

Florida International University
Knight Foundation School of Computing and Information Sciences



Software Engineering Focus

Final Deliverable

Project Title: jTLEX - A Java Library for Timeline Extraction

Team Members:

Capstone 1: Pedro Diaz, Emmanuel Garcia, Luis Robina

Capstone 2: Christopher Eberhard, Adrian Silva

Product Owner(s): Mark Finlayson, Mustafa Ocal

Mentor(s): Mustafa Ocal

Instructor: Masoud Sadjadi

The MIT License (MIT)
Copyright (c) 2016 Florida International University

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Abstract

This document presents the information necessary to gain a good understanding of jTLEX and the work that went into the methods as well as the user stories that were implemented for the process in which TimeML files are taken and parsed to allow for the jTLEX project to take that information and create graphs. The project will then take those graphs and convert them into TCSP which in turn enables jTLEX to produce a timeline of events while also checking that there are not any inconsistencies or indeterminant time points present to impede the creation of the timeline.

Table of Contents

INTRODUCTION	5
CURRENT SYSTEM	5
PURPOSE OF NEW SYSTEM	5
USER STORIES	
IMPLEMENTED USER STORIES	6
PENDING USER STORIES	6
PROJECT PLAN	
HARDWARE AND SOFTWARE RESOURCES	7
SPRINTS PLAN	8
<i>Sprint 1</i>	8
<i>Sprint 2</i>	8
<i>Sprint 3</i>	8
<i>Sprint 4</i>	8
<i>Sprint 5</i>	9
<i>Sprint 6</i>	10
<i>Sprint 7</i>	10
SYSTEM DESIGN	
ARCHITECTURAL PATTERNS	11
SYSTEM AND SUBSYSTEM DECOMPOSITION	12
DEPLOYMENT DIAGRAM	12
DESIGN PATTERNS	13
SYSTEM VALIDATION	14
GLOSSARY	15
APPENDIX	16
APPENDIX A - UML DIAGRAMS	16
<i>Static UML Diagrams</i>	16
<i>Dynamic UML Diagrams</i>	17
APPENDIX B - USER INTERFACE DESIGN	18
APPENDIX C - SPRINT REVIEW REPORTS	18
APPENDIX D - USER MANUALS, INSTALLATION/MAINTENANCE DOCUMENT, SHORTCOMINGS/WISHLIST DOCUMENT AND OTHER DOCUMENTS	20
REFERENCES	22

INTRODUCTION

The jTLEX project is a java library with the goal of timeline extraction from TimeML files. The purpose of jTLEX is to allow for a user to input a TimeML and using the methods provided by the jTLEX java library take said file parse it and produce a timeline of the events captured in that TimeML to determine what order and the length of time in which the events held within the document took place. The jTLEX java library will enable TimeML files to be legible for people, as it would make it simpler thanks to the timeline of events extracted from the TimeML files to be presented in a more concise manner than most TimeML annotated documents are currently.

Current System

Currently the means of TimeML produces an output of its contents as annotated text files, the graphs generated from the files only revealing part of the information in the form of the order of the events and times. As a result of lack of full ordering and data abstraction the process proves to be inadequate.

Purpose of New System

The jTLEX java library attempts to rectify the issues present in the current system by providing graphs created with the exact order in which the events and times take place to enable users to gain a better grasp of the order of events by presenting the information in a more effective and understandable manner.

USER STORIES

The following section provides the detailed user stories that were implemented in this iteration of the jTLEX project. These user stories served as the basis for the implementation of the project's features. This section also shows the user stories that are to be considered for future development.

Implemented User Stories

- The creation of a graph data structures.
- The partitioning of the graph data structures.
- Creating a converter of TimeML to TCSP.
- The timeline data structure.
- Creating a detector to find any inconsistencies within the graphs.
- Creating a calculator to find the number of indeterminant time points within the graphs.

Pending User Stories

- None, all the product owner's user stories were completed.

PROJECT PLAN

This section describes the planning that went into the realization of this project. This project incorporated the agile development techniques and as such required the sprints to be planned. These sprint planning meetings are detailed in the section. This section also describes the components, both software and hardware, chosen for this project.

Hardware and Software Resources

The following is a list of all hardware and software resources that were used in this project:

Software:

- Eclipse IDE for Java Developers
- Subclipse subversion software
- Oracle
- Java JDK
- Zoom
- Trello
- Discord

Hardware:

- Five personal computers
- SVN

Sprints Plan

Sprint 1

- Setup Development environment.
- Review provided materials about jTLEX.
- Review Junit 5 testing procedure.

Sprint 2

- Create the graph data structure.
 - Create constructor functions.
 - Create graph interface.
 - Create functions for graph class.
 - Create ILink functions (addEdge, removeEdge, getEdges).
 - Create INode functions (addVertex, removeVertex, getVertex).
 - Write JUnit Tests for the graph class functions.
- Create method to partition the graph data structure.
 - Create partitioning class.
 - Create main subgraph data structure.
 - Create functions for main subgraph data structure.
- Begin development of the TimeML to TCSP converter method.
 - Read jacop annotation guidelines.
 - Write method for transforming graph nodes to jacop elements.
 - Write method for assigning integers to the shortest solution jacop problem.

Sprint 3

- Continue work on the method of partitioning the graph data structure.
 - Create subordination subgraph data structure.

- Create functions for the subordination subgraph data structure.
 - Create JUnit tests for the methods within the partitioning graph data structure.
- Continue work on the TimeML to TCSP converter method.
 - Write method to transforming each ILink into jacop constraints.
 - Update the method for transforming each graph node to jacop elements.

Sprint 4

- Begin work on the timeline data structure.
 - Create method to add breaking points and connection points to subgraphs.
 - Implement an iterator interface.
 - Implement a timeline class.
- Test the TimeML to TCSP converter against the database of TimeML files.
 - Create JUnit test for the methods of the TCSP converter.
 - Test the TCSP converter with the files in the TimeML file database provided.
 - Produce the total run time of the tests.

Sprint 5

- Continue working on the timeline data structure.
 - Create method to retrieve the main timeline.
 - Create method to retrieve the subordinated timeline.
 - Create a method to get the length of the timeline.
 - Test the methods with the provided graphs.
 - Check the timeline against the solution provided by the TCSP converter's shortest jacop solution.
- Create a method of detecting inconsistencies in the timeline of events.
 - Create the consistency detector interface.
 - Create method to return all inconsistency subgraphs.
 - Create a method to detect inconsistencies within a timeline of events.
 - Create a JUnit test for the inconsistency detector class.
 - Create a method to return the types of inconsistencies that are present within the timeline of events if any are located.

Sprint 6

- Work on a method of locating the total indeterminant timepoints within the graphs.
 - Create an interface for the indeterminacy calculator method.
 - Create a method for returning the indeterminant timepoints of a graph.
 - Create a method to return the percentage of indeterminant links exist within a graph.
 - Create a method to locate all adjacent timepoint links.
 - Create a JUnit test for the methods within the indeterminacy calculator.
- Begin work on the documents for the jTLEX project final deliverable.
 - Create Posters.
 - Create project documentation.
 - Create presentation slides.
 - Create presentation videos.

Sprint 7

- Finish work on the jTLEX project final deliverable for any incomplete files.
 - Create readme text file.
 - Create installation guide.
 - Create Wishlist of features.
 - Create maintenance guide.
 - Create user manual.

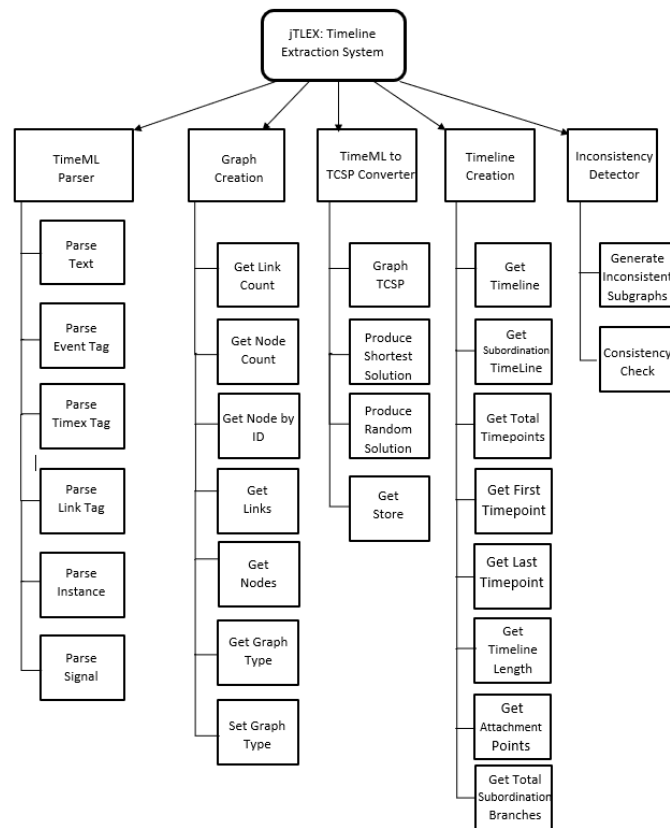
SYSTEM DESIGN

This section contains information on the design decisions that went into this project. The architecture patterns are outlined and explained. The entire system is shown in a package diagram and the subsystems are explained. Finally, the design patterns used in the project are discussed.

Architectural Patterns

Architectural Design	
Name	Description
TimeMLParser	Takes a TimeML file in as a file input stream and parses the times, events, links, signals, instances, and text so that it is possible to order them completely.
Graph	Creates a TimeML graph of events using the information obtained from the TimeMLParser.
GraphTCSP	Translates a TimeML graph into a TCSP and provides the shortest solution the TCSP.
InconsistencyDetector	Locates if the TimeML graph contains any inconsistencies that could prevent it from being solved by the GraphTCSP method. If any inconsistencies are found separates them into their own subordination graphs.
IndeterminacyDetector	Calculates the percentage of timepoint links which are indeterminant within a TimeML graph and returns any of the indeterminant timepoints in the graph.
Timeline	Creates a timeline of events within the TimeML graph using the GraphTCSP method to determine if it possible to create a concise series of events from the given TimeML file.

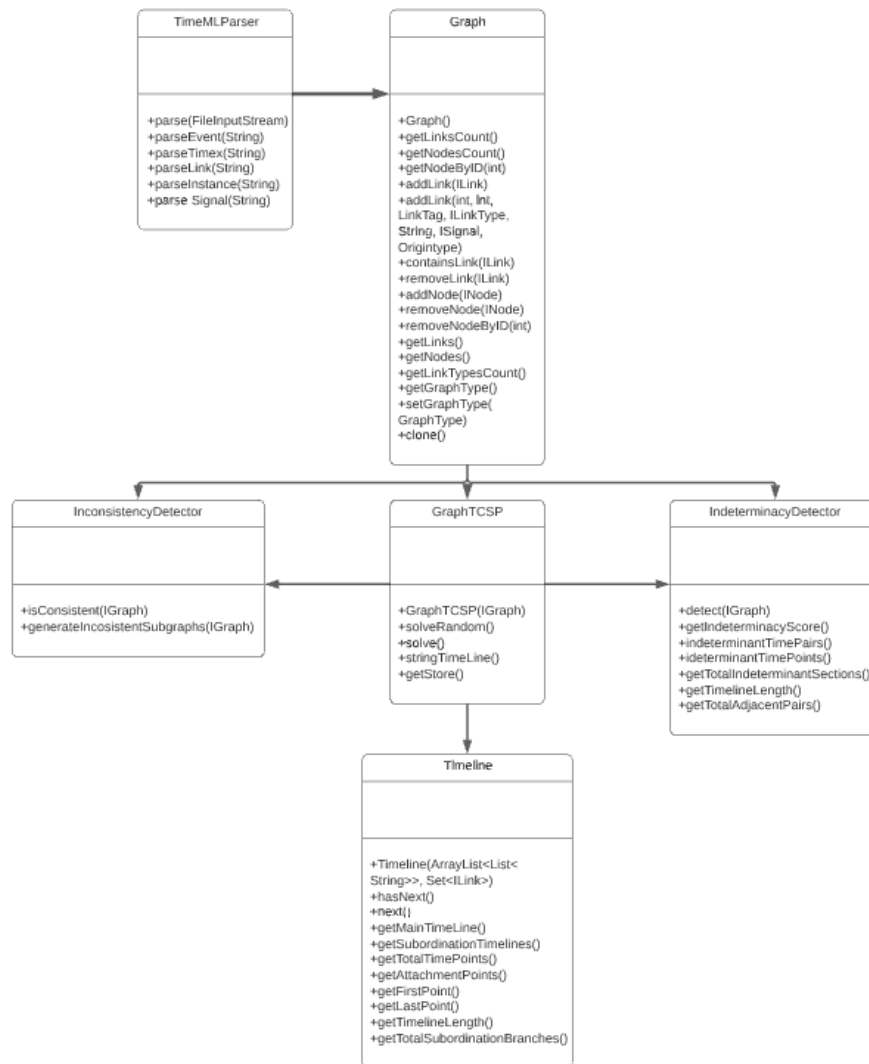
System and Subsystem Decomposition



Deployment Diagram

As a java library jTLEX would not have a deployment diagram as it is not an application in and of itself and is meant to be implemented into other code by importing the jTLEX classes which are going to be used by said code.

Design Patterns



SYSTEM VALIDATION

Test File	testUtils.java
Purpose	To ensure to proper operation of the GraphTCSP constraints generation method
Preconditions	The GraphTCSP method has an existing TimeML graph as input for conversion.
Input	A TimeML Graph.
Expectation	The GraphTCSP method generates a correct list of constraints based on the predetermined list of link type conversion algorithms.
Result	Pass

Test Files	Timebank Corpus
Purpose	To ensure that the TimeML parser method is capable of locating a specific instance in files and retrieve the id of the instance in the file where it was located.
Preconditions	There is a proper TimeML file as input for the method to read.
Input	A list of TimeML files passed to the TimeML parser one at a time.
Expectation	The TimeML Parser will locate the specified instance five times and provide the id of each instance in the file located.
Result	Pass

Test Files	testUtils.java
Purpose	Determine if the Indeterminacy Detector class can return the correct indeterminacy score, list of indeterminant pairings, list of indeterminant timepoints, total number of indeterminant sections, total number of adjacent pairs, and the length of the timeline of events.
Preconditions	There must be a valid TimeML graph presented as input for the Indeterminacy Detector class.
Input	A valid TimeML graph.
Expectation	The Indeterminacy Detector class will return the correct list of indeterminant pairs alongside the listing of all the indeterminant timepoints. The class will also be expected to return the indeterminacy score and timeline length that is expected from the input graph provided.
Result	Pass

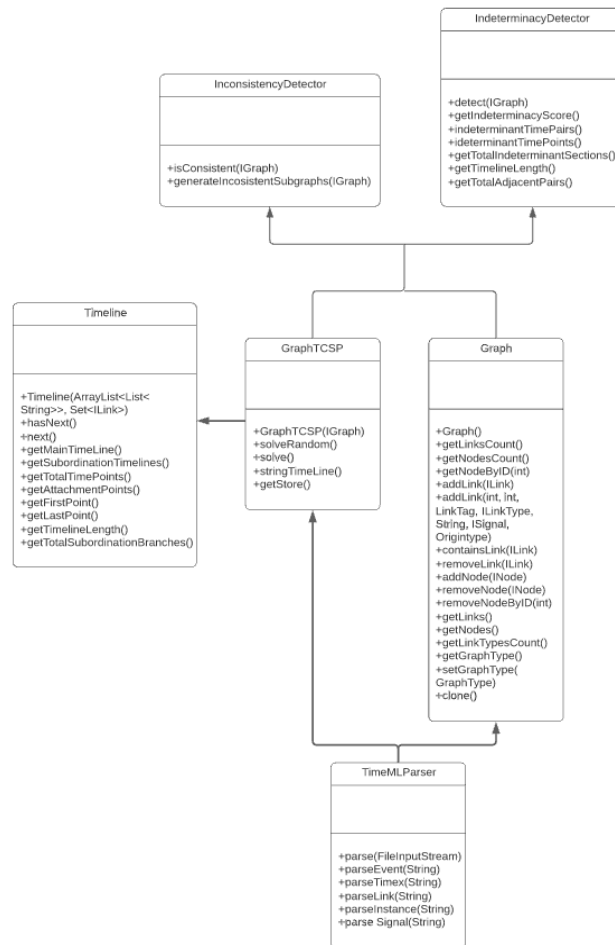
GLOSSARY

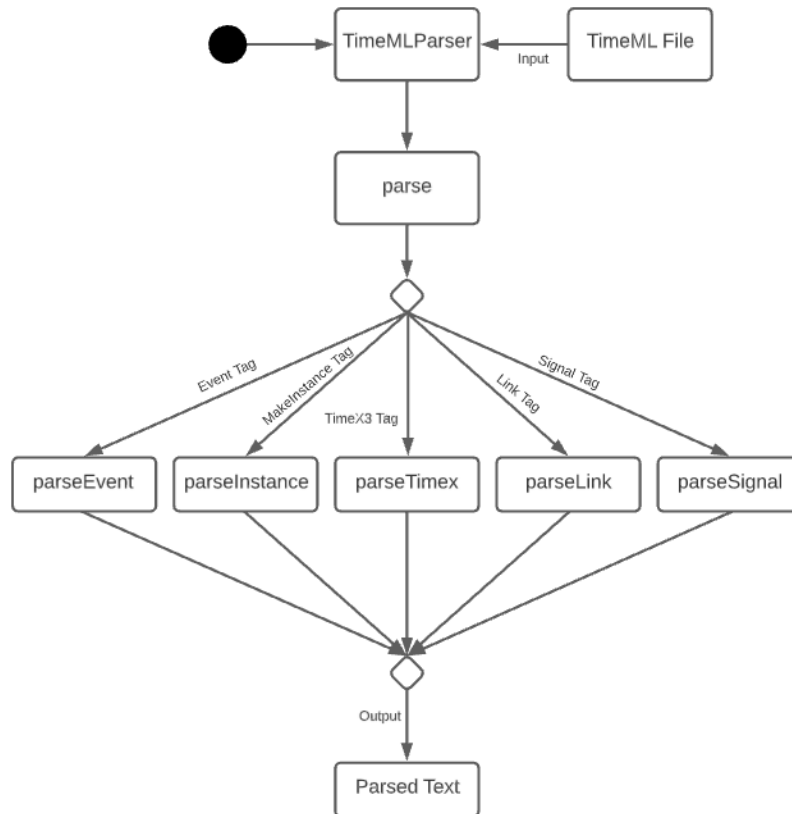
- **TCSP** - Temporal Constraint Satisfaction Problem.
- **TimeML** - A ruleset with the goal of creating a markup language of temporal events within a document.
- **SVN** – A software versioning and revision control system.

APPENDIX

Appendix A - UML Diagrams

Static UML Diagram



Dynamic UML Diagram

Appendix B - User Interface Design

N/A: Due to jTLEX being a java library it does not necessitate a user interface.

Appendix C - Sprint Review Reports

Sprint 2

☒ Checklist Delete

0%

☐ Add links to cards here 🕒 2+ ...

☐  Creating the graph data structure (15 hr)

☐  Writing converter to TimeML to TCSP (60hr)

☐  Partitioning the graph structure (45 hr)

Sprint 3

☒ Checklist Hide completed items Delete

67%

☒  Creating the graph data structure (15 hr)

☒  Partitioning the graph structure (45 hr)

☐  Writing converter to TimeML to TCSP (60hr)

Sprint 4

☒ Checklist Hide completed items Delete

50%

☒  Writing converter to TimeML to TCSP (60hr)

☐  Writing the timeline data structure

Sprint 5

☒ Checklist Hide checked items Delete

100%

☒  Writing converter to TimeML to TCSP (60hr)

☒  Writing the timeline data structure

☒  Writing the consistency detector

☒  Implementing the Indeterminacy Calculator

Sprint 6

☒ Checklist Hide checked items Delete

50%

☒  Implementing the Indeterminacy Calculator

☐  Final Deliverables

Appendix D - User Manuals, Installation/Maintenance Document, Shortcomings/Wishlist Document, and other documents

User Manual

How to Use

jTLEX is a java library made with the intention of taking files in the TimeML format and parsing them to produce a timeline of the events within presented in a means that provides a simpler means of understanding information within. Due to jTLEX being a java library the way in which to make use of the jTLEX library is to download the jTLEX library and add it to your class path. Once the jTLEX library has been added to your class path you will then be able to use the classes from the jTLEX library as you would with any of the default java library classes, by importing the jTLEX class that you wish to use in the code you are currently writing (i.e., import edu.fiu.jtlex.data.GraphTCSP;).

Features

The jTLEX java library contains several classes intended for the purpose of parsing TimeML files. Some of the jTLEX library features are as follows:

- TimeMLParser – a class which goes through a TimeML and parses it to produce the time, event, and link tags from the text within the file to allow for an easier means of understanding the TimeML file.
- Graph – a class which takes a TimeML file and produces a graph with the info from the file.
- GraphTCSP – a class which takes a TimeML graph and converts it into a Temporal Constraint Satisfaction Problem.
- IndeterminacyDetector – a class which locates any timepoint links in a TimeML graph which are indeterminant and returns a list of the indeterminant time points.
- InconsistencyDetector – a class which takes a TimeML graph and determines if it contains any inconsistencies. If any inconsistencies are located splits them of into a subgraph.
- Timeline – a class which creates an ordered list of the extracted events from a TimeML graph.

Installation Guide

jTLEX is a java library for timeline extraction, as jTLEX is a java library the process in which to install the java library requires only a small number of steps.

The following are the steps for how to install and make use of the jTLEX java library:

1. Download the jTLEX jar file to your computer.
2. Load the jTLEX jar file into your java IDE of choice.
3. Add the jar file to your class path.

Once those steps are completed you can make use of the jTLEX class by simply importing the specific jTLEX class you wish to use in the code you are writing.

i.e. (import edu.fiu.jtlex.data.GraphTCSP;)

Shortcomings/Wishlist Document

Shortcomings

We had no shortcomings as we were able to complete all the user stories that the product owners wished to accomplish with this rendition of the jTLEX java library.

Feature Wishlist

The product owners that in the future they would like to create a version of the jTLEX java library which implements visualization aspects to allow for the ability to produce visual representations of the TimeML graphs and timeline produced by jTLEX.

REFERENCES

An Introduction to Constraint-Based Temporal Reasoning, by Barták Roman et al., Morgan & Claypool, 2014, pp. 1–45.

Kuchcinski, Krzysztof, and Radosław Szymanek. *Jacop Library User's Guide*. *CiteSeerX*, 2016, pp. 1-15, 37-38, 45-50, 55-67,
<https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.731.4515&rep=rep1&type=pdf>.

Pustejovsky, James, et al. “(PDF) *The TimeBank Corpus - ResearchGate*.” *Researchgate*, Jan. 2003, www.researchgate.net/publication/228559081_The_TimeBank_corpus.