantic representation that is flexible enough to capture any well-formed Labanotation score.

1.2.1 The Subproblems

1. The first subproblem is to determine whether Labanotation as a movement language is robust enough to be captured using objects.

2. The second subproblem is to determine if an object-oriented approach is a viable representation of Labanotation by providing a proof of concept.

3. The third subproblem is to analyze and evaluate the resulting semantic representation in terms of its advantages and disadvantages over the existing representations.

1.3 The Hypotheses

1. The first hypothesis is that Labanotation as a movement language is robust enough for its semantics to be captured using objects.

2. The second hypothesis is that a Labanotation editor can be created as a proof of concept that the object-oriented approach is a viable representation.

3. The third hypothesis is that the resulting representation’s advantages over the existing representations will outweigh its disadvantages.

1.4 Limitations of the Study

1. This study will be limited to the Labanotation as described in the 3rd edition of Ann Hutchinson Guest’s “Labanotation: The System of Analyzing and Recording Movement.”

2. This study will not evaluate the search capabilities of the resulting MovementXML format.
3. This study will not perform analysis on the resulting semantic representation of movement.

4. The Tinikling Project is only intended as a proof of concept of what an intelligent Labanotation editor can be.

1.5 Definitions of Terms

Notator The person who creates the dance scores.

Semantics pertains to the meaning of the Labanotation symbols in terms of the movements that they represent.

Eclipse Eclipse is an open source Java-based IDE developed by IBM.¹

1.6 Assumptions

1. Labanotation will continue to be the most prevalent movement notation.

2. Dance notators will continue to notate. There will always be a need to preserve or share dances.

3. XML will continue to be the standard form of data interchange.

1.7 The Importance of the Study

Coming up with a semantic representation for human movement is important for many reasons:

1. It allows the preservation of dance scores. The current formats can also preserve the scores but most of them just preserve the graphical layout;

2. It allows the dance score to be shared among different applications. These applications can include dance and movement editors, computer animation, and a semantic web or ontology;

¹http://www.eclipse.org/
3. It allows dance scores to be searched. The XML representation has a very predictable structure which enables it to be searched easily; and

4. It allows for motion and choreographic analysis. The semantic nature of the representation lends itself to higher-level analysis.
Chapter 2

Related Literature

2.1 Labanotation

Laban movement notation was formulated by Rudolf von Laban (1879-1958). Several incarnations of Laban movement notation have sprung up over the years. One such incarnation is the Labanotation (also known as Kinetography Laban) by Ann Hutchinson Guest. Labanotation is a system of analysing and recording of human movement [15]. In Labanotation, it is possible to record every kind of human motion [13].

Labanotation uses a staff similar to that of the music notation. It consists of three lines and runs vertically, rather than horizontal lines; and the score is read from bottom to top, instead of left to right. The advantage of this type of setup is that anything that happens on the left side of the body is written on the left side of the staff, and vice versa [13].

The choice of Labanotation as the basis for this research's movement representation hinges on the fact that it is the most prevalent among the different dance notations. Labanotation is widely accepted as a readable and understandable notation [19], and it is general enough to cover all possible movements in different dance genres.

2.1.1 Other Movement Notations

There are a number of movement notations aside from Labanotation. The ones that are more commonly used are:

Motif Description - Rhonda Ryman described Motif as a movement analysis that "looks at the general movement themes (motifs) and choreographic intentions, apart from
their specific physical realization." Motif describes movements in a high level leaving most of the details to the performer;

**Benesh Movement Notation** - which is used mainly in ballet. This iconographic notation was published by Joan and Rudolf Benesh in the late 1956 [16]. Iconographic systems are essentially stick-based figures [21];

**Sutton DanceWriting and MovementWriting** - these are also iconographic notations which use stick figures and looks very similar to the timeline of a key framed animation. This notation was developed by Valerie Sutton in 1972; and


### 2.2 LabanWriter

LabanWriter is the most widely used Labanotation editor as of this writing. The program was developed at the Dance Department of the Ohio State University by George Karl, Scott Sutherland, and David Ralley under the direction of Lucy Venable [9]. The current version of LabanWriter can only run on Macintosh-based computers.

LabanWriter plays a very important role in the preservation of dance [17]. It is a very powerful layout tool that allows users to create Labanotation scores with ease. However, LabanWriter treats the Labanotation symbols strictly as 2-dimensional graphical objects which limits its usability beyond laying out the scores. This limitation of LabanWriter and its file format inspired the creation of LabanXML by Minako Nakamura as well as the MovementXML.

#### 2.2.1 LabanReader

LabanReader is another application developed at the Dance Department of the Ohio State University by Sheila Marion and A. William Smith. LabanReader is able to read in files created by LabanWriter 4.0 and can be used to focus on particular aspects of movement for teaching or to display various kinds of patterning for movement analysis [9].
CHAPTER 2. RELATED LITERATURE

2.3 LabanEditor

LabanEditor was created by Minako Nakamura, Kozaburo Hachimura, and Kazuya Kojima. The original version of LabanEditor came into fruition because of the need to process dance scores in order to turn them into 3D animation [18].

The second edition of LabanEditor saves dance scores in a new format called LabanXML [19]. It is also able to produce a 3D animation of the scores it created. Very little is known about the editor aside from the few papers written on it by its creators. LabanEditor is also not available for download.

2.4 LabanXML

LabanXML was developed by Minako Nakamura and Kozaburo Hachimura [19]. It is the latest attempt at creating a semantic representation for human movement. LabanXML addresses the shortcomings of the prevalent Labanotation editors: LabanWriter[3], Calaban[1], and Labanatory[2]. These editors store the dance score in a graphical format, storing the symbols as graphical objects with x and y coordinates. The problem with such an approach is that, while it is sufficient for viewing or printing scores, the score itself does not lend itself to movement analysis.

While LabanXML is a big improvement over its predecessors, investigation on its DTD (Data Type Definition) reveals that it is not flexible enough and it falls short being a semantic representation. It has elements such as rpin, hc, hook, and v1 which represent graphical symbols on the page rather than what the symbols mean.

2.5 MusicXML

MusicXML is a music interchange format developed by Michael Good, the founder and CEO of Recordare [4]. Minako Nakamura and her colleagues decided to make compatibility with MusicXML a requirement of LabanXML since dance is usually accompanied by music [19]. In particular, LabanXML borrowed its time structure from MusicXML.

MusicXML has two different ways to describe a score based on its root element:

- **score-partwise** - composed of parts and each part is made up of measures; and
CHAPTER 2. RELATED LITERATURE

- score-time wise - composed of measures where each measure is made up of parts [5].

A closer examination on LabanXML’s DTD (Data Type Definition) reveals that it is using the score-time wise approach.

MusicXML uses a cursor-type mechanism to control the time of the musical notes. The notes have a duration but they don’t have a start time. Every time a note is added, the duration of the note is added to the current time. MusicXML provides the forward and backup elements to allow skipping or the creation of simultaneous notes. This type of time mechanism is not suitable for movement notations since simultaneous actions occur more frequently in human movement.

2.6 Smoliar’s Representation

Stephen Smoliar introduced a data structure in 1978 that can hold the Labanotation scores [21][20]. His representation can capture most of the basic movements and is even more exhaustive than the more recent LabanXML. Smoliar’s representation, however, is concentrated on small movements and does not discuss much about measures, repetitions, and the floor plan.

2.7 LabanDancer

LabanDancer is a nifty tool created by Lars Wilke of Credo Interactive in collaboration with Tom Calvert, Ilene Fox, and Rhonda Ryman [10]. This application can translate a LabanWriter score into 3D animation (Fig. 2.1). The creators of LabanDancer realized that notating a dance is a specialized skill not possessed by everyone in the dance community [9]. To use a dance notation system such as Labanotation requires a thorough understanding in movement analysis and an expert knowledge of its symbolic vocabulary [10]. Creating a tool to visualize a dance score makes the notation more accessible to choreographers, dancers, and dance students [9].

2.8 XML

XML or Extensible Markup Language is a W3C (World Wide Web Consortium) standard for data interchange. It is a simple dialect of SGML (Standard Generalized Markup Language)
which allows documents to be served, received, and processed [8]. These are the qualities that we look for in the human movement representation. XML frees us from having to worry about the language syntax, and XML parsers are readily available for producing and consuming XML documents.

2.9 Semantic Representation

The term "semantics" pertains to meaning or the study of meaning as opposed to "syntax."¹ Semantics have to do with the relationships between various signs and symbols and what they represent² whereas syntax is about describing the structure of those signs and symbols.

Applying the definitions to Labanotation, a semantic representation would be a representation of the meaning of the individual symbols and how they relate to one another. Similarly, a description of the symbols as they appear on the page can be construed as a syntactic representation. The LabanWriter representation falls under syntactic representation while LabanXML is somewhere in between.

¹Source: http://www.reference.com/browse/wiki/Semantics
Chapter 3

The Representation

Analyzing Labanotation is similar to doing domain-specific Natural Language Processing. This should come as no surprise since Labanotation is indeed a language, albeit a graphical one. Labanotation is such a rich and flexible language the notator is free to do almost anything he wants. A single movement can be notated in many different ways. It almost seems like a semantic representation may not be possible.

After thoroughly examining the concepts behind the movements, it was determined that Labanotation is after all a very robust language and a semantic representation is indeed possible. In fact, the resulting representation is even more flexible than Labanotation itself. For instance, MovementXML measures directions in degrees and is not limited by Labanotation’s intermediate movements (Ch. 5.4.3). MovementXML employs a hierarchical composition of elements with the score element at its root. This chapter describes the structure of MovementXML and all its elements.

3.1 Score

The score represents a single human movement or dance composition and is the top-most element in the human movement representation.

A score element has the following properties which serve as its metadata (Fig. 3.1):

- title - the title of the composition;
- date-created - the date the composition was created;
- authors - the authors of the composition; and
• categories - the categories or genres the composition belongs to.

These metadata are useful for indexing, categorizing, and searching a library or an ontology of dance scores.

```
<score title="Don Quixote" date-created="2006-02-17">
  <authors>
    <author last-name="Hatol" given-names="Jonathan" />
    <author last-name="Tan" given-names="Rochelle" />
  </authors>
  <categories>
    <category name="ballet" type="dance" />
  </categories>
  ...
</score>
```

Figure 3.1: A sample score with properties

Title

A score has exactly 1 title. The title is a string attribute that identifies the composition.

Date Created

A score has exactly 1 creation date. The `date-created` attribute is specified in the following form “YYYY-MM-DD”\(^1\) where:

- YYYY indicates the year;
- MM indicates the month; and
- DD indicates the day.

3.1.1 Authors

A score has 1 or more authors. The `authors` element is used to group 1 or more `author` elements together.

The `author` element has 2 properties:

\(^1\)Source: [http://www.w3schools.com/schema/schema_dtypes_date.asp](http://www.w3schools.com/schema/schema_dtypes_date.asp)
CHAPTER 3. THE REPRESENTATION

- last-name - the last name of the author; and
- given-names - the given names of the author.

3.1.2 Categories

A score belongs to 1 or more categories. The categories element contains 1 or more category elements. These category elements describe how the score is to be categorized.

The category element has 2 attributes:

- name - the name of the category; and
- type - the type of activity that is being described by the score (e.g. dance, sports, etc.).

3.1.3 Score Components

The score has 3 main components that relate directly to the description of the movements:

- stage - describes the stage itself and the objects that are in the stage, including the performers;
- defined-movements - this is where the frequently used movements or movement patterns are defined (Ch. 3.3); and
- staffs - a collection of staffs that contain the movement elements.

3.2 Stage

The stage is where the performance occurs. A stage has the following properties:

- width - the width of the stage along the x-axis;
- depth - the depth of the stage along the y-axis;
- height - the height of the stage along the z-axis. If unspecified, the default value for the height is assumed to approach positive infinity; and
- unit - the unit used for the dimensions (width and length) of the stage.
3.2.1 Stage Objects

A stage contains several stage objects. These stage objects include:

- the performers;
- the different body parts of the performers;
- the stage props;
- the different parts of the stage including the empty space;
- the floor; and
- a specific location in space.

A stage-object is defined as the base class of all objects on the stage (Fig. 3.3). These stage objects are used as the subjects (noun) of the movements (verbs).

Not all stage objects need to be defined explicitly in the stage element. Objects such as the floor, the stage-part, the body-part, and the body-part-range are already pre-defined and are assumed to be present.

A stage-object has the following attributes that are inherited by its subclasses:
• **name** - the name of the stage object. e.g. If the object is a performer, then this would be the performer's name; and

• **type** - the type of the stage object. It can have one of the following values: floor, stage-part, performer, or body-part. Table 3.1 shows the different stage objects and their type values.

<table>
<thead>
<tr>
<th>Stage Object</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>floor</td>
<td>stage-part</td>
</tr>
<tr>
<td>stage-part</td>
<td>stage-part</td>
</tr>
<tr>
<td>stage-prop</td>
<td>stage-prop</td>
</tr>
<tr>
<td>performer</td>
<td>performer</td>
</tr>
<tr>
<td>body-part</td>
<td>body-part</td>
</tr>
<tr>
<td>body-part-range</td>
<td>body-part</td>
</tr>
</tbody>
</table>

**Table 3.1: The stage object types**

### 3.2.2 Performers

The **performers** container is where the performers are defined. The **define-performer** element is used to describe a performer's name, gender, and the staff associated with it. A performer, represented by the **performer** element, has to be defined before it can be used in the score.

A performer is associated with exactly one staff. A staff, however, can be associated with more than 1 performer. Staffs are discussed in detail in chapter 3.4. Figure 3.4 shows an example of how performers are defined in MovementXML.

The **define-performer** element has the following properties:

• **name** - a string that identifies the performer. This string must be unique among the performers;

• **gender** - can be one of the following values: "male", "female", or "neutral"; and

• **staff** - the name of the staff associated with the performer.

The **define-performer** element also contains a single element, **initial-location**, which takes in a **stage-part**.
CHAPTER 3. THE REPRESENTATION

3.2.3 Stage Props

The stage-prop element represents the theatrical props that are being used during the performance. Stage props must be defined using define-stage-prop before they can be used in the score. A define-stage-prop element has the following attributes:

- name - the string that identifies the stage prop. This name must be unique among all the stage props; and
- prop-type - a prop type describes what the stage prop is - whether it's a chair, a table, a barre, etc.

Stage props also contain an initial-location element that determines the initial location of the stage prop within the stage (Fig. 3.5).

3.3 Defined Movements

Defined movements are a way of putting smaller movements together to form a higher-level movement. These movements are placed in the complex channel. An example of a defined movement is the body wave. The body-wave element is a complex movement that encompasses several channels.

3.3.1 Body Wave

The body wave movement is represented by the body-wave element. This element has a single attribute:
3.3.2 Define Movement

The define-movement element is used to create new defined movements. The way the movements are defined is beyond the scope of this research.

3.4 Staff

The staff is where the movements of the performers are defined with respect to time. The staff is composed of a base measure and a series of measures and repeats arranged according to time (Fig. 3.6). The base measure, measure, and repeat are collectively known as staff segments.

A staff element has a solitary attribute:

- **name** - this name identifies the staff and must be unique among all the staffs. This name is used as the staff attribute of the performer element (Ch. 3.2.2).
3.4.1 Staff Segment

The staff-segment element is the base class for all staff segments. Subclasses of the staff-segment element include measure, base-measure, and repeat. Figure 3.7 shows the class hierarchy of the staff-segment element.
3.4.2 Measure

A measure can be likened to a measure in a musical score. A measure has the following properties:

- **number** - the measure number used for labelling and identifying the measure;
- **beats** - the number of beats in the measure;
- **beat-type** - the beat type or unit (e.g. note, second, etc.);
- **beat-duration** - the duration of a single beat measured in the unit defined in the beat-type. Table 3.2 gives a partial list of the different beat-duration values for the different note types; and
- **meter** - defines the number of beats per minute.

<table>
<thead>
<tr>
<th>Note type</th>
<th>beat-duration value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Whole</td>
<td>1.0</td>
</tr>
<tr>
<td>Half</td>
<td>0.5</td>
</tr>
<tr>
<td>Quarter</td>
<td>0.25</td>
</tr>
<tr>
<td>Eight</td>
<td>0.125</td>
</tr>
</tbody>
</table>

Table 3.2: The beat-duration values of the different note types

The properties of the measure element may appear to belong to the staff element at first. However, in Labanotation, these properties can change from measure to measure. Thus, it makes sense to put these attributes in the measure element rather than the staff element.

A measure is composed of channels (Fig. 3.8). There are 5 different types of channels: support, gesture, position, state, and complex. A measure can have exactly 1 of every channel. These channels are described in detail in chapter 3.5.

3.4.3 Base Measure

The base-measure element is a restriction of the measure element where the beats attribute is always 1. The base measure is a special measure which defines the starting pose, location, and orientation of the performer associated with the staff. A staff must have exactly 1 base measure and it must be defined before any other staff segments in the staff are defined.
3.4.4 Repeat

A repeat element defines a repetition of the staff segments that it contains. A repeat element can contain measure elements and other repeat elements.

A repeat has the following properties:

- **count** - defines the number of times the contained staff segments must be repeated in series;

- **repeat-type** - valid values for this attribute are:
  - "similar" - equivalent to the ad lib sign in Labanotation wherein a freedom of performance is allowed [15];
  - "exact" - indicates an exact performance;
  - "lateral-symmetry" - exchange of left and right;
  - "sagittal-symmetry" - exchange of front and back; and
  - "oppositional-symmetry" - a combination of both lateral and sagittal symmetry. This is most often used in ballroom dancing.[15]

- **exclude-path** - when set to true, this flag indicates that paths should be excluded from the repetition. Figures B.1 and B.2 show how the exclude-path property can affect the outcome of a movement. This property is set to "false" by default.

3.5 Channels

There are 5 types of channels in a measure, each of which serves a specific purpose:
• **support** - for changes in the support;
• **gesture** - for changes in the gesture;
• **location** - for changes in the performer's path, location, and orientation;
• **state** - for changes in the performer's cross of axes and focal point; and
• **complex** - for complex movements that involve a combination of the first four channels.

### 3.5.1 Support Channel

The support channel, represented by the `support` element, concerns itself with the placement of the weight of the body. Movements in the support channel indicate "progressions of the whole body, that is, progressions of the center of gravity of the body by means of transference of weight, jumps, or falls."[15]

**Absence of Movement**

MovementXML follows the same rules as Labanotation with regards to the absence of movement in the support channel. Unless a hold (Ch. 3.7.3) is in place, the absence of movement in the support channel usually indicates that the body part that had the support is being lifted. In the case where the feet had the support, an absence of movement on both feet indicates a jump. However, the support is retained for body parts other than the feet or the knees. For instance, a person who is in a seated position (i.e. supported by both hips) will remain seated even when there are no subsequent movements in the support channel.

**Jumps and Aerial Turns**

Another way to indicate a lift or a jump is to put a `lift` (Ch. 3.7.5) action in the support channel. A lift is necessary when there are movements in the support channel.

### 3.5.2 Gesture Channel

The gesture channel is represented by the `gesture` element. The gesture channel contains movements that don't carry weight.[15] The movements of the limbs are a good example of movements that can be placed in this channel.
CHAPTER 3. THE REPRESENTATION

Absence of Movement

Unless a hold (Ch. 3.7.3) is in place, the absence of movement in the gesture channel for a particular body part indicates that the body part reverts back to its normal state with respect to its parent body part.

3.5.3 Location Channel

The location channel is represented by the location element. The location channel is used in conjunction with the support channel (Ch. 3.5.1) to define the location, orientation, and path of the performers at specific points in time. Actions in this channel have a direct effect on the stage area.

3.5.4 State Channel

The state channel is where the different performer's state variables are being modified. Once set, the state values are in effect until they are replaced by another value. These variables include:

- cross-of-axes - the global system of reference used by the performer (Ch. 3.8.7); and
- focal-point - the stage object that currently has the focal point.

3.5.5 Complex Channel

The complex channel, represented by the complex element, is where movements that encompass multiple channels are placed. Movements such as the body wave and the defined movements (Ch. 3.3) are used in this channel.

3.6 Timed Element

Everything that goes into the different channels are timed elements. These include movements, states, and paths. A timed element, represented by timed-element, has the following properties:
CHAPTER 3. THE REPRESENTATION

- **start** - indicates when the element takes effect. This is measured in the containing measure’s beats. The containing measure is the measure in which the movement is started; and

- **duration** - the time it takes for this element to complete. This is also measured in the containing measure’s beats.

There are 3 general types of timed elements: movements, states, and paths.

### 3.7 Movement

Movements are the building blocks of a score. These are the instructions that tell what a body part should do at a specific point in time. The movement element is the base class for all movement classes. It is divided into 2 parts:

- **parts** - contains one or more areas of the body that are affected by the movement. Compound movements that are analogous to the addition bow can be created by putting more than one body part in the parts component. Figure 3.9 shows how the parts container is used (See Ch. 3.10 for a more detailed description of the different body parts); and

- **description** - contains the descriptors for the movement. These descriptors can be viewed as the properties of the movement that describe how the movement is to be performed. (See Ch. 3.8 for the different types of descriptors.)

```xml
<move start="0" duration="2">
  <parts>
    <hand side="left" />
  </parts>
  ...
</move>
```

Figure 3.9: A movement that involves the left hand.
3.7.1 Move

The move element represents a directional movement. A directional movement takes on a different meaning depending on whether it is placed in the gesture channel or in the support channel. In the gesture channel, a movement is a movement towards a specific point or destination. In the support channel, movements are motion away from a previous point of support [15]. The move element has a path attribute which describes the path taken by the movement. The path attribute can have the following values:

- "relative" - indicates a normal path which can be affected by other movements. This is the default value;
- "start-direction" - indicates an undeviating planar path defined by the starting direction;
- "end-direction" - indicates an undeviating planar path defined by the ending direction; and
- "final-spot" - indicates an undeviating aim towards the final spot determined at the start of the movement.

```
<move start="0" duration="3" path="final-spot">
  <parts>
    <arm side="right" />
  </parts>
  <description>
    <forward />
    <high />
  </description>
</move>
```

Figure 3.10: The right arm moving forward-high towards a final spot.

3.7.2 Turn

A turn represents a turn sign in Labanotation. A turn is understood to be a longitudinal rotation of the specified body part unless specified otherwise using the cross-of-axes descriptor. The turn movement has 1 additional attribute:
• direction - can be "clockwise", "counter-clockwise", or "any".

Turning the whole body can be indicated by placing a turn movement in the support channel. Turns can be combined with lift (Ch. 3.7.5) to indicate an aerial turn (Fig. 3.12).

```
<turn start="0" duration="2" direction="clockwise">
  <parts>
    <leg side="both" />
  </parts>
  <description>
    <direction>90</direction>
  </description>
</turn>

.lift start="0" duration="2" >
  <parts>
    <leg side="both" />
  </parts>
</lift>
```

Figure 3.11: An aerial turn.

Wheel

A wheel is a specialized turn wherein the distal end of the body part creates a circular path. In some cases, wheeling can also be achieved using the turn movement by using either the standard cross of axes or the constant cross of axes (Ch. 3.8.7).

Somersault

A somersault is a rotation on the lateral (side-to-side) axis[15]. The somersault element also has a direction attribute. The valid values for this attribute are:

• "forward" - for forward somersaults;
• "backward" - for backward somersaults; and
• "any" - for any direction
Cartwheel

Cartwheels are rotations on the sagittal (forward-backward) axis[15]. The valid values for its direction attribute are:

- "left" - for a cartwheel towards the left;
- "right" - for a cartwheel towards the right; and
- "any" - for any direction

3.7.3 Hold

A hold movement is equivalent to the hold or retention signs in Labanotation. The hold element has the following attributes:

- **duration** - inherited from timed-element. Use "0" to indicate that the retention is as long as the longest movement that occurs simultaneously with this hold action. If unspecified, the retention is in effect until it is cancelled by another hold or a return-to-normal (Ch. 3.7.4) action;

- **type** - indicates the type of hold. There are 3 different values that are defined in Labanotation:
  - "body" - this is the default value. This describes a retention of the relation between the body part to its base part;
  - "space" - a retention of the previously established direction; and
  - "spot" - a retention of a specific spot in space.

3.7.4 Return to Normal

The return-to-normal movement instructs the body part to return to its normal state. This also cancels any hold or retention that is applied to the body part. Since return-to-normal is more of a command rather than a timed movement, its duration attribute is always set to "0".
3.7.5 Lift

The absence of movement in the support channel usually indicates a lift or a jump. However, there are cases such as aerial turns wherein lifts can occur together with movements in the support channel. In these cases, we need to explicitly use the lift movement to indicate that a body part is to be lifted. An aerial turn is represented by a combination of turns and lifts (Fig. 3.12).

```xml
<support>
  <turn start="0" duration="2" direction="clockwise">
    <parts>
      <foot side="both"/>
    </parts>
    <description>
      <direction>360</direction>
    </description>
  </turns>
  <lift start="0" duration="2">
    <parts>
      <foot side="both"/>
    </parts>
  </lift>
</support>
```

Figure 3.12: A 360 degree turn in the air

3.7.6 Contraction and Extension

Contraction and extension are opposing movements. Contraction is the drawing in of the extremity of the limb towards its base[15] while extension is the lengthening of the body part away from its base. contraction and extension have an additional attribute, axis, which takes in 3 values:

- "longitudinal" - describes a 1-dimensional extension on the longitudinal axis. This is the default value;

---

2 A jump occurs when the feet have the support and there are no movements on both feet.
• "lateral" - describes a 2-dimensional extension along the lateral axis. This type of extension is used to express movements such as spreading of the fingers; and

• "all" - describes a 3-dimensional extension. This type of extension is used to express movements such as making a fist.

collision and extension usually takes in 2 descriptors:

• degree - indicates the extent of the contraction or extension. A contraction can be expressed as an extension with a negative degree value, and vice versa; and

• direction - describes the physical direction of the intermediate joint.

3.7.7 Folding and Unfolding
Folding and unfolding are opposing movements that describe a bending and unbending movement, respectively. Both movements also take in 2 descriptors similar to contraction and extension:

• degree - indicates the extent of the folding or unfolding. A folding can be expressed as an unfolding with a negative degree value, and vice versa; and

• direction - describes the path that the extremity takes.

3.7.8 Shift
A shift is a movement on a straight line[15]. The shift element has the same attributes and accepts the same descriptors as the move element.

3.7.9 Position
The position element is a movement element that represents the position signs or relationship pins in Labanotation. It is used to describe the position of the subject part with respect to a reference object. The position element this attributes:

• duration - set to "0" since position is more of a state rather than an action.

The position element also contains a part and an optional reference object:

• parts - derived from movement. This refers to the parts that are being positioned;
CHAPTER 3. THE REPRESENTATION

- **reference** - this is an optional element. If unspecified, the reference object is assumed to be the center line of the body;

- **description** - derived from movement. This container accepts a direction (Ch. 3.8.1) and a level (Ch. 3.8.2).

```
<position start="1">
  <parts>
    <foot side="right" />
  </parts>
  <description>
    <direction>0</direction>
  </description>
</position>
```

Figure 3.13: Position right foot to the front of the body.

```
<position start="1" direction="0">
  <parts>
    <foot side="right" />
  </parts>
  <description>
    <direction>0</direction>
  </description>
  <reference>
    <foot side="left" />
  </reference>
</position>
```

Figure 3.14: Position right foot in front of the left foot.

3.7.10 Repeat Movement

A **repeat-movement** is a repetition of the movements of a specific body parts within a specified period of time. The **repeat-movement** has the following important attributes:

- **repeat-measure** - the measure count of the repeated movement. This attribute is
optional. If unspecified, the repeat movement just repeats the movements immediately before it;

- **repeat-start** - the starting beat within the measure of the repeated movement. This attribute is also optional and is only valid if repeat-measure is used;

- **duration** - inherited from timed-element. The duration of this movement is also the duration of the repeated movement; and

- **repeat-type** - same as repeat (Ch. 3.4.4).

The body parts that are included in the parts component are the parts whose movements are being repeated. The performer will use the same body parts to repeat the movements even if the parts in the component belong to a different performer.

### 3.7.11 Continuation

A continue element is equivalent to Labanotation's analogy signs. It is also closely related to the repeat-movement element in that it also has the repeat-type attribute.

### 3.8 Descriptors

Descriptors are the measurements or dynamics of a movement. They describe how or to what extent the movements are to be performed. The use of descriptors, as opposed to static properties, is necessary because of the inherent flexibility of Labanotation. For instance, a turn can be a turn towards a specific object rather than a turn that is measured in degrees.

#### 3.8.1 Direction

A direction refers to either the direction of a movement or a position along the horizontal plane. The direction element accepts a float value in degrees between -179.99 to 180. Table 3.3 contains a list of the predefined directions that are commonly used.

#### 3.8.2 Level

A level is the vertical measure of a movement direction or the height of a position in space. The level element accepts a float value that is measured in degrees. If unspecified, the value is assumed to be 0. Table 3.4 contains a list of the predefined levels.
3.8.3 Degree

The degree descriptor is a more subjective way of indicating the extent in which a movement is to be performed. It takes in an integer value from -7 to 7. If unspecified, the value is assumed to be 0, which indicates a normal movement. Table 3.5 shows what the different values mean to different movements.

<table>
<thead>
<tr>
<th>Degree</th>
<th>Movement</th>
<th>Turn</th>
</tr>
</thead>
<tbody>
<tr>
<td>-7</td>
<td>very short, totally flexed</td>
<td>minimal turn</td>
</tr>
<tr>
<td>0</td>
<td>normal movement</td>
<td>follow other turn descriptors</td>
</tr>
<tr>
<td>7</td>
<td>very long, totally stretched</td>
<td>maximum turn</td>
</tr>
</tbody>
</table>

Table 3.5: Degree values and their meaning.

3.8.4 Towards and Away From

The towards and away-from elements are direction descriptors that describe opposing directions. The towards descriptor, as its name suggests, is a movement towards a specified object. The away-from descriptor, on the other hand, is a movement away from the specified
object. Both elements take in a target stage object.

```xml
<turn start="0" duration="2" direction="clockwise">
    <parts>
        <head />
    </parts>
    <description>
        <towards>
            <focal-point />
        </towards>
    </description>
</turn>
```

Figure 3.15: The head turns clockwise towards the focal point.

### 3.8.5 Accent

An accent is the result of sudden and momentary increase in the use of energy[15]. This increase in energy usually results in a sound, as in a stamp or a clap. There are 3 degrees of accent: 1 being the slightest, and 3 being the strongest.

### 3.8.6 Follow

The follow descriptor indicates a passive movement or a movement to a lesser degree. It is synonymous to Labanotation's inclusion bow, half support, and half gesture. The follow descriptor takes in a body part (which can be another performer's body part).

### 3.8.7 Cross of Axes

The cross-of-axes descriptor serves the same purpose as set-cross-of-axes (Ch. 3.11.1) but at a much smaller scale. This change in the cross of axes is only applicable to the movement it is describing and only for the duration of the movement.

### 3.8.8 Distance

The distance descriptor represents an exact measure of distance. This distance is measured in steps by default but it can also be measured in other units. The distance descriptor has 1 attribute called unit which specifies the type of measurement used.
CHAPTER 3. THE REPRESENTATION

3.8.9 Coordinate

The coordinate descriptor represents an exact point in 3D space. The origin is set at the center of the stage floor with $+x$ pointing towards the front of the stage, $+y$ towards the right (or left if viewed from the audience), and $+z$ towards the top. The coordinate descriptor has the following attributes:

- **unit** - the unit used for measurement. If unspecified, it takes in the unit used by the stage element;
- **x** - the x-coordinate;
- **y** - the y-coordinate; and
- **z** - the z-coordinate.

The coordinate descriptor is usually used with the stage-part element.

3.9 Relationship Movements

A relationship is a movement that establishes a relationship between the subject parts and some other parts. The relationship varies in its degree of involvement from addressing to touching. The relationship element has the following attributes:

- **action** - this refers to how the contact is performed. It can have one of 3 values:
  - "normal" - indicates a normal contact. This is the default value;
CHAPTER 3. THE REPRESENTATION

- "surround" - used for grasping or surrounding; and
- "penetrate" - used for penetrating or intertwining fingers.

- **action-part** - the part that is performing the action. The valid values are:
  - "primary" - the primary part is performing the grasping or penetrating;
  - "other" - the other part is performing the action;
  - "both" - both parts are performing the action; and
  - "any" - either part can perform the action. This is the default value.

- **duration** - inherited from **timed-element**. A "0" value indicates that the contact is momentary or transient. In Labanotation, a transient relationship is shown by drawing a double bow. If this value is not specified, the relationship is retained until it is cancelled or is made invalid by another action.

The **relationship** element has two major components:

- **parts** - inherited from **movement**. These are the primary or active parts in the relationship; and

- **other-parts** - these are the target or the passive parts in the relationship.

The **relationship** element itself is an abstract class and cannot be instantiated. The concrete subclasses of **relationship** with varying level of involvement are: **address**, **approach**, **touch**, and **carry**.

### 3.9.1 Address

Addressing has the lowest degree of involvement among the **relationship** subclasses. The **address** element is used to indicate that a body part is addressing someone or something.

### 3.9.2 Approach

The **approach** element is used to indicate nearness or closeness. That is, when both parts approach each other without making any contact. Figure 3.18 shows how the **approach** element is used.
3.9.3 Touch

Touching in Labanotation is shown by using a contact bow. MovementXML extends this role to include hooks in Labanotation. A hook is defined as a part of the foot that is touching the floor.

3.9.4 Carry

Carrying is a relationship where one part takes weight or supports another part. In this action, the parts components do the supporting while the other-parts are the supported.

3.10 Body Areas

A body area can be a specific body part, a range of body parts, or a surface of a specific or range of body parts.

3.10.1 Body Part

A specific body part is represented by the body-part element. The body-part element is a subclass of stage-object and has the following attributes:

- name - inherited from stage-object. This refers to the name of the specific body part. Table 3.6 shows the common predefined convenience elements that are available in MovementXML;

- side - refers to the side of the body the body part belongs to. This attribute can have the following values:
Figure 3.18: Both hands approach each other.

- "left" - for the left side of the body (e.g. the left arm);
- "right" - for the right side of the body (e.g. the right arm);
- "any" - if the side of the body is not important (e.g. any arm); and
- "both" - refers to both sides of the body (e.g. both arms).

- **performer** - the name of the performer that owns this body part. If unspecified, the body part is assumed to belong to any of the performers that are associated with the staff.

Several convenience classes are available for the different body parts.

**Fingers and Toes**

Fingers and toes have additional attributes to indicate the specific fingers or toes as well as their particular segments:

- **number** - an integer value from "1" to "5": "1" being the thumb or the big toe, and "5" being the pinkie or the little toe;
CHAPTER 3. THE REPRESENTATION

```xml
<touch start="0">
    <parts>
        <foot side="right" part="full-heel" />
    </parts>
    <other-parts>
        <floor />
    </other-parts>
</touch>
```

Figure 3.19: The full heel touching the floor.

```xml
<carry start="0">
    <parts>
        <stage-prop name="bench" />
    </parts>
    <other-parts>
        <knee side="right" />
    </other-parts>
</carry>
```

Figure 3.20: The right knee is supported by the bench.

- **part** - this refers to the part or segment of the finger or toe. Valid values for this attribute are:
  - "whole" - for the whole finger or toe. This is the default value;
  - "base-knuckle" - the knuckle closest to the base palm or the sole;
  - "middle-knuckle" - the knuckle after the base knuckle;
  - "last-knuckle" - the knuckle just before the tip; and
  - "tip" - the tip of the finger or toe.

**Foot**

The foot element can also indicate a portion of the foot. These portions can be used in a touch element to indicate a hook in Labanotation. The foot element has a single attribute:
Table 3.6: The common body part elements and their equivalent body-part representations.

- part - this refers to the part of the foot. The valid values for this attribute as defined in Labanotation are:
  - "whole" - this is the default value;
  - "full-heel";
  - "half-heel";
  - "eight-ball";
  - "quarter-ball";
  - "half-ball";
  - "full-ball";
  - "toe-pad";
CHAPTER 3. THE REPRESENTATION

- "full-toe"; and
- "toe-nail".

3.10.2 Body Part Range

Body part ranges are a collection of body parts that perform as a unit[15] (Fig. 3.21). The body-part-range element is a subclass of stage-object and takes in two body part elements:

- from-part - the starting body part; and
- to-part - the ending body part.

```
<body-part-range>
  <from-part>
    <head side="both" />
  </from-part>
  <to-part>
    <toe side="both" />
  </to-part>
</body-part-range>
```

Figure 3.21: A body part range from head to toe.

3.10.3 Body Part Surface

The body-part-surface element represents a surface of a body part rather than the body part itself (Fig. 3.22). A movement that uses a body part surface is understood to be a facing movement. That is, the surface being referred to faces the direction specified by the movement.

The body-part-surface contains the following elements:

- part - the body-part or body-part-range; and
- description - the description of the surface. This takes in the following descriptors:
  - direction - describes the horizontal position on the surface of the associated body part; and
3.10.4 Phrase

The phrase element represents Labanotation's phrasing bows. A phrase binds several movements together to form a movement phrase. Although the phrase does not affect the movements in any way, it is sometimes helpful to group movements together to show a unity of thought.[15] The phrase element derives from timed-element and has a single component, movements, which takes in a collection of movement elements.

3.11 State Channel Elements

The state channel elements are used to set the state variables in which the movements operate. The elements in the state channel are timed elements in which their duration attributes are set to "0" to indicate that the state is in effect until it is replaced.

3.11.1 Cross of Axes

The cross of axes, or the system of reference, is used to describe how directions should be interpreted. This set-cross-of-axes command is used to set the performer's cross of axes. It can have the following values:
• "standard" - refers to the Standard Cross of Axes wherein the forward direction is defined by where the performer is facing while up and down are defined by the line of gravity;

• "twisted-part" - directions in relation to the front of the free end of the part twisting. This is the default for the torso, pelvis, chest, shoulder, and head;

• "stance" - directions related to the front established by the stance;

• base-of-part-twisted - directions in relation to the front of the base of the twisting part. This is the default for limbs;

• "body" - refers to the Cross of Body Axes wherein the up and down are defined by the body's longitudinal axis;

• "body-twisted-part" - same as "twisted-part" except that it uses the Cross of Body Axes instead of the Standard Cross of Axes;

• "body-stance" - same as "stance" except that it uses the Cross of Body Axes instead of the Standard Cross of Axes;

• "body-base-of-part-twisted" - same as "base-of-part-twisted" except that it uses the Cross of Body Axes instead of the Standard Cross of Axes;

• "constant" - refers to the Constant Cross of Axes wherein the directions remain constant regardless of where the performer is facing; and

• "fixed-point" - directions are in reference to a fixed point. The set-cross-of-axes element needs to take in a stage object that will serve as its fixed point.

3.11.2 Focal Point

The focal point can be any object on the stage including the performers themselves. The set-focal-point is used to set the current focal point. This focal point is in effect until the next set-focal-point command is issued. This element takes in a single stage-object which serves as the new focal point. Figure 3.23 shows how to set the current focal point while figure 3.24 shows how this focal point is used through the focal-point element.
CHAPTER 3. THE REPRESENTATION

3.12 Location Channel Elements

Location channel elements specify the location, orientation, and path of a performer. These elements, combined with the support channel, determine the floor pattern in the stage area.

3.12.1 Location

The set-location command determines the location of the performer’s support at a specific point in time. The location is usually specified as a stage-part but it can also accept any stage-object (Fig. 3.25).

```
<set-location start="0">
  <floor>
    <center />
  </floor>
</set-location>
```

Figure 3.25: Set the location to the center of the stage floor.
3.12.2 Orientation

The set-orientation command is used to determine the direction of a side of the body with respect to its surroundings. This element represents the front signs and the front in relation to a circle in Labanotation. A set-orientation's duration is always set to "0".

The set-orientation command takes in 2 elements:

- **side** - the side of the body fronting the target. This element takes in a direction and a level. The level is only necessary for 3-dimensional orientation; and

- **target** - the target direction. This can be any of the descriptors that describe directions (Fig. 3.26).

```
<set-orientation start="1">
  <side>
    <left />
  </side>
  <target>
    <towards>
      <focal-point />
    </towards>
  </target>
</set-orientation>
```

Figure 3.26: The left side of the body faces the focal point.

3.12.3 Path

A path is a description of the general direction the performer has to take. It can also optionally specify secondary support movements that needs to be done throughout the duration of the path. The set-path action has the following attributes:

- **type** - specifies the type or shape of the path. There are 4 possible values:
  - "straight" - for a straight path;
  - "circular" - for a circular path;
  - "stationary" - tells the performer to stay in place; and
- "any" - the performer has the freedom to choose any path;

- **strict** - a boolean attribute that indicates whether the path should be followed strictly. A "false" value in a circular path is equivalent to the ad libitum sign placed next to the degree of the circling and gives the performer the freedom in interpreting the shape of the circle; and

- **direction** - can be "clockwise", "counter-clockwise", or "any". This attribute is only applicable for circular paths.

The **set-path** command contains 2 other elements:

- **movements** - specifies the movements that need to be performed for the duration of the path; and

- **description** - takes in 0 or more movement descriptors (Ch. 3.8) which describe different aspects of the path (Fig. 3.27).

```xml
<set-path start="0" duration="4" type="circular" strict="true" direction="clockwise">  
  <movements>
    <move>
      <description>
        <forward />
      </description>
    </move>
  </movements>
  <description>
    <direction>360</direction>
    <degree>7</degree>
  </description>
</set-path>
```

Figure 3.27: Make a very big full circle moving forward.
Chapter 4

The Tinikling Project

Tinikling (Bamboo dance) is a Filipino folk dance wherein the dancers would imitate the tikling bird's legendary grace and speed by skillfully maneuvering between large bamboo poles.\(^1\) These poles when placed on the floor have a striking resemblance to a Labanotation staff.

The Tinikling is a Labanotation editor that is developed on the Eclipse platform (Prog. 4.1). The researcher's intention is to use this as a proof of concept of what an intelligent Labanotation editor can be. The Tinikling editor has 3 parts:

- the layout page;
- the palette; and
- the properties view.

Layout Page

The layout page is created using GEF (Graphical Editing Framework). This is where the staff, measure, and all the movement elements are placed. The layout page is a full-blown editor with features like drag and drop.

Palette

The palette actually belongs to the layout page. It contains the different elements that can be dropped into the layout page. These elements include the staff, the measure, and all the

\(^1\)Source: http://www.likha.org/galleries/tinikling.asp
movement elements.

**Properties View**

The properties view displays the properties of the currently selected elements in the layout page. The properties view allows users to change the beat count of a measure, the duration of a movement element, the direction and level of a movement element, etc.

![Tinkling Dance Editor](image)

Program 4.1: Tinikling Dance Editor

**Plug-in Architecture**

The Tinikling editor can take advantage of the Eclipse plug-in architecture to extend the editor to provide features such as: a 3D animation generator for the score; grammar checkers; etc.
Chapter 5

Analysis

5.1 MovementXML vs. Existing Formats

This analysis is a comparison among MovementXML, LabanXML, and LabanWriter. The comparison is based on some of the criteria given by the Ohio report[7] on what its interlingua should be like. Interlingua is the name given to the file format that can be shared among the different Labanotation editors (LabanWriter, Labanatory, and Calaban) and other programs such as LabanDancer.

<table>
<thead>
<tr>
<th>Feature</th>
<th>MovementXML</th>
<th>LabanXML</th>
<th>LabanWriter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Score preservation</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Searchable</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Pattern searchable</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Grammar check</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Transferability</td>
<td>Yes</td>
<td>Limited</td>
<td>No</td>
</tr>
<tr>
<td>Analyzable</td>
<td>Yes</td>
<td>Limited</td>
<td>No</td>
</tr>
<tr>
<td>Extensibility</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Flexibility</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Table 5.1: MovementXML feature comparison

Score Preservation

This criteria pertains to the ability of a score to be preserved when it is saved and loaded again. Note that MovementXML and LabanXML always put movements in their proper columns. This is because both MovementXML and LabanXML do not store the column
information. This prevents notators from arbitrarily putting movements in the wrong spots ensuring a score that is always correct.

**Searchable**

This refers to the ability of the score to be searched. Both MovementXML and LabanXML are highly structured and therefore allow more predictable searching. LabanWriter, on the other hand, uses a graphical representation that is not searchable. This is because the relationship among LabanWriter elements are not well defined.

**Pattern Searchable**

This refers to the ability of the score to be searched for patterns. Again, both MovementXML and LabanXML allow this type of search because of their structured nature.

**Grammar Check**

This refers to the ability of the score to be grammar checked. The absence of column information in both MovementXML and LabanXML ensure that the elements are always in the correct place. Their hierarchical structure and their deterministic nature allows grammar checkers to easily traverse their hierarchy.

**Transferability**

Transferability refers to the score’s ability to be used outside the editor that created it. Although LabanWriter files can be played by the LabanDancer, it is only made possible by converting the LabanWriter file into an intermediate representation that is similar to that of MovementXML and LabanXML. This process of conversion is not always accurate since LabanWriter allows its elements to be placed anywhere.

LabanXML, on the other hand, has only limited transferability. This is because LabanXML is only a semi-semantic representation. Elements like relationship pins do not mean anything to applications that are not familiar with the Labanotation objects. In MovementXML, relationship pins are referred to as position movements.
**Analyzable**

This refers to the score’s ability to be analyzed – be it a choreographic or movement analysis. Again, LabanWriter files cannot be analyzed because of its lack of structure. LabanXML can only be analyzed to a certain extent because of its semi-semantic nature. MovementXML can be analyzed fully.

**Extensibility**

Extensibility refers to how well the representation can be extended. Only MovementXML has this capability, and it does this through the defined movements. New movements can be created by combining other movements. We can even create Motif-like movements and make MovementXML behave like Motif.

**Flexibility**

Flexibility refers to how flexible the movements are in accepting new behaviors. LabanXML’s movements are very rigid. It contains a constant number of properties.

LabanWriter, on the other hand, is too flexible in that it can have movements that don’t make sense at all. This is because LabanWriter is a purely graphical editor which does not have a sense of what its elements mean.

MovementXML abstracted the movement descriptors away from the movements themselves. For example, a move element can be made to move towards a part of the stage or another performer. This separation between the movement and the descriptor allows notators to mix and match movements to descriptors.

**5.2 A Semantic Representation of Human Movement**

Labanotation in itself is a very rich language to describe human movement. The key to coming up with a semantic representation of human movement is to describe what the different symbols or combination of symbols mean.

LabanWriter, Calaban, and Labanatory are graphical representations of a Labanotation score. They contain information on the score’s layout but have very little to offer in terms of what the different symbols mean.
LabanXML, on the other hand, is a semantic representation of the notation rather than what the notation represents. A good example of this is the representation of a relationship pin. In Labanotation, a pin is used to describe a full range of miniature movements with respect to the center line of the body. LabanXML represents a relationship pin as a relationship pin, which is a semantic description of the notation itself rather than what the symbol represents.

A huge amount of effort went into analyzing Labanotation as a language during the creation of MovementXML. First, the different support and gesture columns were merged together resulting in the creation of the support and gesture channels, respectively (Ch. 5.4.1 describes the differences between channels and columns in more detail). Second, relationship pins were dropped and replaced by the position movement. The position movement extends the meaning of relationship pins to describe subtle movements of body parts with respect to any body part, not just the center line of the body. Third, the ability to use defined movements in the complex channel allows MovementXML to represent higher-level movements. This also allows MovementXML to describe other human movement notations such as Motif. Foremost, the separation of the body parts (subject) and the descriptors (adverbs) in movements (verb) makes MovementXML just as descriptive and flexible as Labanotation.

5.3 Tinikling

The Tinikling editor was created as a proof that a Labanotation editor can be created with well defined elements similar to that of MovementXML. These elements also enforce some simple grammar rules like: not being able to drop a directional move outside of measures; measures can only be added to staffs; etc. This type of enforcement is missing from LabanWriter.

All the elements in the layout page have properties that are modifiable. Figure 5.1 shows how a movement's direction can be changed.

5.4 MovementXML vs. Labanotation

MovementXML is based on Labanotation but there are some subtle differences between the two. In particular, MovementXML moved away from objects that are purely aesthetic in
nature. Some objects like the pins were changed in favor of elements or attributes that convey what the pins mean.

5.4.1 Channels vs. Columns

Support Channel

There are two support columns in Labanotation, one for each side of the body. These support columns are combined as the support channel in MovementXML. Having a separate channel for the left and the right sides of the body is unnecessary and does not add any value to the representation since the body parts already contain this information.

In Labanotation, a movement in the support column is presumed to be a movement of the foot unless otherwise specified. This is not the case with MovementXML. All movements in MovementXML must specify the body part that the movement applies to.

Gesture Channel

There are several gesture columns in Labanotation, one for every major part of the body. Extra subsidiary columns can also be added when necessary. There is only one gesture
channel in MovementXML and this channel contains all the information in all the gesture columns in Labanotation.

**Location Channel**

The location channel encompasses several aspects of Labanotation. These aspects include:

- **the floor plan** - using the set-location action (Ch. 3.12.1)
- **paths** - using the set-path action (Ch. 3.12.3)

### 5.4.2 Complex Channel Advantage

The complex channel has no equivalent in Labanotation. At the very least, the complex channel can be seen as the combination of all the columns. The complex channel provides the following advantages:

- **Reuse** - complex movement phrases and patterns that are being used repeatedly can be named and turned into defined movements (Ch. 3.3);
- **Extensibility** - having defined movements opens up the MovementXML to new ways of describing movements. For instance, a library of Motif movements can be defined, enabling us to compose MovementXML scores entirely in Motif;
- **Analysis** - having higher-level movements that are focused on the intent of the movement rather than the structural details enables us to perform a higher-level analysis of the movement; and
- **Readability** - using named complex movements makes the score shorter and more readable.

### 5.4.3 Intermediate Movements

Labanotation tries to keep things simple by providing only the major direction and level symbols. Intermediate directions and levels can be achieved by using the halfway point or the third way point.

MovementXML does not need to represent the halfway and third way point indicators since its levels and directions are measured in degrees. Moreover, unlike Labanotation, a
specific movement can only take a single form. This has the added advantage of being able to search a movement without having to go through all the different combinations of halfway and third way points.

5.4.4 Simultaneous Movements

In Labanotation, simultaneous movements are drawn with a vertical bow or by splitting the columns. There is no equivalent representation for simultaneous movements. Such a representation is unnecessary.

In MovementXML, simultaneous movements are movements whose execution times overlap. It is up to the editor to determine whether these movements should be shown in split columns or by using a simultaneous bow.

5.4.5 Jumps

MovementXML follows the same rules as Labanotation as far as absence of movement in the support channel is concerned. However, if movements in the support channel are present, a lift movement must be invoked explicitly in order to convey a jump. The lift movement is similar to Labanotation's vertical lines.

5.4.6 Center of Gravity

The center of gravity is just another body part in MovementXML. All the actions that apply to other body parts also apply the the center of gravity.

5.4.7 Inclusion Bow

Inclusion bows are replaced by the follow descriptor (Ch. 3.8.6). The follow descriptor indicates that a movement is to a lesser degree just like the inclusion bow. The follow descriptor can also be placed in other movements such as a turn to indicate a passive turn.

5.4.8 Addition Bow

There is no need for an addition bow in MovementXML since movements can take in several body parts. This has the same effect as the addition bow.
5.4.9 3rd Dimension

MovementXML extends Labanotation to the 3rd dimension. The stage has a new height attribute while the facing pins can now have levels (vertical directions).
Chapter 6

Conclusion and Recommendations

MovementXML is a semantic representation of human movement based on Labanotation. It is an interchange format that is based on a standard (XML), and it succeeds in its role to preserve choreography. MovementXML’s strict hierarchical structure allows for predictable searching, pattern matching, and motion and choreographic analysis.

This research succeeded in extracting and organizing the meaning behind the Labanotation symbols. The resulting representation is simpler, more flexible, and more extensible than Labanotation itself. The defined movements and complex channel features can be extended to allow MovementXML to capture scores from Motif and other movement notations.

6.1 Future recommendations

6.1.1 Extending MovementXML

Further research can be made on how MovementXML can fit into other notations such as Motif, Benesh Movement Notation, DanceWriting, and Eshkol-Wachman Movement Notation. In particular, further studies can be made on how the complex channel and the defined movements feature can accommodate these other notations.

6.1.2 Finish Tinikling Editor

The Tinikling editor has much potential. The editor is very well designed and it is currently the best path to building a full-blown editor that supports MovementXML.
6.1.3 Dance Ontology

Minako Nakamura indicated in her research that the next challenges after creating LabanXML are dance analysis and dance archive [19]. There is a trend towards creating ontologies and semantic webs for different knowledge domains. Ontologies are often used in artificial intelligence and knowledge representation\(^1\). The OWL Web Ontology Language [6] is a formal language for representing ontologies in the semantic web [14] and can be a tool for intelligent archiving that will lead to intelligent queries on dance databases.

A dance ontology language describes the essential components of a dance score: genres, author, title, etc. Using such ontology language results in a semantic network that acts as a knowledge base. This semantic network facilitates better classification of dance scores and allows complex queries to be made.

A future goal would be to come up with a dance repository that is updated and queried through OWL using Protégé. The dance library will be built on top of a dance ontology language that describes the essential components of a dance score such as the one described above (Fig. 6.1). This feature can be built in easily as a plug-in to Tinikling.

---

Appendix A

MovementXML Schema

```xml
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
xmlns:xs="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified"
attributeFormDefault="unqualified">
  <xs:element name="score">
    <xs:annotation>
      <xs:documentation/>
    </xs:annotation>
    <xs:complexType>
      <xs:sequence>
        <xs:element name="authors">
          <xs:complexType>
            <xs:choice maxOccurs="unbounded">
              <xs:element name="author">
                <xs:complexType>
                  <xs:attribute name="last-name" type="xs:string" use="required"/>
                  <xs:attribute name="given-names" type="xs:string" use="required"/>
                </xs:complexType>
              </xs:element>
            </xs:choice>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```
<xs:complexType>
  <xs:element name="categories">
    <xs:complexType>
      <xs:choice maxOccurs="unbounded">
        <xs:element name="category">
          <xs:complexType>
            <xs:attribute name="name" type="xs:string" use="required"/>
            <xs:attribute name="type" type="xs:string" use="required"/>
          </xs:complexType>
        </xs:element>
      </xs:choice>
    </xs:complexType>
  </xs:element>
  <xs:element name="defined-movements"/>
  <xs:element name="stage">
    <xs:complexType>
      <xs:attribute name="width" type="xs:double" use="required"/>
      <xs:attribute name="depth" type="xs:double" use="required"/>
      <xs:attribute name="height" type="xs:double"/>
      <xs:attribute name="unit" type="xs:string" use="required"/>
    </xs:complexType>
  </xs:element>
  <xs:element name="staffs">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="staff">
          <xs:complexType>
            <!-- Further elements or attributes go here -->
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:complexType>
<xs:sequence>
  <xs:element name="base-measure"
    type="staff-segment"/>
  <xs:element name="measure" maxOccurs="unbounded">
    <xs:complexType>
      <xs:complexContent>
        <xs:extension base="staff-segment">
          <xs:sequence>
            <xs:element name="support"/>
            <xs:element name="gesture"/>
            <xs:element name="location"/>
            <xs:element name="state"/>
            <xs:element name="complex"/>
          </xs:sequence>
          <xs:attribute name="beats" type="xs:integer"/>
          <xs:attribute name="beat-type" type="xs:string"/>
          <xs:attribute name="beat-duration" type="xs:double"/>
          <xs:attribute name="meter" type="xs:integer"/>
        </xs:extension>
      </xs:complexContent>
    </xs:complexType>
  </xs:element>
</xs:sequence>
<xs:element name="repeat" maxOccurs="unbounded">
  <xs:complexType>
    <xs:complexContent>
      <xs:extension base="staff-segment">
        <xs:sequence/>
        <xs:attribute name="count" type="xs:integer"/>
        <xs:attribute name="repeat-type" type="xs:string"/>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:element>
</xs:complexType>
</xs:complexType>
</xs:element>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:element>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
<xs:choice>
  <xs:element ref="initial-location"/>
</xs:choice>

<xs:attribute name="name" type="xs:string"/>
<xs:attribute name="prop-type" type="xs:string"/>
</xs:complexType>
</xs:element>
</xs:choice>
</xs:complexType>
</xs:element>
</xs:sequence>
<xs:attribute name="title" type="xs:string" use="required"/>
<xs:attribute name="date-created" type="xs:date" use="required"/>
</xs:complexType>
</xs:element>
</xs:complexType name="stage-object" abstract="true">
<xs:attribute name="name" type="xs:string" use="required"/>
<xs:attribute name="type" type="xs:string" use="required" form="unqualified"/>
</xs:complexType>
</xs:element>
<xs:element name="performer">
<xs:complexType>
<xs:complexContent>
<xs:restriction base="stage-object">
  <xs:attribute name="type" fixed="performer"/>
</xs:restriction>
</xs:complexContent>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
<xs:element name="stage-prop"/>
<xs:complexType>
  <xs:complexContent>
    <xs:extension base="stage-object">
      <xs:attribute name="prop-type" type="xs:string" use="required"/>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>

<xs:element name="initial-location"/>

<xs:complexType name="staff-segment"/>

<xs:complexType name="timed-element">
  <xs:attribute name="start" type="xs:double" use="required"/>
  <xs:attribute name="duration" type="xs:double"/>
</xs:complexType>

<xs:element name="move">
  <xs:complexType>
    <xs:complexContent>
      <xs:extension base="movement">
        <xs:attribute name="path" type="xs:string"/>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
</xs:element>

<xs:complexType name="movement">
  <xs:complexContent>
    <xs:extension base="timed-element">
      <xs:sequence>
        <xs:element name="parts">
          <xs:complexType/>
        </xs:element>
        <xs:element>
          <xs:complexType name="description"/>
        </xs:element>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
</xs:sequence>
</xs:extension>
</xs:complexContent>
</xs:complexType>
<xs:element name="turn">
<xs:complexType>
<xs:complexContent>
<xs:extension base="movement">
<xs:attribute name="direction" type="xs:string"/>
</xs:extension>
</xs:complexContent>
</xs:complexType>
</xs:element>
<xs:element name="wheel">
<xs:complexType>
<xs:complexContent>
<xs:extension base="movement">
<xs:attribute name="direction" type="xs:string"/>
</xs:extension>
</xs:complexContent>
</xs:complexType>
</xs:element>
<xs:element name="somersault">
<xs:complexType>
<xs:complexContent>
<xs:extension base="movement">
<xs:attribute name="direction" type="xs:string"/>
</xs:extension>
</xs:complexContent>
</xs:complexType>
</xs:element>
<xs:element name="cartwheel">
<xs:complexType>
<xs:complexContent>
  <xs:extension base="movement">
    <xs:attribute name="direction" type="xs:string"/>
  </xs:extension>
</xs:complexContent>
</xs:complexType>
</xs:element>

<xs:element name="hold">
  <xs:complexType>
    <xs:complexContent>
      <xs:extension base="movement">
        <xs:attribute name="type" type="xs:string"/>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
</xs:element>

<xs:element name="return-to-normal" type="movement"/>

<xs:element name="lift" type="movement"/>

<xs:element name="contract">
  <xs:complexType>
    <xs:complexContent>
      <xs:extension base="movement">
        <xs:attribute name="axis" type="xs:string"/>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
</xs:element>

<xs:element name="extend">
  <xs:complexType>
    <xs:complexContent>
      <xs:extension base="movement">
        <xs:attribute name="axis" type="xs:string"/>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
</xs:element>
<xs:complexType>
	<xs:complexContent>
		<xs:extension base="movement">
			<xs:attribute name="repeat-measure" type="xs:integer"/>
			<xs:attribute name="repeat-start" type="xs:double"/>
			<xs:attribute name="repeat-type" type="xs:string"/>
		</xs:extension>
	</xs:complexContent>
</xs:complexType>
</xs:element>

<xs:element name="continue">
	<xs:complexType>
		<xs:complexContent>
			<xs:extension base="movement">
			</xs:extension>
	</xs:complexContent>
</xs:complexType>
</xs:element>
<xs:attribute name="repeat-type" type="xs:string"/>
</xs:extension>
</xs:complexType>
</xs:element>
<xs:complexType name="descriptor">
<xs:element name="direction" type="descriptor"/>
<xs:element name="level" type="descriptor"/>
<xs:element name="degree" type="descriptor"/>
<xs:element name="towards" type="descriptor"/>
<xs:element name="away" type="descriptor"/>
<xs:element name="accent" type="descriptor"/>
<xs:element name="follow" type="descriptor"/>
<xs:element name="cross-of-axes" type="descriptor"/>
<xs:element name="distance" type="descriptor"/>
<xs:element name="coordinate">
<xs:complexType>
<xs:complexType name="coordinate">
<xs:complexType name="relationship">
<xs:complexType>
<xs:extension base="movement">
<xs:sequence>
<xs:element name="other-parts"/>
APPENDIX A. MOVEMENTXML SCHEMA

</xs:sequence>

<xs:attribute name="action" type="xs:string"/>
<xs:attribute name="action-part" type="xs:string"/>
</xs:extension>
</xs:complexType>
</xs:element>
<xs:element name="address">
<xs:complexType>
<xs:complexContent>
<xs:extension base="relationship">
<xs:attribute name="side" type="xs:string"/>
<xs:attribute name="performer" type="xs:string"/>
</xs:extension>
</xs:complexContent>
</xs:complexType>
</xs:element>
<xs:element name="approach">
<xs:complexType>
<xs:complexContent>
<xs:extension base="relationship">
<xs:attribute name="number" type="xs:integer"/>
<xs:attribute name="part" type="xs:string"/>
</xs:extension>
</xs:complexContent>
</xs:complexType>
</xs:element>
<xs:element name="touch">
<xs:complexType>
<xs:complexContent>
<xs:extension base="relationship">
<xs:attribute name="number" type="xs:integer"/>
<xs:attribute name="part" type="xs:string"/>
</xs:extension>
</xs:complexContent>
</xs:complexType>
</xs:element>
<xs:element name="carry">
<xs:complexType>
<xs:complexContent>
<xs:extension base="relationship">
<xs:attribute name="number" type="xs:integer"/>
<xs:attribute name="part" type="xs:string"/>
</xs:extension>
</xs:complexContent>
</xs:complexType>
</xs:element>
<xs:element name="body-part">
<xs:complexType>
<xs:complexContent>
<xs:extension base="stage-object">
<xs:attribute name="side" type="xs:string"/>
<xs:attribute name="performer" type="xs:string"/>
</xs:extension>
</xs:complexContent>
</xs:complexType>
</xs:element>
<xs:element name="finger">
<xs:complexType>
<xs:complexContent>
<xs:extension base="body-part">
<xs:attribute name="number" type="xs:integer"/>
<xs:attribute name="part" type="xs:string"/>
</xs:extension>
</xs:complexContent>
</xs:complexType>
</xs:element>
<xs:element name="toe">
<xs:complexType>
<xs:complexContent>
<xs:extension base="body-part">
<xs:attribute name="number" type="xs:integer"/>
</xs:extension>
</xs:complexContent>
</xs:complexType>
</xs:element>
<xs:attribute name="part" type="xs:string"/>
</xs:extension>
</xs:complexContent>
</xs:complexType>
</xs:element>
<xs:element name="foot">
<xs:complexType>
<xs:complexContent>
<xs:extension base="body-part">
<xs:attribute name="part" type="xs:string"/>
</xs:extension>
</xs:complexContent>
</xs:complexType>
</xs:element>
<xs:element name="head" type="body-part"/>
<xs:element name="neck" type="body-part"/>
<xs:element name="body" type="body-part"/>
<xs:element name="shoulder" type="body-part"/>
<xs:element name="chest" type="body-part"/>
<xs:element name="waist" type="body-part"/>
<xs:element name="pelvis" type="body-part"/>
<xs:element name="arm" type="body-part"/>
<xs:element name="upper-arm" type="body-part"/>
<xs:element name="lower-arm" type="body-part"/>
<xs:element name="elbow" type="body-part"/>
<xs:element name="wrist" type="body-part"/>
<xs:element name="hand" type="body-part"/>
<xs:element name="hip" type="body-part"/>
<xs:element name="leg" type="body-part"/>
<xs:element name="upper-leg" type="body-part"/>
<xs:element name="lower-leg" type="body-part"/>
<xs:element name="knee" type="body-part"/>
<xs:element name="ankle" type="body-part"/>
APPENDIX A. MOVEMENTXML SCHEMA

<xs:element name="center-of-gravity" type="body-part"/>
<xs:element name="body-part-range">
  <xs:complexType>
    <xs:complexContent>
      <xs:extension base="body-part">
        <xs:sequence>
          <xs:element name="from-part"/>
          <xs:element name="to-part"/>
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
</xs:element>
<xs:element name="body-part-surface">
  <xs:complexType>
    <xs:complexContent>
      <xs:extension base="body-part">
        <xs:sequence>
          <xs:element name="part"/>
          <xs:element name="description"/>
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
</xs:element>
<xs:element name="phrase">
  <xs:complexType>
    <xs:complexContent>
      <xs:extension base="timed-element">
        <xs:sequence>
          <xs:element name="movements"/>
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
</xs:element>
APPENDIX A. MOVEMENTXML SCHEMA

</xs:complexContent>
</xs:complexType>
</xs:element>
<xs:element name="set-cross-of-axes" type="timed-element"/>
<xs:element name="set-focal-point" type="timed-element"/>
<xs:element name="set-location" type="timed-element"/>
<xs:element name="set-orientation">
<xs:complexType>
<xs:complexContent>
<xs:extension base="timed-element">
<xs:sequence>
<xs:element name="side"/>
<xs:element name="target"/>
</xs:sequence>
</xs:extension>
</xs:complexContent>
</xs:complexType>
</xs:element>
<xs:element name="set-path">
<xs:complexType>
<xs:complexContent>
<xs:extension base="timed-element">
<xs:sequence>
<xs:element name="movements"/>
<xs:element name="description"/>
</xs:sequence>
<xs:attribute name="type" type="xs:string"/>
<xs:attribute name="strict" type="xs:boolean"/>
<xs:attribute name="direction" type="xs:string"/>
</xs:extension>
</xs:complexContent>
</xs:complexType>
</xs:element>
</xs:schema>
Appendix B

Examples
<repeat count="1" repeat-type="exact" exclude-path="true">
  <measure beats="4" beat-type="note" beat-duration="0.25" meter="96">
    <support>
      <move start="0" duration="1">
        <part><foot side="right" /></part>
        <description><forward /></description>
      </move>
      <move start="1" duration="1">
        <part><foot side="left" /></part>
        <description><forward /></description>
      </move>
      <move start="2" duration="1">
        <part><foot side="right" /></part>
        <description><forward /></description>
      </move>
      <move start="3" duration="1">
        <part><foot side="left" /></part>
        <description><forward /></description>
      </move>
    </support>
    <location>
      <set-path start="0" duration="4" type="circular" strict="true" direction="clockwise">
        <description>
          <direction>360</direction>
        </description>
      </set-path>
    </location>
  </measure>
</repeat>

Figure B.1: Repeat example resulting in taking 8 steps to complete a circle.
Figure B.2: Repeat example resulting in taking 4 steps to complete a circle; 2 circles are completed.
Bibliography


Index

3D animation, 8
3rd Dimension, 53

A. William Smith, 6
Absence of movement, 20, 21
Accent, 31
Addition bow, 22, 52
Address, 33
Aerial turn, 24
Aerial turns, 20, 26
Analogy signs, 29
Analysis, 51
ankle, 37
Ann Hutchinson Guest, 5
Approach, 33
arm, 37
Artificial intelligence, 55
authors, 11
Away from, 30
Base measure, 18
base-knuckle, 36
beat-duration, 18
beat-type, 18
beats, 18
Benesh Movement Notation, 6, 54
body, 37
Body areas, 34
Body hold, 25
Body part, 34
Body part range, 38
Body part surface, 38
Body wave, 15, 21
body-part, 14
body-part-range, 14, 38

body-wave, 15
Calaban, 7, 48
Carry, 34
Cartwheel, 25
categories, 12
Center of gravity, 52
center-of-gravity, 37
Channels, 19
chest, 37
Circular path, 42
Complex Channel, 21
Complex channel, 49, 51
Compound movements, 22
Contact bow, 34
Containing measure, 22
Continuation, 29
continue, 29
Contraction, 26
Coordinate, 32
count, 19
Credo Interactive, 8
Cross of Axes, 31, 39
Cross of Body Axes, 40
cross-of-axes, 21

Dance composition, 10
Dance Ontology, 55
DanceWriting, 6, 54
Data Type Definition, 8
date-created, 11
David Railey, 6
define-movement, 16
define-performer, 14
define-stage-prop, 15
INDEX

lower-arm, 37
lower-leg, 37
Lucy Venable, 6

Measure, 18
Measure number, 18
Metadata, 10
meter, 18
Michael Good, 7
middle-knuckle, 36
Minako Nakamura, 6, 7, 55
Motif, 49, 51, 54
Move, 23
Movement, 22
Movement description, 22
Movement parts, 22
MovementWriting, 6
MovementXML, 6
MovementXML schema, 56
MusicXML, 7

Natural Language Processing, 10
neck, 37

Ohio State University, 6
Orientation, 42

Palette, 44
Path, 42
pelvis, 37
performer, 14
Performers, 14
Phrase, 39
Phrasing bow, 39
Plug-in Architecture, 45
Position, 27
Position signs, 27
Properties View, 45

quarter-ball, 37
Quaternions, 1

Readability, 51
Recordare, 7

relationship, 32
Relationship movements, 32
Relationship pin, 49
Relationship pins, 27
Repeat, 19
repeat-measure, 28
repeat-movement, 28
repeat-start, 29
repeat-type, 19, 29
Retention signs, 25
Return to Normal, 25
return-to-normal, 25
Reuse, 51
Rhonda Ryman, 8
Rudolf Benesh, 6
Rudolf von Laban, 5

Score, 10
Score components, 12
score-partwise, 7
score-timewise, 8
Scott Sutherland, 6
Semantic Representation, 9, 48
Semantic Web, 55
Semantics, 9
set-cross-of-axes, 39
set-focal-point, 40
set-location, 41
set-orientation, 42
set-path, 42
SGML, 8
Sheila Marion, 6
Shift, 27
shoulder, 37
Simultaneous Movements, 52
Somersault, 24
Space hold, 25
Split columns, 52
Spot hold, 25
Spreading, 27
Staff, 16
Staff segment, 17
staff-segment, 17
INDEX

Stage, 12
Stage area, 21, 41
Stage Objects, 13
Stage props, 15
stage-object, 13
stage-part, 14, 41
stage-prop, 14, 15
Standard Cross of Axes, 40
Standard Generalized Markup Language, 8
start, 22
State Channel, 21
State channel elements, 39
Stationary path, 42
Stephen Smoliar, 8
Straight path, 42
Subsidiary columns, 50
Support Channel, 20
Support channel, 50
Syntax, 9
System of Reference, 39

Theatrical props, 15
Third way point, 51
Timed element, 21
Tinikling, 44
tip, 36
title, 11
toe, 37
toe-nail, 38
toe-pad, 37
Toes, 35
Tom Calvert, 8
Touch, 34
Towards, 30
Turn, 23

Unfolding, 27
upper-arm, 37
upper-leg, 37

Valerie Sutton, 6

W3C, 8

waist, 37
Wheel, 24
whole, 36, 37
width, 12
World Wide Web Consortium, 8
wrist, 37
XML, 8