Graduate Institute of Applied Linguistics

Thesis Approval Sheet

This thesis, entitled

Constituent Charting for Discourse Analysis: Information Model and Presentation Model

written by

Lars Andrew Huttar

and submitted in partial fulfillment of the requirements for the degree of

Master of Arts in Applied Linguistics

has been read and approved

by the undersigned members of the faculty

of the Graduate Institute of Applied Linguistics.

_____

Gary F. Simons, Ph.D. (Mentor)

_____

Shin Ja J. Hwang, Ph.D.

_____

Robert B. Reed, Ph.D.

_____

August 5, 2003

CONSTITUENT CHARTING FOR DISCOURSE ANALYSIS: INFORMATION MODEL AND PRESENTATION MODEL

By

Lars Andrew Huttar

Presented to the Faculty of
the Graduate Institute for Applied Linguistics
in partial fulfillment of the requirements
for the degree of

Master of Arts in Applied Linguistics

Graduate Institute of Applied Linguistics
June 2003

THESIS DUPLICATION RELEASE

I hereby authorize the Graduate Institute of Applied Linguistics Library to duplicate this thesis when needed for research and/or scholarship.

Agreed: _____
(student signature)

Refused: _____
(student signature)

ABSTRACT

CONSTITUENT CHARTING FOR DISCOURSE ANALYSIS: INFORMATION MODEL AND PRESENTATION MODEL

Lars Andrew Huttar

Master of Arts in Applied Linguistics

The Graduate Institute of Applied Linguistics, June 2003

Supervising Professor: Gary F. Simons, Ph.D.

A common task in discourse analysis is the preparation of constituent charts, to make patterns in a text more visible. Domain-specific computer software could make this task easier. But in order for the products of such tools to be shareable and reusable in the long term, the tools must be designed to separate information from presentation, and meet certain other portability requirements (see Bird and Simons 2003).

This thesis demonstrates such a design, using XML documents to encode the information and a DTD to specify the information model. An XSLT stylesheet, representing the presentation model, is used to generate *display charts* (presentation forms) from *abstract charts* (representations of the information in a constituent chart, without regard to presentation format). The information model and presentation model are then shown to be adequate for most constituent charts by encoding three example charts, and then generating presentation forms from them that are accepted as adequate by the authors of the original charts. Consideration is also given to generating abstract charts from annotated texts via stylesheets.

DEDICATION

To my heavenly Father: His grace has brought me safe thus far.

To my wife: for being interested in my interest in XSLT;

and for “doing me good” (Proverbs 31:11-12).

ACKNOWLEDGEMENTS

I wish to acknowledge the gracious help of my thesis committee. Dr. Gary Simons, the committee chair, challenged me with inspiring ideas and often drew me back from the ideal to the possible and necessary. Thank you for taking time for thoughtful review despite many other demands on your time. Thanks also to Dr. Shin Ja Hwang – for teaching me charting and chunking, for answering many questions, and for wise advice on writing. My appreciation is also extended to Dr. Robert B. Reed for contributing from his experience in software tools for linguistics.

I am grateful to Dr. Robert Longacre and Dr. Stephen Levinsohn for their prompt answers to my frequent questions about their column charts; to Dr. Pete Unseth, for kind encouragement, and for offering me "the most important piece of furniture in your office;" and to Paul Headland, for much prayer and patient encouragement. Thanks also to Kent Spielmann and Larry Hayashi, who shared with me their enthusiasm and ideas on software for discourse analysis and linguistic annotation. Thanks are due to my wife Kathy for detailed editorial work on the format of my References section, performed without the benefit of separated information and presentation in the bibliographic data.

I also owe gratitude to several friends and family with whose gracious financial help and encouragement I was able to continue attending the Graduate Institute of Applied Linguistics these four years. Your investment is bearing fruit in my life. May "he who sows and he who reaps rejoice together" (John 4:36-38).

August 5, 2003

CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

ACH	Association for Computers and the Humanities
ACL	Association for Computational Linguistics
AG	Annotation graph
ALLC	Association for Literary and Linguistic Computing
CELLAR	Computing Environment for Linguistic, Literary, and Anthropological Research
CSS	Cascading Style Sheets
DTD	Document Type Definition
HTML	Hypertext Markup Language
IT	Interlinear text
LTR	Left-to-right
PDF	Portable Document Format
RTL	Right-to-left
SGML	Standard Generalized Markup Language
SIL	Summer Institute of Linguistics (International)
TEI	Text Encoding Initiative

URI	Uniform Resource Identifier
W3C	World Wide Web Consortium
WYSIWYG	What You See Is What You Get
XCES	XML Corpus Encoding Standard
XML	Extensible Markup Language
XSL	Extensible Stylesheet Language
XSLT	Extensible Stylesheet Language Transformations

CHAPTER 1.

INTRODUCTION

1.1 Charting for discourse analysis

The term *discourse analysis,* while broadly meaning the study of language in context (Longacre 1996:1), is used to refer to many diverse fields. This thesis is concerned with the study of the linguistic structure of whole texts, that is, with textlinguistics. The literature on discourse analysis and its development is reviewed in chapter 2.

Field linguists performing discourse analysis on texts produce displays of various types to aid in finding patterns, or to summarize their findings: constituent charts, Thurman charts, Semantic Structural Analysis trees, Rhetorical Structure Theory analysis trees, and so on (see Fig. 1 for an example of a constituent chart).

But there is a lack of suitable domain-specific tools to facilitate creation of consistent displays and to help analysts maximize the benefit of their labor. For instance, at a discourse analysis workshop attended by twenty participants in fall of 2001, a survey of attendees revealed that practically all present were using general-purpose tools such as word processors or even pencil and paper to produce their charts and other displays.[1] General-purpose tools have the advantage of being readily available, but their disadvantages are considerable: they

[1] Personal observation. Discourse Analysis Workshop, International Linguistics Center, Dallas, TX, 24 Sept.-5 Oct. 2001.

fail to address the problems of maintenance of the data, sophisticated querying, and reuse of the data by other programs.

	Introducer		Preposed Dep Cl				Independent Cl		
	S-in	S-med	Conj	S	P	0	S	P	0
1	In a minute						a third slave	was struggling	in the air.
2							It	was	dreadful.
3							I	turned	my head away a moment,
		and	when	I	turned	back	I	missed	the king!
4							They	were blindfolding	him!
5							I	was paralyzed;	
6							I	couldn't move,	
7							I	was choking,	
8							my tongue	was petrified.	
9							They	finished blindfolding	him,
10							they	led	him under the rope.
11							I	couldn't shake off	that clinging impotence.

Fig. 1. A simple constituent chart of a Mark Twain passage.
From Longacre and Hwang (n.d.). Used by permission.

When the software uses proprietary formats, reuse is even more limited as the data may become inaccessible after five to ten years when the software that reads the format becomes obsolete (Bird and Simons 2003:3). For the data to remain accessible long-term and to a wide audience, linguistic analysis tools should use open standards (see also Cover 2003) so that researchers can access the data without being forced to use one company's software. The data takes on even wider usefulness when free, open-source software tools are available for manipulating it (Raymond 2003).

1.2 Separation of information from presentation

Another fundamental requirement for reuse of data is the separation of information from presentation. Since humans and computers have very different requirements for the

accessibility of information, a chart designed to be visually effective for humans tends to be difficult for computers to interpret. Moreover, human users themselves have a variety of needs for presentation of the same information. Therefore, to gain full benefit from data, it should be encoded in a way that computers can easily process for various purposes, including flexible rendering into diverse presentation forms. Markup systems make data easy for computers to read (unless they are proprietary), but only descriptive markup makes the data accessible as information, so that it can be reworked for new purposes. *Markup* is defined as "the property of textual data that has to do with how the information above the character strings themselves is represented" (Bird and Simons 2003:§3.2). The following comparison of descriptive markup with presentational markup may elucidate both.[2]

A typical word processor of today intersperses data with directives that encode italics, font colors, table line thicknesses, and so on.[3] These directives are *presentational markup* as distinct from the text content. Italics might distinguish, e.g., a gloss line from the vernacular text in an interlinear display. This makes the presentation attractive to the human eye, but difficult for a computer to determine what the function of each piece of data is. For instance, italics might be used for a number of different purposes, and there is nothing to tell a computer what italics stands for in a particular display. The Hypertext Markup Language (HTML, the main language of web pages) is another example of a (mostly) presentational markup system.[4]

---

[2] See also Coombs et al. (1987) for a comparison of various types of markup.

[3] This markup may take the form of binary codes that are invisible to the user, or that are visible as special symbols. Coombs et al. (1987) refer to these as *concealed* and *disguised* markup respectively.

[4] Some have categorized HTML as descriptive markup rather than presentational, and there is some truth to this: while some tags, such as `<font>`, are purely presentational, others, such as `<h3>`, are descriptive. However, HTML is inadequate as a markup system for linguistic analysis because (a) its vocabulary is fixed: you cannot

Descriptive markup takes the other approach, identifying the function of each data item instead of its appearance. For example, the Extensible Markup Language (XML; see Bos 1999) uses matching *start* and *end tags* with names corresponding to the function of a datum. In XML, a gloss line would be marked not as italic text but as a series of gloss elements (using tags such as `<gloss>…</gloss>`). A computer-readable definition of the markup system, called a Document Type Definition (DTD), can also be provided. This combination of descriptive markup with the ability to provide a definition of the markup system makes the data easily accessible to computer software for many purposes, including those unanticipated by the author of the data at the time it was created.

However, the processing of data is of limited worth if results cannot ultimately be presented to humans in a readable form. Thus a descriptive markup system must be accompanied by a process for rendering the data into a presentation form.[5] The Extensible Stylesheet Language (XSL) serves this purpose by providing a framework for stylesheets that transform XML documents into presentation forms such as HTML or PDF (Portable Document Format, a publishing format from Adobe Corporation). Thus, the combination of descriptive markup with a rendering process gives the researcher the best of both worlds: high-quality presentation forms, and reusable data.[6]

---

create a `<relative-clause>` tag if you need to describe certain data as such; and (b) its grammar rules are so lax and so inconsistently observed that it is difficult to write software that interprets it all in a standard way.

[5] This point is the last of six fundamental requirements for linguistic computing developed in Simons (1997a) and Simons (1998).

[6] On the other hand, there is the ease-of-use issue: typically, encoding data functionally for later rendering requires more abstract thought and planning, and appeals less to visual thinkers. Some projects such as Quill (Chamberlin et al. 1988) and Serna (Syntext 2003) have sought to address this issue by attempting to combine the visual feedback of a WYSIWYG interface with the flexibility of a descriptive markup language.

A *presentation model* is a specification of how a particular type of document[7] can be rendered into a presentation form. (A *presentation form* is a particular way a document is rendered visually. For example, the presentation form of a chart might take the form of a word processor document, an HTML file, or ink on paper.) A stylesheet is one way to express a presentation model. An *information model,* by contrast, specifies for a given type of document what concepts or information items may or must be present in such a document, and the relationships among the items. (An information model is also known as a *conceptual model*.)

1.3 The notion of abstract chart

An *abstract chart* (properly *abstract constituent chart*) is defined as a collection of the information contained in a specific constituent chart, independent of any particular output format (such as HTML or PDF). That is, an abstract chart is the set of pieces of information that a visual chart is intended to communicate. An abstract chart is encoded using descriptive markup, not presentational markup. The *abstract chart model,* that is, the information model for an abstract chart, specifies what pieces of information an abstract chart may or must include, and how they are to be related to each other. Once an abstract chart document has been created, a stylesheet can be used to render it into a presentation form.

There are significant benefits to modeling a constituent chart in terms of information rather than merely presentation form. Information models for discourse displays like constituent charts will lay the groundwork for the development of widely usable domain-

[7] In this thesis, as in markup terminology, *document* means a set of abstract data (usually conforming in structure to a specific *document type*), not a presentation form of that data.

specific software tools for discourse annotation, producing highly reusable data. Unlike general-purpose tools, software specific to discourse annotation will be able to record and use the information produced in discourse analysis, as opposed to merely recording the presentation associated with the information. Such software could take advantage of this information to assist with otherwise tedious tasks like searching for the presence of data in a particular slot (for example, finding all participant references in dependent clauses in a chart and presenting statistics on the forms they take), toggling the highlighting of all mainline verbs, or making structural changes in a tree (for example, moving one constituent up a level, a time-consuming task if a word processor table or drawing tool is used to model the tree).

An additional use for such an information model would be as a benchmark by which to evaluate existing or proposed discourse annotation software. The model would give a framework for determining to what degree the tool or markup system covers the concepts involved in the task.

Moreover, tools based on information models could offer a variety of views of the annotation and text (i.e. they could offer a rich set of presentation models). This would allow discourse researchers with varying interests to collaborate more easily. If Researcher B wants to build on Researcher A's results, creating a modified version of A's display, he or she need not go through the entire chart and make tedious consistent changes in order to fulfill B's different display requirements. Instead, an adjustment of display parameters or a different view selection can produce a new display based on the same (or supplemented) annotations. Conversely, Researcher A can then work in any new annotations B has added, without having to manually convert all of B's displays back to A's preferred style.

A final benefit of tools based on an information model is the ability to export the annotations to other useful forms, such as an interchange format (e.g. an annotation graph [Bird and Liberman 1999a] or XCES annotation framework [Ide and Romary 2003]), or an archival format used by a text corpus. See Simons (1997a) for more on the benefits of information modeling over presentation modeling.

1.4 Information models and presentation models in XML

As can be seen from the above discussion, a DTD essentially encodes an information model for an XML document.[8] It does so by specifying what kinds of information (elements and attributes) are allowed or required in a given document type, and in what relation to each other they must occur. Moreover, an XSL stylesheet can encode a presentation model, in that it describes a transformation from the information in an XML document to a presentation form. In other words, it specifies how XML data will be displayed.

The main products of this thesis are an abstract chart information model (expressed as a DTD) and a presentation model (expressed as an XSL[9] stylesheet). The diagram in Fig. 2 illustrates how the pieces in this puzzle are related.

In this diagram, solid-bordered rectangles represent *instance documents* (that is, collections of data specific to a particular chart), while double-bordered rectangles represent models (which govern charts in general). Rectangles with dotted borders represent processes.

---

[8] To a reasonable degree. DTD's do have limitations in their expressive power, compared to full programming languages. Their potential successors, XML Schemas, are more expressive but still not always to the degree desired.

[9] The stylesheet described in this thesis properly consists only of XSLT code, that is, it does not use the Formatting Objects (FO) component of XSL.

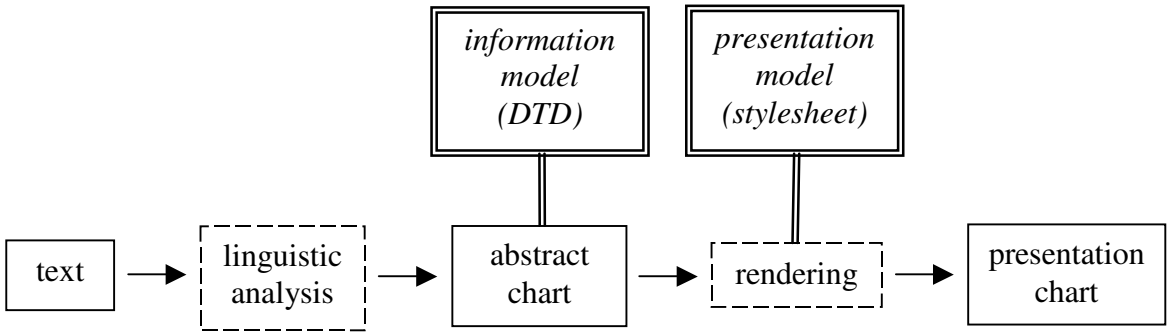

Fig. 2. Overview diagram.

The diagram shows that an abstract chart (an XML instance document) is created as the result of linguistic analysis,[10] starting from a base text. The abstract chart DTD defines the structure of the abstract chart, ensuring that it conforms to the information model.[11] Then a stylesheet processor renders the information in the abstract chart into a presentation chart, guided by the rules specified in the rendering stylesheet. The presentation chart may be, for example, an HTML or PDF document.

How the abstract chart is initially created is beyond the scope of this thesis, but one may imagine a chart creation software tool (perhaps something like a spreadsheet) that assists a linguist in producing the abstract chart. Alternatively, a more general linguistic annotation[12] software tool might help the linguist mark up the base text with regard to constituent

---

[10] This analysis envisioned here is done by a linguist, not by computer software—at least not until significant advances are made in computational linguistics—although the process might be facilitated by software.

[11] Conformance checking, known as *validation,* is performed by a validating XML parser, which rejects any document that lexically or syntactically fails to abide by the constraints specified in the DTD. In the above diagram, validation may be done when the stylesheet processor opens the abstract chart, before rendering.

[12] "'Linguistic annotation' covers any descriptive or analytic notations applied to raw language data. The basic data may be in the form of time functions – audio, video and/or physiological recordings – or it may be textual." (Bird and Liberman 1999:1). In this thesis the basic data is assumed to be a written (and electronically encoded) text. Leech et al (1998:1.1) define annotation, in contrast to *representation*, as "additional levels of linguistic information which are added to the orthographic transcription." Examples of annotation would include identifying a group of words as a noun phrase, marking a phrase as the subject of a clause, or tagging a text as belonging to the "hortatory" genre. The CES documentation (EAGLES 2000:0.2.4) divides the two somewhat differently, considering the marking of linguistic units (e.g. paragraphs, quotations, names, etc.) to be *primary data,* not linguistic annotation.

structure and other information, whereupon an abstract constituent chart could be generated from this markup. See §6.1 for further discussion.

A number of promising linguistic annotation and display tools are available or under development. Many of them use open standards, open-source software, and XML encoding to separate information from presentation. However, at present none of them facilitates creation or rendering of a constituent chart for discourse analysis. Design of a suitable information model for constituent charts would contribute to the development of software to help discourse analysts and students make constituent charts and get the most out of them.

1.5 Methodology and scope

This thesis focuses on constituent charting, to the exclusion of Thurman charting, tree diagramming, and other discourse tasks for which software tools are needed (but see §6.1 for future directions). This focus was based partly on Spielmann (2000) and on my own experience of the need for intelligent charting tools in a discourse analysis class.[13]

The approach taken in this thesis is as follows. Three existing constituent charts, created by three different analysts, were chosen as input to the process: two of them published, and one of them having been used for some years in teaching discourse analysis. As such, each could be confidently accepted as an example of a well-formed constituent chart. The charts are shown in Fig. 3, Fig. 4, and Fig. 5.

Fig. 3 is a chart created by Levinsohn, published in Longacre and Levinsohn's (1978) article which recommends this type of chart as an aid to "field analysis of discourse." The

[13] Note that this thesis only lays some groundwork for the design of such tools. Further work will be needed before constituent charting software is actually available to be used by non-computer specialists.

chart demonstrates pre-analytical[14] techniques described in the article, using an Inga language text with interlinear English glosses. Fig. 4, a chart by Longacre (1992), gives a simple display of a narrative text in English, "Ordeal in the Winter Woods." The third chart, part of which is shown in Fig. 5, is the most complex. This chart of an English short story, "Little Hans," was developed by Hwang (1997) over several years as an example for teaching discourse analysis to graduate students.

These three original charts were taken as representative of constituent charts currently in use for analyzing discourse. A list of requirements for the abstract chart model was then created based on the information presented in the original charts. Based on these requirements, an abstract chart information model was designed that would be capable of encoding that information. A DTD was then created to encode this information model.

Each chart was then encoded into an XML document conforming to the abstract chart model DTD. A stylesheet was written in XSL to render the abstract chart instance documents into presentation forms coded in HTML. After each chart was rendered, the resulting HTML display form was sent to the author of the original chart for evaluation to answer the question, "Does the rendered display chart still communicate clearly the information that the original chart was meant to convey?" Once all the original authors were satisfied with the quality of the rendered display charts, the abstract chart information model (DTD) and presentation model (XSL stylesheet) were judged adequate. During this process, corrections or updates were made to each of the abstract charts and the models according to the wishes of the authors until each author was satisfied with his or her chart.

---

[14] Stephen Levinsohn (personal communication, 2003) notes that the constituent chart is presented in workshops as a pre-analytical tool, that is, it precedes discourse analytical decisions. The chart does not of course precede *grammatical* analysis, some of which is required before constituents can be identified.

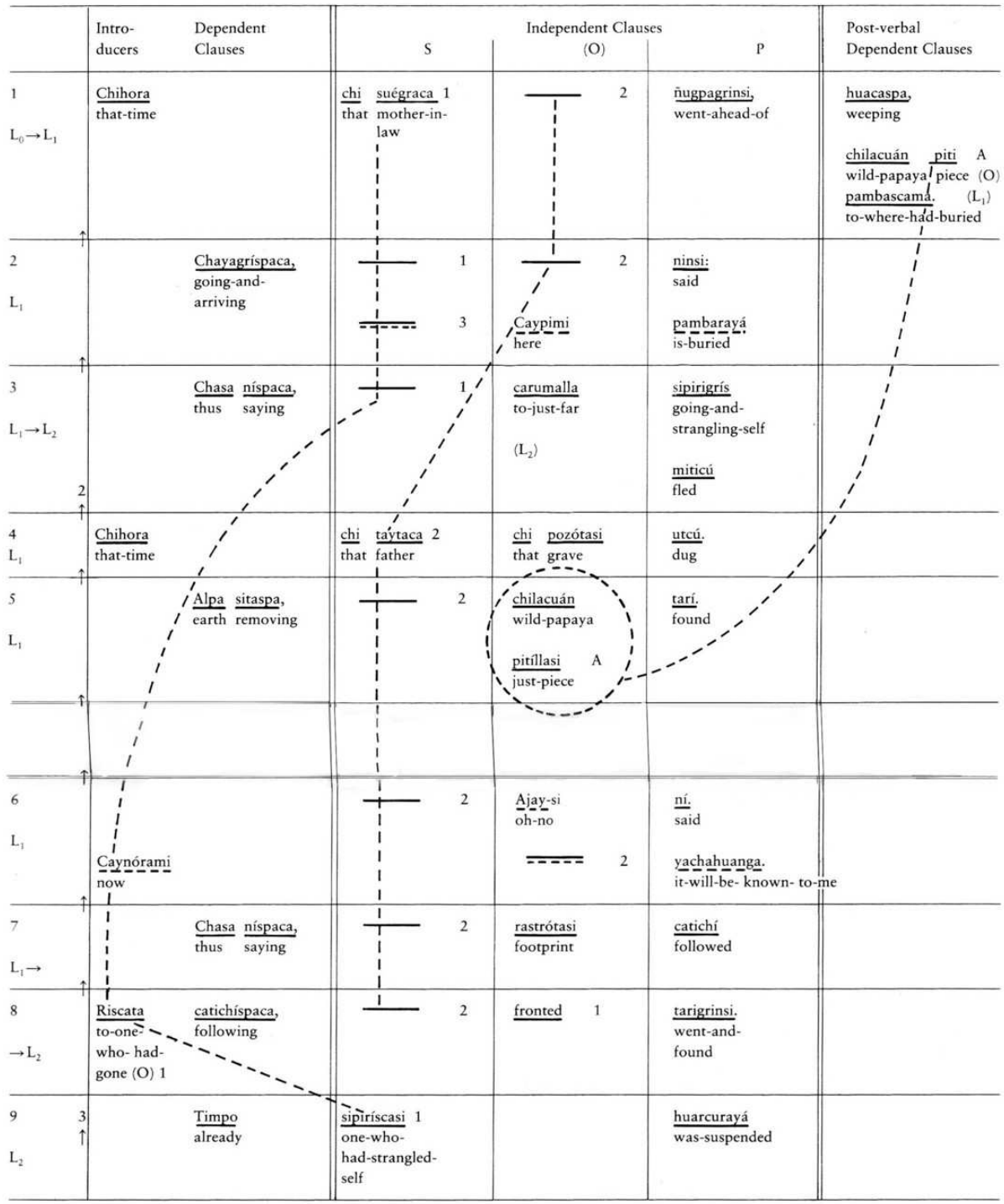

Table I – Initial display of discourse

Fig. 3. Original Inga chart.
Excerpted from Longacre and Levinsohn (1978). Used by permission.

	Preposed	S	V	O	Postposed
1.	Early in the afternoon of the tenth day, a Friday	Robin	suddenly heard	the swish of skis	
		He	saw	a cross-country skier	on a course that would bring him about 100 feet away.
3.		Robin	shrieked	"Hello"	
4.		The skier	was	22-year old, John Steinmetz	a state-parks life-guard in Sta. Cruz.
5.	Also a lone skier,	he	was trying		
			to come to grips with	a problem of his own.	
6.	A few weeks earlier,	he and others	had failed		
			to recuscitate	a drowned swimmer.	
7.		He	was ridden	with remorse.	
8.	Some quality in Robin's shout		made		
		John	swerve to a halt and pole his way	back.	
9.	<<How('s)	it	going?>>		
		he	asked	the injured skier.	
10.		<<I	have	a broken leg and ankle.>>	
11.	<<How long (have)	you	been out here?>>		
12.		<<--------	Ten days ------->>		
13.		<<It	took	me	about 45 minutes
		to get here			from McCabe Lake>>
		John	said.		
14.	<<Well>>	Robin <<it	answered, took	me	six days.>>

Fig. 4. Original "Ordeal" chart.
Excerpted from Longacre (1992). Used by permission.

NOTES	Introducers		Preposed Dependent Clauses				Independent Clauses			Postposed Dependent Clauses			
	S-initial	S-med.	Conj	S	P	O, etc.	S	P	O, Comp, Others	Conj	S	P	O, etc.
1 Stage							The winter afternoon	was	dark and grey over Old Strasbourg.				
2		and					Little flurries of snow a biting wind	came whirling down blew	between the chimneys in the narrow streets.				
3	Above the roofs,			0	rising	high into the clouds,	[in O]	stood	the great cathedral,		its stones	0	dim in the gathering gloom, its windows catching the lights within.
4							Fine people	were hurrying	up the broad steps --ladies with furs, gentlemen in splendid attire,		many of them coming		in their carriages.
5							Little Hans	watched	them.				
6 Durative		and		0 0 0 0 0 0	Perished with cold, ragged, an unwanted bit of humanity,		he 0 <*he*	snuggled wished dare go	between two buttresses -- a retreat from the wind-- into the cathedral {where all was warm and bright, and where (as he could dimly hear) the organ was pealing loudly.}>				

Fig. 5. Original "Little Hans" chart (first page).
Excerpted from Longacre and Hwang (n.d.). Used by permission.

1.6 Organization of thesis

After a brief review of the literature (chapter 2), the requirements for the abstract chart information model are presented (chapter 3). In chapter 4, an information model that meets these requirements is presented (expressed as a DTD). Chapter 5 shows and discusses some example sections of an XML-encoded abstract chart document, briefly describes the XSL stylesheet that guides the rendering process, and then presents the rendered display charts. Finally, chapter 6 reviews what has been accomplished, including an experiment in generating an abstract chart from an annotated text, and suggests directions for further work.

CHAPTER 2.

REVIEW OF THE LITERATURE

Before requirements for an abstract chart model are presented, a brief review of background literature on discourse analysis and discourse annotation software will help set the stage.

2.1 Discourse analysis

2.1.1 Scope of this section

The phrase *discourse analysis* covers a very broad and diverse field of study. Schiffrin et al. (2001) groups the many approaches to discourse into three categories: (1) anything beyond the sentence; (2) instances of language in actual usage as opposed to constructed examples; and (3) social use of language with the result of promoting ideologies or power structures.

This thesis is concerned only with (1) and (2) above: discourse analysis in the sense of textlinguistics, that is, linguistic analysis of a whole, naturally-occurring text, including structure above the sentence level. In particular, the development of two specific types of column charts used in discourse analysis, namely Thurman charts and Longacre-Levinsohn charts, is traced in this literature review; other branches of discourse analysis are beyond the scope of this study.

2.1.2 The origins of discourse analysis

Linguistics under Bloomfield (and later Chomsky) long ignored or gave scant attention to the structure of language above the level of the sentence (Harris 1952, Longacre 1979). But in 1952, Zellig Harris began to buck this trend by developing a method of "discourse analysis" that examined connected sequences of sentences in a text to discover equivalence chains, and thereby, to unearth its content structure. While Harris's work was influential, his method has not continued to be widely used. Another tributary to discourse analysis, from the field of narratology, came from Propp (1928) and Dundes (1962-64) who examined the structure of folktales in a way that, as Lakoff (1964) pointed out, could be generated by a simple grammar. In literary theory, Benveniste (1974) described, among many other topics relevant to discourse, the significance of the choice of pronouns and tense/aspect in narrative.

The first forays into discourse grammar, at least on this side of the Atlantic, were attempted by Bible translators, in the process of documenting the grammars of Native American languages – starting with Loriot (1970) (written in 1958), and Pickett (1959).

In 1964, several publications urged the consideration of discourse-level structure. Longacre (2002, personal communication) credits Weinrich and Gleason with being the key pioneers of discourse analysis in Europe and in the United States, respectively. Weinrich (1964) described a need for a "textual linguistics" (Partridge 1995:6, Longacre 1993:51), to address a theoretical gap. Gleason's 1964 article made the radical assertion that a paragraph structure grammar could account for patterns in his field data more simply than a sentence grammar. Other articles approached beyond-the-sentence structure from the points of view of

poetry (Bateson 1964), composition (Pike 1964a), tagmemics (Pike 1964b), generative grammar (Lakoff 1964), and computational linguistics (Jacobson 1964, Harper 1964).

2.1.3 Gleason, Pike, Longacre, and the constituent chart

Gleason and his successors (Taber 1966, Cromack 1968, Stennes 1969) went on to produce the first full-fledged discourse grammars of languages, again, in the pursuit of quality bible translation. Pike's (1954, 1964b, 1967) and Longacre's (e.g. 1965) work on tagmemics laid further groundwork for a major school of discourse analysis within American linguistics (Grimes 1975). This school focused on analyzing and classifying surface structure patterns found in text. Longacre was a leading contributor in this movement, authoring or editing volumes of discourse studies of languages in the Philippines (1968), Mexico (Reid et al. 1968), Papua New Guinea (1972), South America (Longacre and Woods 1976-77), and Africa (1990), and applying methods of discourse analysis to dozens of languages around the world (see Longacre 1996:27). Longacre and his colleagues, including Stephen Levinsohn, have conducted numerous field workshops aimed at encouraging the application of discourse analysis to languages under study. Out of these workshops emerged a method of text charting designed to make discourse patterns more easily visible (Longacre and Levinsohn 1978). The Longacre-Levinsohn chart, also known as a constituent chart (Hohulin 2001), was an extension of earlier clause analysis charts (Longacre 1964:46). Constituent charts consist of columns corresponding to clause or sentence constituents, arranged in the order in which the constituents most commonly occur. Constituents are arranged in the same linear order in the chart as they occur in the text; if any constituent occurs out of the usual order, it cannot be placed in its usual column, and therefore stands out visually as a variation requiring further

study. For example, if an SVO clause appears in a mostly-VSO text, either the S or the V constituent must be displaced from its usual column. Patterns of such displacements can be detected more easily when they are laid out, aligned in a chart.

The method of using a constituent chart as an aid to analysis is presented in Dooley and Levinsohn (2001), and in Hwang (1993) in modified form. Hwang (1997) and Longacre and Hwang (1994) give examples of the use of a constituent chart in the analysis of specific narrative texts.

2.1.4 Other approaches to discourse charting

Gleason (1968) and his successors established an important beachhead into analyzing discourse by exploring the distinctions among *kinds of information* communicated by various parts of a text (Grimes 1975:33). For example, material in a discourse may refer to events, participants, or background information. The separation of material into these "bands" of information is referred to as band analysis, which Longacre (1981) later developed into spectrum analysis. The latter organizes bands of information into a cline from most dynamic to most static, based in part on Hopper and Thompson's (1980) groundbreaking work on "transitivity parameters."

Thurman (1975) and Grimes (1975:82ff) developed Gleason's and Cromack's diagrams into the so-called Thurman chart, which consists of columns corresponding to the bands of information contained in the text under study. Although Thurman charts resemble constituent charts in their column-oriented layout, unlike constituent charts they are not constrained to preserve the order of the original text. Thurman charts are beyond the scope of this thesis, but some possibilities for supporting them are presented in §6.1.

While constituent charts and Thurman charts typify table-like displays, other discourse analysis displays use a tree diagram of some sort. One example is the diagramming of *interclausal* and *intersentential relations* (Longacre 1996:51-122), showing the constituent structure of a text and the relations between constituents. This information may be displayed as a tree or as an indentation diagram (see for example Hwang 1989), laid out horizontally (with the tree's root on the left or right). *Semantic structural analysis* (Beekman, Callow, and Kopesec 1981) uses similar tree diagrams to show a modified form of interclausal relations. Another kind of tree diagram, in which the display is vertical (root is at the top), is the kind used for *rhetorical structure theory* (RST) displays (Mann and Thompson 1986).

The information in these tree-shaped displays shares several common features, such as hierarchical containment, slot and class information,[15] and a nuclear vs. peripheral distinction for constituents. For this reason it is likely that a single information model and software tool with multiple configurations could be useful for working with all of them.

Another type of discourse analysis display does not consist of an arrangement of the text, but of a presentation of statistics or results of computations on parts of the text. An example would be topic continuity statistics for participants in a text (see Givón 1994).

However, these other approaches are beyond the scope of this thesis. Further information on discourse charting software needs can be found in Spielmann (2000).

[15] Each unit in such a tree has both *slot* and *class* (Longacre 1996:269ff; definitive statement in Pike 1967). For example, in an intersentential relations diagram, a given unit may fill the slot of "Simul Thesis 1" with respect to its parent unit, while having a class of "Sequence Paragraph." A unit's class determines what slots are available for its constituents (see Longacre 1996:120 Diagram 4.2).

2.2 Use of computers for linguistic annotation and discourse analysis

This section reviews the development of the use of computer software and text markup for research on language and literature in general, and discourse analysis in particular.

2.2.1 The beginnings of computational linguistics

Almost as long as computers have been used in research, they have been used to facilitate the study of language and literature. In 1949 Father Roberto Busa, considered the "founder of literary and linguistic computing" (Vetch 2001), began pioneering techniques using card-sorting machines to encode a complex text corpus. His final product was a comprehensive concordance of the works of Thomas Aquinas. By the 1960's the use of computers in the humanities was widespread (Harbin 1998; see also Grimes 1965).

2.2.2 The development of standardized markup languages

From early on, a critical consideration was how to most effectively encode the language data that was to be studied; in other words, what kind of markup should be used. As the amount of encoded language data increased, the potential benefits of being able to reuse that data became more obvious, so researchers needed ways to make markup more portable—that is, less dependent on a particular computing environment for its use. As was discussed in chapter 1, descriptive markup made encoded data more reusable by separating information (the more broadly useful part of the data) from presentation (the more environment-specific part). The movement toward using descriptive markup is said to have begun with a presentation by William Tunnicliffe, of the Composition Committee of the

Graphic Communications Association (GCA). This committee developed the so-called "GenCode(R)" concept, advocating *generic coding*, that is, descriptive markup (Goldfarb 1996).

The generic coding concept was developed into Generalized Markup Language (GML) by Charles Goldfarb, Edward Mosher and Raymond Lorie (first published as Goldfarb 1973). Besides descriptive markup, GML also provided formal document type descriptions[16] (Goldfarb 1996). GML later served as the basis for the creation of the more powerful Standard Generalized Markup Language (SGML), which became a draft standard of the American National Standards Institute (ANSI) in 1980, and an official standard of the International Organization for Standardization (ISO) in 1986 (SGML Users' Group 1990:§3). SGML has been used successfully in many projects, especially in text processing.

In 1990, Tim Berners-Lee began working on the World Wide Web initiative at the European Organization for Nuclear Research (CERN). For web page markup he developed HTML to encode hypertext constructs and document elements. HTML was designed as an "SGML application," that is, a language conforming to SGML rules. The first formal specification for HTML was released in 1995 (Berners-Lee and Connolly 1995), by which time HTML was already being used on thousands of web sites ("The Stats Map of Net History," Gromov 2002:8).

2.2.3 The development of XML

As the World Wide Web mushroomed in popularity, the question of what markup language standards would be used for data exchange became pressing. SGML was

[16] The basis for SGML's document type *definitions*.

considered by many to be too complex and difficult to implement for widespread application on the Web. HTML, while simpler than SGML, was not extensible: it did not allow an application developer to define new markup vocabularies (tag sets) to describe new kinds of data. Moreover, widespread informal implementation and nonstandard enhancement of HTML made the language very difficult to check for strict conformance to a DTD, one of the important benefits of a standard markup language.

To fill the need for a simpler but extensible markup language for the web, the World Wide Web Consortium (W3C) developed XML, a subset of SGML. The XML 1.0 specification[17] was released in 1998 (Bray et al. 2000). In the five years since then, the usage of XML has far outpaced that of SGML, especially in the hard sciences and in the business world.[18] Many software tools for working with XML are already available at little or no cost.

Three further developments in the world of XML deserve mention. One is the Extensible Stylesheet Language (XSL). In the information vs. presentation paradigm, where XML is used to encode information, XSL is a language for expressing the presentation model needed to provide displays of that information. XSL Transformations (XSLT), a subset of XSL, is used for that purpose in this thesis. XSLT became a W3C Recommendation in 1999 (Clark 1999).

---

[17] The simplicity of XML relative to SGML is reflected in the size of their respective specifications: twenty-six pages versus over five hundred (Sol 1999).

[18] One indication of XML's widespread use is that a web search engine gives some 19 million results for the term XML, while SGML nets less than a tenth of that number (Google 2003). InfoWorld, a business magazine, gives over a hundred times more results for a search on XML than on SGML (InfoWorld 2003). A search of articles in the journal Literary and Linguistic Computing shows that in the last three years, four articles have mentioned XML, and one, SGML (Association for Linguistic and Literary Computing 2003). Another indicator is the decision of both the Text Encoding Initiative (TEI) and the Corpus Encoding Standard (CES) to support XML. The reasons cited: "in order to make use of the many XML tools becoming available" (TEI Consortium 2001) and "because the XML framework provides us with means to go well beyond the capabilities of SGML" (Ide and Suderman 2002).

Another important development for XML is the advent of more sophisticated systems for describing XML document types and their constraints. XML Schema, a fairly new Recommendation (Fallside 2001) from W3C, offers greater functionality than DTDs in several areas. One of the most important for interchange of data throughout a diverse research community is that of extensibility. XML Schema allows element types in one schema to be extended in another.[19] XML Schema is not however used in this thesis, for reasons discussed in §6.1.

The third development with implications for interchange of data is the Semantic Web. In any diverse and changing research community, there will probably always be a variety of terminologies (or tag sets) used to encode any given type of data, resulting in confusion when an attempt is made to compare or combine data from multiple sources. As noted by Bird and Simons (2003:17), if terminologies are mapped onto a common ontology of terms, different encodings can in principle be converted into mutually-compatible forms. The Semantic Web, using the Resource Description Framework (RDF) and the Web Ontology Language (OWL), provides a means of defining ontologies and describing mappings between terminologies (Miller 2003). RDF became a W3C Recommendation in 1999 (Lassila and Swick 1999).

2.2.4 Standard languages for encoding text and annotation

SGML and XML are general-purpose markup languages; in themselves, they provide only a language skeleton and the means to fill in the blanks. They are often described as "meta-languages," i.e. languages for describing languages. In order for them to be used for a

[19] Certain types of extension are possible with DTDs, but this is achieved either through significant complexity (as with the TEI "Pizza Chef" [Burnard 1999]), or by losing control (within the DTD) of constraints on the extensible parts (Dashofy et al. 2001:3.2).

specific application, a vocabulary (tag set) and structure must be specified, typically using a DTD or other schema description. The Text Encoding Initiative (TEI) and one of its refinements, the Corpus Encoding Standard (CES), define two such SGML languages.

In 1987, the TEI was begun by a group of associations concerned with computing and the humanities in order to address the need for specification of a standard markup language for the encoding and interchange of text (TEI Consortium 2003b). The main product of the TEI, released in 1994, was the TEI guidelines, known as TEI P3 (Sperberg-McQueen and Burnard 1994): Detailed recommendations for the encoding of "textual material of all kinds in all languages from all times" (TEI Consortium 2003a). TEI P3 included a large, modular SGML DTD, consisting of core tag sets and additional, optional ones tailored to the needs of specific applications such as language corpus creation and textual criticism. While the TEI was originally designed in SGML, it has now been made available in XML as well. The current version of the TEI Guidelines is P4 (Sperberg-McQueen and Burnard 2002).

While the TEI provided comprehensive guidelines for encoding practically any kind of text, CES was created to focus more specifically on the needs of language corpus creation. CES is TEI-conformant (EAGLES 2000). It consists of a selection of TEI tag sets relevant to corpus encoding, plus extensions for aspects not covered by TEI. Because it is much smaller than TEI, is geared toward the encoding of text and annotation for linguistic corpora, and covers large discourse units as well as morphosyntactic units, CES is a good candidate for a standard encoding to use for discourse analysis. If CES were made available as an XML application instead of SGML, it would be even more useful, given the availability of XML tools. The Corpus Encoding Standard for XML (XCES) is now under development and is in

beta stage (Ide and Suderman 2002). For extensive discussion of the merits of TEI and (X)CES, see Bański (2001:§1-3).

There are many other SGML and XML languages for encoding annotated text, but TEI and CES seem to be the ones with the most widespread acceptance as standards.

2.2.5 Existing software projects related to discourse annotation

In this section we review a few of the software projects, conceptual models and coding schemes relating to discourse annotation and display that have been worked on so far. The potential list is very large, so many significant projects are not covered here. For surveys of existing annotation models and tools, see Carletta et al. (2002:§3), Bird and Liberman (2001, 1999:§2), or Leech et al. (1998:§3.6.9).

None of these tools specifically supports discourse analysis constituent charts, either in modeling or by rendering into a presentation form. However they are related to this thesis insofar as they involve software tools for linguistically annotated text, usually for discourse annotation. The emphasis for most of them though is on discourse in the sense of spoken dialogue.

The Multilevel Annotation Tools Engineering (MATE) project, sponsored by the European Commission's Telematics/Language Engineering Programme, was completed in 2000 (Dybkjær and Bernsen 2000). The main results of the project are the MATE markup framework and the MATE workbench, the latter consisting of open-source software tools that are freely available for download. The MATE markup framework is a proposed standard for markup of spoken dialogue corpora, and incorporates concepts from more than sixty existing coding schemes from several different encoding levels. This framework is claimed

to be the most comprehensive such framework in existence. The MATE project provides a conceptual model within which users can define custom "coding schemes," including tag sets. Ready-to-use "best practice" coding schemes are also included. The model is a rich one, allowing constituent relationships to be expressed both by nesting of elements and by links (Bird and Liberman 1999b). The MATE Stylesheet Language (MSL), similar to XSL, allows users to design stylesheets for new visualizations (views).

Alembic Workbench (Day et al. 1997) is a set of fairly WYSIWYG tools for semi-automatic annotation of multilingual text, for the purpose of annotating large corpora quickly. The software is available for download.

LinguaLinks (Stutzman 2003) is a set of integrated tools from the Summer Institute of Linguistics (SIL International) for "word analysis," built on the object-oriented infrastructure of the Computing Environment for Linguistic, Literary, and Anthropological Research (CELLAR) (Simons and Thomson 1995). LinguaLinks facilitates interlinear annotation of texts, assisted by morphological parsing and a lexical database. The tools keep data consistent by linking to objects everywhere instead of copying data. Moreover, because the data is stored independently of any particular presentation, multiple views of the data are possible, depending on the task: entering a record, browsing many data records, or exporting a presentation for publication.

For an import/export interchange format, LinguaLinks uses PTEXT ("Parsed Text"), another proposed standard for SGML encoding of (morphologically) analyzed text. PTEXT was designed as a conceptual model for parsed text, and for "interchange of parsed texts

among natural language processing applications" (Simons 1997b). Two successor projects, FieldWorks and WordWorks, expand on the functionality of LinguaLinks.

The *annotation graph* (AG) formalism (Bird and Liberman 1999b) is less a conceptual model for annotation than a simple "common conceptual core" framework for representing more complex annotations. AGs are defined simply as "networks consisting of nodes and arcs, decorated with time marks and labels," and are primarily intended for annotating linear signals such as audio. AG aims to address problems of complexity said to be inherent in the richer annotation formats, such as the difficulty of maintaining coherence of recursive annotation structures in real time over large documents. A formal algebra on AGs is presented, allowing the authors to demonstrate by proof that certain desirable properties of AGs (related to consistency) are preserved over operations such as set union (combining sets of annotations). It is claimed that despite its simplicity, the AG formalism is expressive enough to represent most kinds of annotation currently being done with other models. Bird and Liberman argue convincingly that maintaining coherence of large annotation networks can be done much more efficiently if they are modeled as AGs. What is less clear is whether other operations, such as following links in a hierarchy, becomes significantly more difficult since hierarchical connections are not explicitly represented in the AG formalism. An open-source toolkit has been developed as a basis for annotation and transcription software based on AG, and several tools have already been implemented using the toolkit (Bird et al. 2002).

A related project, based on an extension to AG, is Architecture and Tools for Linguistic Analysis Systems (ATLAS), undertaken by the National Institute for Standards

and Technology (NIST), the Linguistic Data Consortium (LDC), and MITRE Corporation. The ATLAS approach is to interpose a logical layer between applications and encoding schemes, so that applications can be developed independently of the burden of supporting multiple encoding formats. The intermediate layer "defines an abstract annotation model that associates structured data to regions that have been identified in a signal. The data model defines core ontologies that can then be manipulated by applications" (NIST 2003; see also Laprun et al. 2002).

Multilinear Discourse Analysis (MDA) is not an annotation tool, but a tool for producing displays from annotated text (Quick 1996). The input to MDA is a text with Standard Format Marker (SFM) annotations encoding participant reference information. MDA then computes quantitative measures such as topic persistence and referential distance, and outputs tables showing the results.

RSTTool is an open-source tool that facilitates WYSIWYG annotation of texts according to rhetorical structure theory principles (O'Donnell 2000). The annotations are stored in RS2 format and RSTTool can be used to display them in configurable ways. The tool can also compute statistics over annotations on a text, e.g. the prevalence of particular relations.

Many other tools and models for linguistic annotation are available. As has been mentioned, there are over sixty coding schemes for spoken dialogue corpus annotation alone (Dybkjær and Bernsen 2000:1). Dealing with incompatible coding schemes is therefore a major hurdle in the development of specialized software (Bird and Simons 2003:§2.2). Many projects (including MATE, XCES, AG, ATLAS, and PTEXT) aim to provide an interchange

format between other formats or applications. It is clear that any software designed to be able to interoperate with other software on annotated text data must take into account the need to be able to import and export data in a wide variety of formats, without loss of information. Using a sufficiently general interchange format may be a way to solve this problem, though making this solution extensible without losing compatibility will require something like web ontologies.

2.2.6 Information models for displays

A few publications focus explicitly on the information model of a linguistic display. Schmidt (2002) is one: As part of the EXMARaLDA[20] system, he develops a model of interlinear text (IT) display. Schmidt also proposes a rendering process similar to the one proposed in this thesis for constituent charts (Fig. 2). In his process, an instance of an abstract IT document is produced programmatically from annotated text data, and the IT document is then transformed (by a Java program) into a presentation form (Schmidt 2002:Fig. 25).

Another model of IT display is developed in Bow, Hughes, and Bird (2003). In their proposed processing model two transformation steps are involved (see Bow, Hughes, and Bird 2003:Fig. 29). An abstract IT representation containing the text data is transformed via one of several XSL stylesheets (chosen according to the style of display desired) into a surface representation in which the parameters of the display are specified; and the surface representation is transformed (again via one of several stylesheets) into a presentation form in some formatting/layout language.

[20] "EXtensible MARkup Language for Discourse Annotation."

In order to avoid confusion of terminology, it may be worthwhile to note that Bow, Hughes, and Bird's abstract IT representation is roughly, but not strictly, at the same abstraction level as the abstract constituent chart described in this thesis. Their abstract IT representation is somewhat more abstract than the abstract chart (since the former does not specify ordering and selection of slices of data [IT rows]), but is less general than an annotated text (as referred to in §6.1), in that it is limited to information required for IT as opposed to general linguistic annotation.

A third investigation of linguistic displays is found in Weber (2000), where the possible range of content and intricacies of presentation for the "linguistic example" are explored. While Weber does not present a conceptual model as such, he proposes a framework for processing and rendering linguistic examples, and gives many suggestions for useful features of a presentation form, especially for interactive media.

2.2.7 XSL presentation models for linguistic data

The OpenText.org project (O'Donnell, Porter, and Reed 2001) is one of several using XSLT stylesheets to display views of XML encoded data. Its goal is to provide annotated Greek texts, along with open source software tools for analysis and viewing. Annotations include discourse, grammatical, and other features. In one demonstration, XSLT stylesheets give twelve different views of a single XML annotated text: for example, displaying the constituents of each clause, or the distribution of participants. Some of the HTML output of these stylesheets includes interactive features implemented in JavaScript, so that the view can be tailored instantly without re-running the stylesheet processor.

XSL stylesheets provide a standardized way of defining a presentation model that allows the developer and the user to take advantage of widely available tools, thus reducing the complexity and increasing the portability of applications and documents.

2.3 Summary

This chapter has reviewed some of the relevant literature on discourse analysis, on the development of the use of computers for annotation of text, and on software for discourse analysis and linguistic annotation. The next chapter presents requirements for an abstract constituent chart information model.

CHAPTER 3.

REQUIREMENTS FOR INFORMATION MODEL

3.1 Purpose of this chapter

The form and purpose of a constituent chart has already been described. An abstract chart, as mentioned in §1.2, is a document providing the information content and layout of such a chart, independently of any particular output format (such as HTML or PDF). This chapter[21] describes what is required from an abstract chart, in order to develop an abstract chart information model that satisfies those requirements.

3.2 Discussion

3.2.1 Abstract chart information model

An abstract chart plays a mediating role between a particular annotated text and a rendered presentation of that data in a particular format. In Simons's (1997) terms, an abstract chart model is a *conceptual model* of a chart, for which the *visual model* would be a stylesheet that produces a document in HTML, PDF, or some other presentation format.

Each abstract chart could be used to generate multiple presentation forms: for example, one for computer screens (with color and interactive buttons), and another for

[21] The form of this chapter is based on Simons (1993).

black-and-white printouts. The former might be encoded in HTML with JavaScript, the latter in PDF.

3.2.2 Goals guiding requirements

The first goal guiding requirements for the abstract chart model is to be able to encode the information typically contained in a constituent chart, so that a researcher may encode a given chart into an abstract chart and then render it into a presentation form. A second goal is to ensure that the abstract chart (and, to the degree reasonable, the presentation form) will be "portable," that is, will continue to be useful for the long term, and will be accessible to others with different computing environments. Portability requirements cover issues of content, format, discovery, access, citation, preservation, and rights. These issues, along with recommended best practices, are detailed in Bird and Simons (2003).

It has been pointed out earlier that in order for linguistic software tools and the data produced by them to be reusable, they must separate information from presentation. In the case of constituent charts, the chart is in some sense a presentation element, relative to a text which the researcher has marked up (annotated). The annotated text includes information such as a certain range of words being marked as the subject of an independent clause; the corresponding abstract chart would contain the information that this range of words should appear in a particular column of a table. However, relative to the final presentation form, the abstract chart is information. The aforesaid range of words in the abstract chart would be rendered into the presentation form with formatting codes, e.g. in HTML, something like

```
<td>Little Hans</td>
```

in the appropriate context to be displayed in the correct column in a web browser table.

Thus, as the set of information corresponding to a display, an abstract chart is not designed to be the primary source of linguistic analysis (although it could serve as such in the absence of a better arrangement); the primary source of analysis would ideally be an annotated text. Instead, the abstract chart contains information about the structure of the chart display.

3.2.3 Circumscription of purpose

This section clarifies the purpose of the abstract chart model by stating what is not part of that purpose. The goal for the abstract chart model is not to encapsulate every kind of information that has ever been encoded in a constituent chart. Rather it is to cover the commonly used notions. In addition, the abstract chart model is not required to store the information necessary to reproduce every visual aspect of the original charts. Some of the visual nuances in the charts do not fit well into a formal model where information is mapped systematically onto display forms.

Rather than reproducing presentation exactly, the aim is for the abstract chart model to be able to contain the *information* communicated by the original chart, in such a way that the presentation model can render it into a presentation form that satisfactorily communicates the same information.

As a final note, the following list covers basic requirements, which have actually been implemented in this project. Other potentially desirable features are considered in §6.1.

3.3 Requirements

In software development, a requirements document facilitates collaboration by providing a basis for discussion and enabling the consensus-building on what a piece of software should do. This allows those involved to identify and address weaknesses in the understanding of the problem. Requirements make it possible in principle to design software to do the things it has been agreed that it should do, and later to determine whether the software actually does what it should.

These requirements are not intended to specify *how* the information must be modeled; in particular, they do not prescribe the structure or design of a DTD. Rather, the requirements specify *what* information must be modeled in order for other parts of an application (e.g. a rendering process) to be able to do what they are designed to do.

The following requirements are grouped into categories by subject area, but the requirements are numbered continuously without regard to grouping. After its number, each requirement statement begins with a name in bold type, followed by a description of what that requirement entails.

3.3.1 Requirements for analytic configuration

The abstract chart model must contain the information necessary to produce a table (or some equivalent form) in the output with text in columns that are labeled with the names of certain constituents. We are categorizing the column information as “analytic configuration” since it is based on analytical decisions (generally about the language, and perhaps the text genre, under study), such as the most common order of constituents in a clause or sentence.

1. **Columns.** The model must include a list of columns.
    a. **Name.** Each column has a linguistically relevant name, to be displayed in the header row of the output chart.
    b. **Order.** The columns have a defined logical order with respect to each other, within their parent column group (or within the chart as a whole, if they are not within a group). Currently, logical order is assumed to correspond to a left-to-right rendering order (but see requirement 22 in §6.1).
    c. **Width.** It is desirable to be able to specify a preferred width for each column, as a percentage of the total table width. For example, in a given language and genre of text for which independent clauses are far more common than postposed clauses, a well-balanced table should have a wider column for independent clauses and their constituents. If unspecified, the rendering process will decide the width of the columns; but it would be desirable not to have to depend on this because as automatic table renderers (such as Microsoft Internet Explorer rendering an HTML table) currently go, they are unlikely to estimate optimal widths for columns very well when there is a lot of text.
    d. **Abbreviation.** A column may need to have an abbreviation different from its caption, by which it can be referred to in user-visible text in the body of the chart. For example in Fig. 5, sentence 3, “[in O]” is used to mark movement, not “[in O, etc.]”, even though the caption of the column referred to is “O, etc.” In general there may be more space available for header captions than for column references.

2. **Column groups.** There must be a way to group columns together in hierarchical arrangements, e.g. to group S, O, and V columns under the column group "Independent Clause", as in Fig. 3 (above); then the rendering process can produce visual cues that indicate which columns are grouped together (see requirement 12 on column styles).
    a. **Name.** Each column group has a linguistically relevant name.
    b. **Order.** The column groups have a defined logical order with respect to each other within their immediate context (i.e. the chart as a whole, or a parent column group).
3. **Reusability.** It is desirable for the analytical configuration information to be independently reusable by multiple charts. For example a researcher (or group of them) may be charting a series of texts in the same language, and wish to make the charts easily comparable (see diagram in Fig. 6 below).

3.3.2 Requirements for text structure

4. **Sentences.** The charted text consists of a sequence of sentences. A sentence is typically rendered as a group of rows of cells.
    a. **Order of sentences.** Sentences occur in sequential order in the abstract chart as they are to appear in the output. Order is also used to number each sentence in the output. (Note that not every grammatical sentence in the text needs to be charted as a separate sentence. E.g. in *He said, "I am."*, the utterance *"I am."* need not be charted as a separate sentence unless the researcher wishes it to have its own separate row.)

b. **Cells.** Each sentence contains cells.

c. **Rows.** A sentence is rendered as a series of rows of cells. Row information is implied and does not need to be explicitly modeled. Rows are used for intra-sentence numbering (if there are multiple rows within a sentence, they may be numbered *a, b, c*…), and for styling of rows' top and bottom borders (row borders that coincide with sentence borders will typically be rendered with greater weight [thicker or darker] than those within sentences). Every sentence is assumed to start a new row in the output.

5. **Cells.** Sentences in the abstract chart consist of a sequence of cells. Abstract chart cells are typically rendered as table cells in the output.

   a. **Text in cell.** Each cell contains a (possibly empty) sequence of words or other text content.

   b. **Column.** A cell is associated with a column; the table cell will be rendered in that column.

   c. **Order of cells.** Cells have a defined order with respect to each other within their immediate context. This corresponds to the order in which they will appear in the output chart, which is based on the order of the words in the source text.

   d. **Row.** Every cell is rendered in a particular row of the output table. However a cell's row does not need to be modeled explicitly as it can be inferred from other information (in particular, the analytic configuration and the association of cells to columns).

3.3.3 Requirements for text content

This section describes requirements for text content, including annotation on the text. *Annotation* refers to "any descriptive or analytic notations applied to raw language data" (Bird and Liberman 1999b:1). In the case of an abstract chart, "raw language data" means written (electronically encoded) text, and annotation includes marking movement, noting zero forms, and marking spans of text as having certain characteristics.

6. **Words and morphemes.** Text in a chart consists of words, which may be subdivided into morphemes (as in the Inga chart, Fig. 3, sentence 6: *Ajay-si*).
    a. **Order of words and morphemes.** There is a defined order on all words and morphemes with respect to each other within their immediate context. This order is the order in which text will appear in the rendered chart (which is also the order in which the words or morphemes appear in the original text). (Left-to-right and top-to-bottom rendering order is assumed, but see requirement 22 in §6.1.)
    b. **Encoding.** It must be possible to encode any text that can be represented in Unicode. This is the most portable and stable way to represent characters like the accented vowels in the Inga chart (Fig. 3). This would be even more important for more exotic characters such as those in non-Roman scripts.
    c. **Vernacular and gloss.** Each word or morpheme has a vernacular form and may have an associated gloss, for interlinear display (as in the Inga chart sentence 1: *Chihora* has the gloss 'that-time').

d. **Notes.** Chart text may include not only the words of the original text, but also words added by the researcher, such as notes. This text may be specially marked by a span with a feature ID specifically for analytical notes (see requirement 7).

7. **Spans.**[22] It must be possible to associate a range of text in the abstract chart with a particular linguistic feature. This will allow the rendering process to render ranges of text specially according to their associated features.

   a. **Range.** A span covers a range of words and/or morphemes (sub-morpheme granularity is not required), as well as inserted notation (like zero form markers and movement markers—see requirements 8 and 9). For example in the Inga chart, sentence 6, there would need to be a span (associated with a "direct-quotation" feature) covering *Ajay* but not covering the *-si* morpheme which is part of the same word. Another span would cover the word *Caynórami,* a zero anaphora notation, and the word *yachahuanga.*

   b. **Linguistic feature.** A span of text must be associated with an ID corresponding to a linguistic feature. For example, one may need to identify a range of text (consisting of a single word) as a present-tense verb, by associating a span covering that word with a '`Pres`' feature ID, so that the rendering process can then render that word in a style specified in the stylesheet for present-tense verbs. Another example is found at the end of sentence 6 in the Little Hans chart (Fig. 5), where the text starting with "*he* dare go" consists of a complement clause. This

[22] The term *span* is used in this thesis to mean a range of text in the abstract chart model corresponding to an annotation unit in the annotated text in a one-to-one relationship. A span will be implemented as one or more *stretches* in the abstract chart DTD and XML document. A stretch is a range of text in the abstract chart DTD and XML document, subject to the limitations of XML, i.e. not being able to alternate its boundaries with those of sentences, cells, or other elements.

text needs to be associated with a "complement clause" feature ID, so that it can be rendered in <angle brackets>.[23] The abstract chart model does not specify the styles/formatting with which the features are rendered (e.g. underlining, color, or begin-and-end punctuation); that is left to the rendering stylesheet. (But see §6.1, requirement 26.)

c. **Crossing cell and sentence boundaries.** The text in a given span must be allowed to begin in one cell or sentence and end in another, and it must be possible for a span to cover some but not all of the text in a given cell or sentence. In the example referred to above (sentence 6 of Fig. 5), the complement clause begins in the "S" column and ends in the "O, Comp, Others" column.

d. **Original formatting.** It is sometimes necessary to preserve formatting from the original text, without regard to any particular analytic meaning. For example, in the Little Hans chart, the word *he* in *he dare go* is italicized in the original text, and the researcher wished to preserve that information in the chart. This can be modeled with the same mechanism as a linguistic feature.

8. **Zeroes.** It must be possible to encode putative occurrences of "zero" forms in the text, to indicate zero anaphora, verb gapping, etc. (The rendering of such forms is specified as in requirement 11.)

---

[23] Another example of linguistic features that could be treated specially by the rendering process would be location and time, as shown in the Inga chart (Fig. 3). Location change is represented in the original chart by the notation $L_0 \rightarrow L_1$ in sentence 1, while temporal succession is represented by an arrow in the first column linking sentence 2 to sentence 1. However, the author (Levinsohn 2003, personal e-mail) no longer recommends including this information (or notation) in a constituent chart. There are now considered to be better ways to handle this information.

9. **Movement.** There must be provision for indicating fronting or other movement. (The rendering of moved text will be determined by the stylesheet; but see §6.1, requirement 26).
    a. **Range.** The range of text that moved must be specified.
    b. **Source.** The location from which the text moved (i.e. the location where such text would normally be expected) must be specified.
    c. **Destination.** The location to which the text moved (i.e. the location where it actually occurred) must be specified.

3.3.4 Requirements for aesthetic configuration

10. **Row numbering.** A rendering preference indicating whether to number rows (a, b, …) within sentences, throughout the chart.
11. **Zero marker appearance.** A string indicating preferred rendering of zero markers (e.g. for elision or zero anaphora) throughout the chart, typically "Ø" or "——".
12. **Column styling.** Border formatting styles (color, thickness, line style [e.g. double, dotted, solid]) may be specified for columns and column groups, keyed to "rank" (depth of nesting; see requirement 2). For example, one style for top-level columns or groups, a second style for columns or groups nested within those, and so on. These styles apply to vertical borders both in the headers and in the body of the output chart. The formatting must be specified in an output-format-neutral way.
13. **Row styling.** Formatting for horizontal borders in the chart is specified similarly to that for vertical borders (see previous requirement). There should be a style for

sentence borders and another style for row borders (within sentences). Typically sentence borders would be darker or thicker than row borders.

14. **Reusability.** It is desirable for the aesthetic configuration information to be independently reusable by other charts, and to be able to conveniently render one abstract chart with various aesthetic configurations; for example, one type to use with a stylesheet that produces HTML, and another to use with a stylesheet that produces PDF.
15. **Legend.** Different researchers visualize their annotations in different ways, and even a single researcher may not remember a few years later what the various formatting styles were intended to indicate. Therefore it is desirable to document this correspondence in a legend that is attached to the chart. For this the following data is needed:
    a. **Features and other items.** A list of which span feature IDs and other display-configurable items (such as zero forms and moved text) need to be included in the legend.
    b. **Sample text.** For each feature ID, a short string to use when rendering a sample span with that feature ID in the legend.
    c. **Description.** A short, user-friendly text description of each span feature ID's meaning (e.g. "relative clause" or "Participant 3").

3.3.5 Requirements for documentation

The requirements in this section have to do with documenting the chart and its sources. While only the title and attribution affect a typical display of the chart, having all of

this information in the abstract chart promotes accountability and immutability (see Bird and Simons 2003:§6.1,6.5) by providing unambiguous identification of the chart and its sources.

16. **Title**. A text string identifying the chart, to be displayed in title position, e.g. above the chart.
17. **Attribution.** An optional text string that gives credit to contributing sources, to be displayed with the chart, in order to fulfill distribution requirements that call for a copyright or "used with permission" notice or the like to be displayed with the chart.
18. **Version.** The abstract chart's version string, so that this chart can be uniquely identified.
19. **DTD Version.** The version string of the DTD for the abstract chart, so that the chart can be validated properly.
20. **Source.** A reference to the source document, including a version string: for accountability, for access to the metadata in the source document, and in order to be able to update the source data and regenerate the chart if the chart is generated.

3.4 Summary

In this chapter, requirements have been enumerated for an information model to be able to encode the information presented in three example charts. The next chapter presents an information model that meets these requirements.

CHAPTER 4.

ABSTRACT CHART INFORMATION MODEL

The previous chapter described the requirements that an abstract chart information model must fulfill. This chapter presents the main result of this thesis, the information model for an abstract chart designed to fulfill those requirements. Example XML code showing how the information model can be instantiated is given in chapter 5, along with a description of a stylesheet expressing a presentation model, and figures showing the display forms produced by the stylesheet.

4.1 Structure of the abstract chart model

The overall structure of the abstract chart model is shown in Fig. 6:

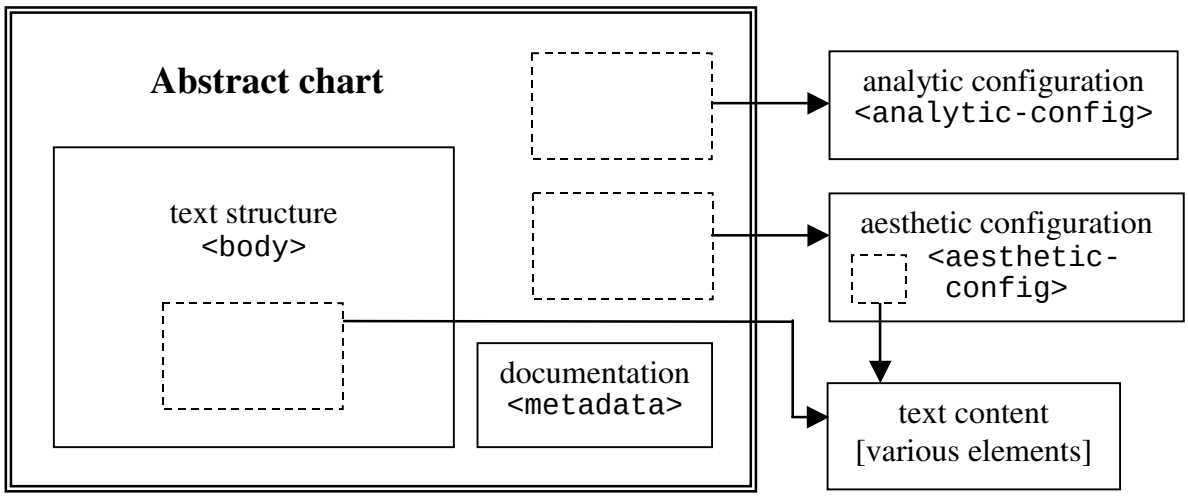

Fig. 6. Information in an abstract chart.

The solid rectangles in the diagram represent modules of the information model. The five modules are: text structure (containing the sentences and cells that make up the structure of the text), text content (which includes text and annotation, intermingled), documentation, analytic configuration, and aesthetic configuration. The string in `<angle-brackets>` in each module in the diagram denotes the name of that module's main XML element. Note that the aesthetic and analytic configuration modules are separated from the main body of the model. (The dashed rectangles and arrows show references to modules.) This separation reflects the fact that the aesthetic and analytical configuration data in an abstract chart XML document can be located in separate files (*external entities,* in XML terminology) from the main body of the document. The purpose of this separation is to make the configuration modules available to be shared by other abstract charts. (See §4.3 for elaboration on this topic, and the sample abstract chart document in §5.2 where this is demonstrated for the aesthetic configuration module.) The text content module is also separated from the main body because its definition is referenced by both the aesthetic configuration module and the text structure module, as will be seen below. However, unlike the configuration modules, it is not intended that text content in XML documents should be separated out into other files.

The DTD for the abstract chart is structured like the above diagram, with the DTD fragments for the configuration and text content components located in separate files. The main file for the DTD is `cc.dtd` (see Fig. 7). This file includes the definition of the text structure module (`<body>` element, shown in Fig. 11) and the documentation module (`<metadata>` element, Fig. 8). The analytic and aesthetic configuration modules are stored in `cc-analyconf.dtd` (Fig. 9) and `cc-aesthconf.dtd` (Fig. 10), respectively. Finally,

the DTD fragment describing text content element types (words, glosses, and so on) has been placed in another separate file, `cc-content.dtd` (Fig. 12), since these element types are used in both the main body of an abstract chart and in the aesthetic configuration module.

4.2 Abstract chart DTD

The following figures present the actual XML code comprising the parts of the abstract chart DTD. Each listing contains comments (between `<!--` and `-->` markers) explaining the function of the various elements and attributes. In comments, the names of attributes are preceded by the `@` symbol, to differentiate them from element names and improve readability.

For those who may not be familiar with DTD's, a typical XML DTD is composed mostly of `ELEMENT` and `ATTLIST` statements. The statement

```
<!ELEMENT eltname (children…)>
```

defines the element type called *`eltname`* and specifies what types of child elements (or in linguistic terms, constituents) it may (or must) have. The statement

```
<!ATTLIST eltname attrname1 … attrname2 …>
```

defines the attributes that elements of type *`eltname`* may/must have. It also declares each attribute's type, its required/optional status, and a default value if any.

Fig. 7, below, gives an overview of the abstract DTD. This portion defines the document element, `cc` (for "constituent chart"), which is the container for all other elements. The `cc` element has four required child elements: `metadata`, `analytic-config`, `aesthetic-config`, and `body`. The structure of each of these child elements is shown in

separate figures below. The DTD fragments defining the `analytic-config` and `aesthetic-config` elements are referenced from the main DTD file at the bottom of Fig. 7. The definitions for the `<metadata>` and `<body>` element types, while contained in the main DTD file, are each displayed in their own separate figure (Fig. 8 and Fig. 11).

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
    DTD for abstract constituent chart
    DTD version 2003-07-28T13:29:00-0500

    Copyright (c) 2003 by Lars Huttar

    Includes cc-aesthconf.dtd and cc-analyconf.dtd.

    Namespace:
      xmlns:cc = "http://purl.oclc.org/net/cicada/cc" for abstract
      constituent chart
-->

<!-- cc:cc - document element for abstract chart -->
<!ELEMENT cc (metadata, analytic-config, aesthetic-config, body)>
<!ATTLIST cc xmlns CDATA #FIXED "http://purl.oclc.org/net/cicada/cc" >
<!-- The above enforces a default namespace for the document. See
      http://www.rpbourret.com/xml/NamespacesFAQ.htm#q7_6 for rationale.
      -->

<!-- The definition of the <metadata> element goes here (see Fig. 8). -->

<!-- include DTD fragment for analytic configuration (see Fig. 9). -->
<!ENTITY % cc-analyconf.dtd SYSTEM "cc-analyconf.dtd">
%cc-analyconf.dtd;

<!-- include DTD fragment for aesthetic configuration (see Fig. 10). -->
<!ENTITY % cc-aesthconf.dtd SYSTEM "cc-aesthconf.dtd">
%cc-aesthconf.dtd;

<!-- The definition of the <body> element goes here (see Fig. 11). -->
```

Fig. 7. DTD for abstract chart: `cc.dtd`.

The following figure (Fig. 8) gives the portion of the DTD that defines the `<metadata>` element, which captures information *about* the abstract chart document—that is, its documentation, or metadata. This element's attributes and child elements store documentation of the abstract chart: its title and attribution (strings to be rendered in the

presentation form); version identifiers of the abstract chart, the DTD, and the source document; and some kind of reference to the source document (such as a Uniform Resource Identifier [URI] or a bibliography entry).

Note that instead of designing our own `<metadata>` structure, another option would have been to use a standardized format, such as that specified by the Open Language Archives Community (OLAC)'s proposed Metadata standard (Simons and Bird 2002), to encode metadata. Such a design would have made it easier to inform others about the existence of an abstract chart, and to share it, via archives[24] that make use of the standard metadata format. However, it was decided that the metadata desired for this project could be encoded most unambiguously in the abstract chart model by using custom markup. This custom markup can later be translated "down" into the OLAC Metadata standard's interchange format without difficulty, when a chart is ready for archiving or distribution.

```
<!-- cc:metadata - documentation for chart -->
<!ELEMENT metadata (title, attribution?)>
<!ATTLIST metadata
  document-version CDATA #REQUIRED
  dtd-version CDATA #REQUIRED
  source-document CDATA #REQUIRED
  source-version CDATA #REQUIRED
>
<!-- The version attributes may contain any string, such as a revision
      date or a version number. -->
<!-- @source-document is the URI of the document from which this one was
      generated, such as an annotated text; or it may be a reference to an
      unmarked-up text or some other data from which this chart was
      derived, perhaps manually. -->
<!-- The metadata element's attributes are not used for processing or
      display, only for documentation of the abstract chart XML document.
      The title and attribution elements are used for display. -->
<!-- While all these attributes are required, @source-document for example
      may be set to 'none' if there is no source document. -->
```

[24] Such as those participating in the Open Archives Initiative (Lagoze et al. 2002).

```
<!-- cc:title, a string, will typically be displayed at top of
      presentation form. -->
<!ELEMENT title (#PCDATA)>

<!-- cc:attribution is a message to be displayed giving credit to some
      party, if such a message is required. -->
<!ELEMENT attribution (#PCDATA)>
```

Fig. 8. DTD section for `<metadata>` element.

Fig. 9, below, shows the DTD fragment for the `<analytic-config>` element (the file `analytic-config.dtd`). This portion of the DTD uses hierarchical containment of `<column-group>` and `<column>` elements to define the grouping of chart columns. Each column-group and column may have a caption; each column may also have an abbreviation and a specified width (as a percentage of table width). Every column also has a key, which allows `<cell>` elements to specify the column to which they should be mapped. The other column and column-group attributes defined below exist to simplify validation and rendering.

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
    DTD fragment for analytic-config element of abstract constituent
      chart.
    DTD version 2003-07-28T13:29:00-0500

    Copyright (c) 2003 by Lars Huttar

    Included by cc.dtd.

    Namespace:
      xmlns:cc = "http://purl.oclc.org/net/cicada/cc" for abstract
      constituent chart
-->

<!-- cc:analytic-config - contains column and column group configuration.
      -->
<!ELEMENT analytic-config ((column-group | column)+)>
<!ATTLIST analytic-config position CDATA #FIXED "1">
<!-- cc:analytic-config/@position=1 is convenient for validating the
      @position of descendant nodes. -->

<!-- cc:column-group - specifies a grouping of columns or column sub-
      groups. -->
```

```
<!ELEMENT column-group (caption?, (column-group | column)+)>
<!ATTLIST column-group
  ncol CDATA #REQUIRED
  rank CDATA #REQUIRED
  position CDATA #REQUIRED
>
<!-- @ncol is the number of columns spanned, which is equal to
      count(.//column). Used for validation. -->
<!-- @rank is depth of nesting, inverted. The @rank of a column-group is
      its child's @rank + 1. -->
<!-- @position is the column index from which this group starts. Used for
      validation. -->

<!-- cc:caption - string(s) for identifying this column or column-group to
      the user. -->
<!ELEMENT caption (long, short?)>
<!ATTLIST caption xml:lang NMTOKEN #REQUIRED>
<!-- @xml:lang is currently unused, but could be used to build an analytic
      configuration that can be viewed in various analysis languages. -->

<!-- cc:long - long caption string for column(-group) -->
<!ELEMENT long (#PCDATA)>
<!-- cc:short - short caption string for column(-group). The current
      rendering stylesheet uses the short string if available; otherwise
      the long string. -->
<!ELEMENT short (#PCDATA)>

<!-- cc:column - element containing caption and other configuration data
      for a column -->
<!ELEMENT column (caption?)>
<!ATTLIST column
  ncol CDATA #FIXED "1"
  rank CDATA #REQUIRED
  position CDATA #REQUIRED
  styleID NMTOKEN #REQUIRED
  key CDATA #REQUIRED
  abbrev CDATA #IMPLIED
  width CDATA #IMPLIED>
<!-- @ncol - A column is always 1 column wide, but this attribute exists
      for uniformity with column-groups. -->
<!-- @rank is depth of nesting, inverted. A column's @rank is its parent's
      rank - 1 (or 1 if no parent). -->
<!-- @position is the index of the column in a table, starting from 1
      (furthest left).  -->
<!-- @styleID, which is the concatenation of 'col' and @position, is
      helpful for linking cells in columns with CSS[25] styles. -->
<!-- @key is a unique id for each column, typically constructed from its
      own caption and its ancestor column-groups' captions. -->
```

[25] The Cascading Style Sheet mechanism (CSS) is "a simple style sheet mechanism that allows authors and readers to attach style (e.g. fonts, colors and spacing) to HTML documents" (Lie and Bos 1996).

```
<!-- @abbrev is a short name by which to refer to this column, probably
      similar to one of the header captions; may be used to render short
      items like text movement source notes (e.g. '[in O]') -->
<!-- @width specifies a desired percentage of the total table width that
      this column should take. -->
```

Fig. 9. DTD fragment for analytic configuration: `cc-analyconf.dtd`.

The following listing, Fig. 10, gives the contents of the file `aesthetic-config.dtd`, the DTD fragment for the `<aesthetic-config>` element. It consists of the optional elements `<settings>` (preferences for rendering), `<column-styles>` (giving information on border styles for columns and column groups, according to rank), `<row-styles>` (specifying styles for row borders within and between sentences), `<stretch-features>` (giving a list of possible `featureID` values for validation of feature-marked stretches elsewhere), and `<legend>`, a list of items to include in the legend table, including sample text and a description. At the end, the fragment references `<cc-content.dtd>`, which defines elements used by both `<legend>` and `<cell>`.

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
    DTD fragment for aesthetic-config element of abstract constituent
      chart.
    DTD version 2003-07-28T13:29:00-0500

    Copyright (c) 2003 by Lars Huttar

    Included by cc.dtd.
    Includes cc-content.dtd.

    Namespace:
      xmlns:cc = "http://purl.oclc.org/net/cicada/cc" for abstract
      constituent chart
-->

<!-- cc:aesthetic-config - contains aesthetic configuration info -->
<!ELEMENT aesthetic-config (settings?, column-styles?, row-styles?,
      stretch-features?, legend?)>
<!ATTLIST aesthetic-config xmlns CDATA #FIXED
      "http://purl.oclc.org/net/cicada/cc" >
```

```
<!-- The above line enforces a default namespace for the element.
    See http://www.rpbourret.com/xml/NamespacesFAQ.htm#q7_6 for rationale.
      -->

<!-- cc:settings - chart-global rendering preferences -->
<!ELEMENT settings (zero-marker | number-rows)*>

<!-- cc:zero-marker - string to use for rendering zero marker -->
<!ELEMENT zero-marker (#PCDATA)>

<!-- cc:number-rows - whether or not to including row numbering in output
      ('true' or 'false'); default 'true' -->
<!ELEMENT number-rows (#PCDATA)>

<!-- cc:column-styles - container for cc:column-style elements -->
<!ELEMENT column-styles (column-style*)>

<!-- possible values for border-style -->
<!ENTITY % border-style-values '(dotted | dashed | double | none |
      solid)'>

<!-- cc:column-style - holds style attributes for rendering of a given
      rank of column (group)s. -->
<!ELEMENT column-style EMPTY>
<!ATTLIST column-style
  rank CDATA #REQUIRED
  border-size CDATA #IMPLIED
  border-color CDATA #IMPLIED
  border-style %border-style-values; #IMPLIED>
<!--@rank tells what group-rank of column to apply this style to.
      Redundant; @rank = position() for a given column-style. Used for
      efficiency. -->
<!-- @border-size, @border-color, and @border-style currently use CSS
      values. -->

<!-- cc:row-styles - container for cc:row-style elements -->
<!ELEMENT row-styles (row-style+)>

<!-- cc:row-style - holds style attributes for rendering of a particular
      style of row border. -->
<!ELEMENT row-style EMPTY>
<!ATTLIST row-style
  styleID (intra-sentence | inter-sentence) #REQUIRED
  border-size CDATA #IMPLIED
  border-color CDATA #IMPLIED
  border-style %border-style-values; #IMPLIED
>
<!-- @styleID tells what row borders to apply this style to. Only two
      @styleID values are meaningful in the stylesheet being used
      (although a stylesheet could conceivably recognize more styles):
      ‘intra-sentence’ and ‘inter-sentence.’ -->
```

```
<!-- cc:stretch-features - a list of feature IDs, so that featureID
      references can be validated. -->
<!ELEMENT stretch-features (stretch-feature*)>

<!-- cc:stretch-feature - a container for a featureID attribute of type
      ID, for IDREFs to refer to. -->
<!ELEMENT stretch-feature EMPTY>
<!ATTLIST stretch-feature featureID ID #REQUIRED>

<!-- cc:legend - container for legend-item elements -->
<!ELEMENT legend (legend-item*)>

<!-- cc:legend-item - specifies a row for the legend, consisting of a
      stretch of text or other material as a sample, and a description.
      Each sample stretch will have a @featureID attribute. -->
<!ELEMENT legend-item (wordStretch | morphemeStretch | noText)>
<!ATTLIST legend-item description CDATA #IMPLIED>

<!-- include DTD fragment for text content (including stretches and
      noText). -->
<!ENTITY % cc-content.dtd SYSTEM "cc-content.dtd">
%cc-content.dtd;
```

Fig. 10. DTD fragment for aesthetic configuration: `cc-aesthconf.dtd`.

The next listing gives the part of the main DTD file that defines the `<body>` element. `<body>` consists of `<sentence>` elements, which consist of `<cell>` elements. Each `<cell>` contains text, which generally consists of words and possibly also stretches, and some non-text elements. Every cell also has a `column-key` attribute which maps it to the appropriate column.

The description of requirement 5.c, regarding rows, notes that the row of a cell (unlike a cell's column) can be inferred from other information present. Specifically, if the cell is the first in a sentence, it is placed in a new row (the row below that of the previous cell). If however the cell is not first in its sentence and it is in the same column as, or is in a column to the left of, the previous non-empty cell, the new cell goes in a new row. Otherwise the new cell goes in the same row as the previous non-empty cell. (The model does not

provide for blank rows.) For this reason there is no need for explicit row information in the abstract chart model.

Note that although the definition of the <cell> element type uses the definitions in cc-content.dtd, there is no entity reference to that file in this section (or in the main DTD file at all). This is because cc-content.dtd is already referenced by cc-aesthconf.dtd, which is referenced by the main DTD file. Element type definitions legally can only be included once in a DTD.

```
<!-- cc:body - the body of the abstract chart, an ordered sequence of
      sentences. -->
<!ELEMENT body (sentence+)>

<!-- cc:sentence - an ordered sequence of cells, which are to be wrapped
      into rows. -->
<!ELEMENT sentence (cell+)>

<!-- cc:cell - a sequence of text etc. that should be displayed in a
      particular cell. -->
<!ELEMENT cell (word | noText | wordStretch)+>
<!ATTLIST cell
  column-key CDATA #REQUIRED
  column-span CDATA "1"
  glossed (yes | no) "no"
>
<!-- @column-key gives reference to a column (corresponding to
      cc:column/@key in cc-analyconf.dtd) in which this cell should be
      rendered. -->
<!-- @column-span is currently unused. -->
<!-- @glossed aids validation: in a given cell, either all words and
      morphemes must be glossed (in which case @glossed = 'yes'), or none
      are glossed (when @glossed = 'no'). This also simplifies interlinear
      rendering. -->

<!-- cc:word, cc:noText, and cc:wordStretch are defined in cc-content.dtd
      which is included by cc-aesthconf.dtd, because cc-aesthconf.dtd
      might conceivably be used apart from this file, whereas this file
      (cc.dtd) will always require cc-aesthconf.dtd. -->
```

Fig. 11. DTD section for <body> element.

In the next listing, Fig. 12, the elements that can be in a <cell> element are defined. <wordStretch> and <morphemeStretch> elements exist to associate feature IDs (corres-

ponding to linguistic features) with series of words or morphemes, or other elements. Words consist of either a vernacular string and optional gloss, or a series of morphemes or other elements. Other elements include zero markers (`<zeroMarker>`, which may stand in for words or morphemes with a null surface form), and text movement sources (`<moveSource>`, marking the spot from which word[s] or morpheme[s] moved). `<moveText>` elements have a `movedTo` attribute that references the `destID` of the stretch to which the text is said to have moved.

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
    DTD fragment for text content of cells (and legend) in abstract
      constituent chart.
    DTD version 2003-07-28T13:29:00-0500

    Copyright (c) 2003 by Lars Huttar

    Included by cc-aesthconf.dtd.

    Namespace:
      xmlns:cc = "http://purl.oclc.org/net/cicada/cc" for abstract
      constituent chart
-->

<!-- attributes for wordStretch and morphemeStretch elements -->
<!ENTITY % stretchAttrs '
  featureID IDREF #REQUIRED
  destID ID #IMPLIED
  unitGroup NMTOKEN #REQUIRED
  frontEnd (yes | no) "yes"
  backEnd (yes | no) "yes"'>
<!-- @featureID specifies the ID of a linguistic feature that will be
      mapped into a formatting style by the stylesheet. -->
<!-- @destID marks the stretch as moved text, and gives the corresponding
      moveSource element a destination ID to point to. When this happens,
      the moved text is contained in the destination stretch, not in the
      moveSource. -->
<!-- @unitGroup gives the key of a text annotation unit, so we can tell
      which stretches belong to the same unit. (These do not necessarily
      have to correspond to unit IDs in the annotated text, though they
      could.) -->
<!-- @frontEnd and @backEnd control whether or not to render span-initial
      / span-final punctuation, such as brackets, since multiple stretches
      may be used to model one span, but the punctuation should not be
      repeated. -->
```

```
<!-- cc:wordStretch - a stretch to associate a style with given words.
      Stretches cannot be empty. -->
<!ELEMENT wordStretch (word | noText | wordStretch)+>
<!ATTLIST wordStretch %stretchAttrs;>

<!-- cc:word - element containing one word of text, either subdivided into
      morphemes or not. -->
<!ELEMENT word ((vernac, gloss?) | (morpheme | morphemeStretch |
      noText)+)>

<!-- cc:vernac - a word or morpheme in the text under study. -->
<!ELEMENT vernac (#PCDATA)>

<!-- cc:gloss - a gloss for the preceding cc:vernac element. -->
<!ELEMENT gloss (#PCDATA)>

<!-- cc:morpheme - element containing one morpheme of text. -->
<!ELEMENT morpheme (vernac, gloss?)>

<!-- cc:morphemeStretch - a stretch to associate a style with given
      morphemes. -->
<!ELEMENT morphemeStretch (morpheme | noText | morphemeStretch)+>
<!ATTLIST morphemeStretch %stretchAttrs;>

<!-- cc:noText - element containing non-text item, such as a zero marker
      or point from which some text moved. -->
<!ELEMENT noText (zeroMarker | moveSource)>

<!-- cc:zeroMarker - element marking zero anaphora or some other zero
      form. -->
<!ELEMENT zeroMarker EMPTY>

<!-- cc:moveSource - element marking point from which some text moved. -->
<!ELEMENT moveSource EMPTY>
<!ATTLIST moveSource
  movedTo IDREF #REQUIRED
  note CDATA #IMPLIED
>
<!-- @movedTo gives the @destID of the stretch indicating where text moved
      from the moveSource location. -->
<!-- @note, if present, is used for a custom string to display instead of
      '[in O]', where O is the column where the destination stretch is
      found. -->
```

Fig. 12. DTD fragment for text content: `cc-content.dtd`.

Regarding the above DTD fragment, footnote 22 on p. 40 mentions that spans are modeled as *stretches* in an XML document and in the DTD. This is because spans must be able to cross boundaries (partially overlap) with other objects, such as sentences, cells,

words, and other spans. But XML elements are not allowed to cross boundaries with other elements. Therefore a span in the abstract chart information model is implemented as a series of stretches in the DTD—as many stretches as necessary to cover the span, nested as needed to accommodate the other data structures. Each stretch for a given span is marked with the same `featureID` attribute, so that the whole span is uniformly rendered. Furthermore, the first and last stretches for each span have the attributes `frontEnd="yes"` and `backEnd="yes"` respectively. This allows appropriate rendering of beginning and ending punctuation for styles that require it, for example, brackets of various kinds for embedded clauses, or quotation marks for direct quotations. Finally, all stretches corresponding to a particular span are given the same `unitGroup` ID attribute, making it possible to identify spans unambiguously. This provision is made to support hypothetical software that may edit abstract charts as primary objects, rather than generating them from annotated text. In such a case the preservation of the unity of spans in the encoded data would enable the software to be more intelligent about spans, making maintenance of charts much more convenient.

A requirement related to spans is 7.d, which states that it may be necessary to preserve formatting from the original text. This requirement is fulfilled without special treatment in the information model; formatted original text is modeled as just another type of styled span. The span styles `italic, bold,` and `underlined` are recognized by the rendering stylesheet and appropriate formatting markup is generated for the output. If additional types of original formatting are needed (not considered a basic requirement for constituent charts), the stylesheet must be augmented to recognize additional styles.

4.3 Sharing of configuration content

In §4.1 it was mentioned that configuration modules can be separated from the main XML file of an abstract chart. The following diagram (Fig. 13) shows how a single aesthetic configuration file might be shared by several abstract charts, for different texts in different

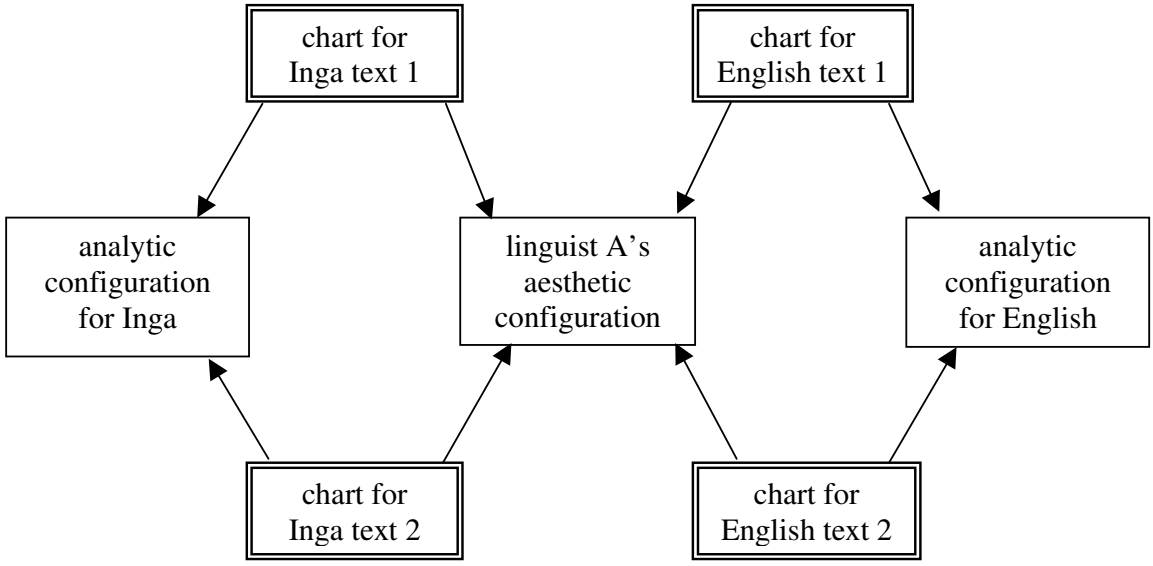

Fig. 13. Sharing configuration files.

languages, giving an overall uniform look. The charts for the two Inga texts share a common analytic configuration file, since their needs for columns and column grouping are similar. Likewise the English texts share one analytic configuration file between them. This would allow a researcher to change the column design (e.g. to adjust the column widths) of a whole group of charts even after much charting has already been done, simply by changing one configuration file rather than modifying each chart individually. Sharing one analytic configuration file could also serve as a guarantee of the comparability of the charting being done on several charts.

Another possible arrangement (Fig. 14) would be to create different aesthetic configurations for different output formats, e.g. HTML and PDF, and swap them depending on the type of output desired. For example, the aesthetic configuration for HTML might use colors for participant references and light gray intra-sentence row borders, while the one for PDF might use black-and-white only, and dashed lines for intra-sentence row borders. Besides swapping aesthetic configurations, it would also of course be necessary to swap stylesheets in order to generate the different output format languages; but the aesthetic configurations could hold settings tailored specifically to those formats.

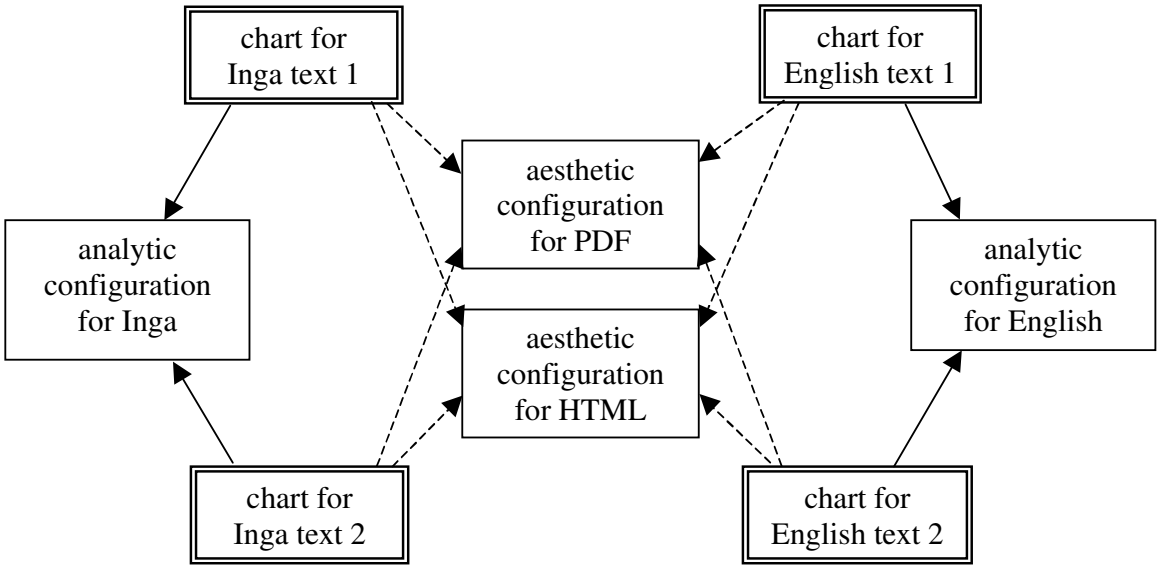

Fig. 14. Swappable configuration files for different purposes.

With the current abstract chart model, it is possible to swap configurations by changing external entity references in each abstract chart to reference different configuration components, and by running different stylesheets. Ideally, an application would take from the user the burden of managing entity references and modifying the XML files. For convenience, the application could enable the user to create a project profile listing several

common scenarios. Each scenario would specify the abstract charts to render, configuration components, stylesheets to use, and the associations between them. Then the user could select the desired scenario according to the need of the occasion and run it, without having to modify the abstract charts.

4.4 Summary

This chapter has presented an information model for abstract charts, expressed as a DTD. In the next chapter, it is demonstrated how the information model was instantiated for the source charts shown in Fig. 3-Fig. 5. Then a presentation model (implemented as a stylesheet) is briefly described, and the resulting presentation forms of the charts are shown.

CHAPTER 5.

PROOF OF CONCEPT

5.1 Introduction

This chapter demonstrates the adequacy of the abstract chart information model. The information in the three original charts was entered into instances of the abstract chart model, then HTML presentation forms of the charts were generated using an XSL stylesheet. These generated HTML forms were judged to be adequate by the authors of the original charts (personal communication, 2003).

In the following sections, sample XML code is presented to illustrate the use of the abstract chart model. Then a stylesheet used to render the HTML chart displays is described in general terms. Finally, the rendered chart displays themselves are shown.

5.2 Sample abstract chart XML

This section shows excerpts from an XML encoding of an abstract chart, demonstrating how the abstract chart information model is instantiated. The abstract chart is based on the original Inga chart shown in Fig. 3. As before, XML comments in <!-- -->, as well as comments in prose, elucidate some aspects of the XML. This abstract chart forms the basis from which the presentation form in Fig. 20 is rendered.

Fig. 15, the first figure, is an overview of the XML document. The DOCTYPE statement indicates that the document is intended to conform to the DTD cc.dtd (shown in Fig. 7), and can be validated against it. It also shows the entity declaration for referencing the aesthetic configuration, which is located in a separate file, aesthconf1.xml. After the DOCTYPE statement comes the document element, <cc>, which contains everything else in the document. Comments show where the other parts of the document (documentation, analytic and aesthetic configurations, and chart body) plug in.

```
<?xml version="1.0" encoding="utf-8"?>
<?xml-stylesheet type="text/xsl" href="cc-to-html.xsl"?>
<!DOCTYPE cc SYSTEM "cc.dtd" [
  <!-- Entity declaration for aesthetic configuration component. -->
  <!ENTITY aesth-conf SYSTEM "aesthconf1.xml">
]>

<!-- Document element. -->
<cc xmlns="http://purl.oclc.org/net/cicada/cc">

  <!-- Documentation (<metadata>) goes here (see Fig. 16). -->

  <!-- Analytic configuration goes here (see Fig. 17). -->

  <!-- Entity reference for aesthetic configuration (see Fig. 18). -->
  &aesth-conf;

  <!-- Body of chart goes here (see Fig. 19). -->

</cc>
```

Fig. 15. Overview of Inga abstract chart XML document.

Fig. 16 shows the instantiation of the documentation of the chart, encoded in the <metadata> element. The DTD section for <metadata> is shown in Fig. 8. The metadata attributes contain information that is not intended for display (although it could be displayed): the version of this document, the version of its DTD, the source from which it is derived, and some form of version string identifying the version of the source. The

elements within <metadata>, on the other hand, hold information that is to be rendered as part of the presentation form: the title of the chart, and an attribution string.

While the <attribution> element and the source-document attribute contain some similar information, their purposes are different. The source-document attribute is designed to help ensure that the abstract chart contains the information necessary to recover the source of the data from which the abstract chart was derived. The <attribution> element, on the other hand, provides a place for a message (if any) that should always be displayed with the chart, including obligatory phrases such as "Used by permission" that do not help identify the source. Moreover, the attribution may refer to a different document than the source document. For example, the attribution may reference a visual presentation of the chart in a published work, while the source document is an annotated text XML document from which the abstract chart was generated.

```
<!-- Documentation -->
<metadata
    source-document="Longacre, Robert E. and Stephen H. Levinsohn. 1978.
      Field analysis of discourse. In Current Trends in Textlinguistics,
      ed. by Wolfgang U. Dressler, 103-22. Berlin: de Gruyter."
    source-version="1978"
    dtd-version="2003-07-28T13:29:00-0500"
    document-version="2003-07-28T13:29:00-0500">

  <title>Constituent chart: "Inga Text"</title>

  <attribution>from Longacre, Robert E. and Stephen H. Levinsohn. 1978.
      Field analysis of discourse. In Current Trends in Textlinguistics,
      ed. by Wolfgang U. Dressler, 103-22. Berlin: de Gruyter. Used by
      permission.  [With modifications based on Dooley and Levinsohn
      (2001) and personal communication with Levinsohn,
      2003]</attribution>

</metadata>
```

Fig. 16. Documentation module of Inga abstract chart.

Next, Fig. 17 shows the analytic configuration portion of the Inga abstract chart XML document. (See Fig. 9 for the DTD fragment for an analytic configuration.) For this particular abstract chart, the analytic configuration is not located in a separate file (entity), but is part of the main XML file.

The purpose of an analytic configuration is to define the columns of the chart and how they are grouped. This particular document defines six columns, three of which belong to a column group. The first column (which has no caption) is used for displaying the sentence number; then follows the column for pre-nuclear elements, followed by the "Independent Clauses" column group, which contains three columns: "S", "(O)", and "P"; and last is the "Post-nuclear elements" column. The top-level columns and column group (that is, those that are not part of a column group) are of rank 2, while the second-level columns (those that are part of a first-level column group, namely, "S", "(O)", and "P") are of rank 1. The rank is used by the stylesheet to render the top-level columns and column group with thicker borders (see `<column-styles>` in Fig. 18).

```
<analytic-config>

  <!-- Sentence number column (no caption). -->
  <column position="1" rank="2" styleID="col1" key="Snum"/>

  <!-- Pre-nuclear elements column. -->
  <column position="2" rank="2" key="pre" abbrev="Pre-nuc"
          styleID="col2">
    <caption xml:lang="en"><long>Pre-nuclear elements</long></caption>
  </column>

  <!-- Group the S, (O), and P columns under the Independent
    Clauses column-group: -->
  <column-group position="3" rank="2" ncol="3">
    <caption xml:lang="en"><long>Independent Clauses</long></caption>

    <column ncol="1" key="ind-cl/s" position="3" styleID="col3"
          abbrev="S" rank="1">
      <caption xml:lang="en"><long>S</long></caption>
```

```
      </column>

      <column ncol="1" key="ind-cl/o" position="4" styleID="col4"
            abbrev="O" rank="1">
        <caption xml:lang="en"><long>(O)</long></caption>
      </column>

      <column ncol="1" key="ind-cl/v" position="5" styleID="col5"
            abbrev="P" rank="1">
        <caption xml:lang="en"><long>P</long></caption>
      </column>

    </column-group>

    <column ncol="1" position="6" key="post" abbrev="Post-nuc" rank="2"
            styleID="col6">
      <caption xml:lang="en"><long>Post-nuclear elements</long></caption>
    </column>

  </analytic-config>
```

Fig. 17. Analytic configuration module of Inga abstract chart.

The contents of the aesthetic configuration for the Inga chart are shown in Fig. 18. For the aesthetic configuration DTD fragment, see Fig. 10. Unlike the analytic configuration, in this abstract chart the aesthetic configuration is located in a separate file, `aesthconf1.xml`.

The aesthetic configuration consists of five elements: `<settings>`, which are rendering preferences that apply to the whole chart; `<column-styles>`, which determine what styles of borders are used to render columns and column-groups of various ranks; `<row-styles>`, which determine what styles of horizontal borders are rendered between sentences and within sentences; `<stretch-features>`, which enumerate the list of valid feature IDs that stretches in the body of the chart may use; and `<legend>`, which lists the items to be documented in the legend table that will be rendered with the chart.

Note that the actual formatting for a given stretch-feature (e.g. bold, blue color, etc.) is nowhere specified in the aesthetic configuration nor in the rest of the abstract chart; that is determined solely by the style sheet used for rendering (but see §6.2, paragraph 26).

Each legend item has a description attribute, and contains sample text to display. This sample text is encoded using the same markup as the text content of a cell in the chart body. The DTD for text content is shown in Fig. 12.

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- <!DOCTYPE aesthetic-config SYSTEM "cc-aesthconf.dtd"> -->
<!-- Since this file is an entity included in another XML file, it
      cannot have its own DOCTYPE declaration. -->

<aesthetic-config>

  <!-- Chart-global rendering preferences. -->
  <settings>
    <!-- Yes, number the rows. -->
    <number-rows>true</number-rows>
    <!-- Represent zero marker as two em dashes. -->
    <zero-marker>&#x2014;&#x2014;</zero-marker>
  </settings>

  <!-- Style info for column(-group)s, by rank. -->
  <column-styles>
    <!-- No special styling for rank 1. -->
    <column-style rank="1"/>
    <!-- Column-groups of rank 2 are rendered with double-line
      borders. -->
    <column-style border-style="double" rank="2"/>
  </column-styles>

  <row-styles>
    <!-- Row borders betweem sentences are solid. -->
    <row-style styleID="inter-sentence" border-size="1" border-
      style="solid" border-color="black"/>
    <!-- Row borders within sentences are dashed. -->
    <row-style styleID="intra-sentence" border-size="1" border-
      style="dashed" border-color="black"/>
  </row-styles>

  <!-- Stretch feature IDs allowed in this chart (for validation)
      -->
  <stretch-features>
    <stretch-feature featureID="move-dest" />
```

```
    <stretch-feature featureID="direct-quote" />
    <stretch-feature featureID="partcpt-1" />
    … <!-- additional stretch-features omitted -->
  </stretch-features>

  <!-- Items to be included in legend table. -->
  <legend>

    <legend-item description="Moved (e.g. fronted) text">
      <wordStretch unitGroup="u1" featureID="move-dest" destID="m-sample">
        <word><vernac>sample</vernac></word>
      </wordStretch>
    </legend-item>

    <legend-item description="Direct quotation">
      <wordStretch unitGroup="u2" featureID="direct-quote">
        <word><vernac>sample</vernac></word>
      </wordStretch>
    </legend-item>

    <legend-item description="Participant 1: mother-in-law">
      <wordStretch unitGroup="u3" featureID="partcpt-1">
        <word><vernac>...</vernac></word>
      </wordStretch>
    </legend-item>

    … <!-- additional participant legend items omitted -->

    <legend-item description="Zero anaphora ">
      <noText><zeroMarker/></noText>
    </legend-item>

  </legend>

</aesthetic-config>
```

Fig. 18. Aesthetic configuration module of Inga abstract chart.

Finally, in Fig. 19, we fill in the main body of the chart, which like most of the chart is located in the main XML file, `inga-cc.xml`. The DTD section for the `<body>` element is shown in Fig. 11.

Here only the first sentence in the chart is shown, as an illustration of the markup system. The sentence contains a series of six cells. The first five cells occur in successive columns, and therefore are rendered in a single row. The sixth cell occurs in the same column as the fifth, and is therefore rendered on the following row.

The first cell is associated with the "Pre-nuclear elements" column by means of that column's key, "`pre`". The cell's contents consist of one word, with a vernacular component (*Chihora*) and a gloss ('that-time'). The second cell contains an example of a stretch, associating the words *chi suégraca* with the feature ID "`particpt-1`", identifying the phrase as a reference to participant number 1 in a list, which is the mother-in-law in this story. This stretch's `unitGroup` ID, "`u7`", is unique to this stretch, indicating that it corresponds to a complete span, and does not need to be united with other stretches in order to arrive at the linguistic unit to which this feature ID was originally applied. (See discussion below Fig. 12 for motivation of this attribute.)

The third cell contains two non-textual items, a zero marker (which is marked as a reference to "`particpt-2`" by means of a surrounding stretch), and a movement source, which indicates that some text has moved from this location to a location identified by the destination ID "`m1`". The movement source has a value for the optional `note` attribute supplying the string "Postposed Goal" to be used instead of the default "in Post-nuc" in a note in square brackets. (The default note and square brackets are determined by the rendering stylesheet.) The destination ID "`m1`" is found in the last cell in this sentence, where it is identified by a surrounding stretch with a `destID` attribute value of "`m1`". That stretch contains another stretch, which uses the feature ID "`partcpt-A`" to associate its contents, *chilacuán piti,* with participant (or prop) A, a piece of papaya. This example demonstrates that it is possible to associate multiple features with a given linguistic unit by using nested stretches.

```
<body>

  <!-- first sentence -->
  <sentence>

    <!-- first cell goes in Pre-nuclear column -->
    <cell column-key="pre" glossed="yes">
      <word>
        <vernac>Chihora</vernac>
        <gloss>that-time</gloss>
      </word>
    </cell>

    <cell column-key="ind-cl/s" glossed="yes">
       <!-- Stretch associates this phrase with featureID for
          participant 1. -->
      <wordStretch unitGroup="u7" featureID="partcpt-1">
        <word>
          <vernac>chi</vernac>
          <gloss>that</gloss>
        </word>
        <word>
          <vernac>suégraca</vernac>
          <gloss>mother-in-law</gloss>
        </word>
      </wordStretch>
    </cell>

    <cell column-key="ind-cl/o" glossed="no">
      <wordStretch unitGroup="u8" featureID="partcpt-2">
        <!-- A use of a zero marker, referring to participant 2 -->
        <noText><zeroMarker/></noText>
      </wordStretch>
      <noText>
         <!-- Text moved from here. -->
        <moveSource movedTo="m1" note="Postposed Goal"/>
      </noText>
    </cell>

    <cell column-key="ind-cl/v" glossed="yes">
      <word>
        <vernac>ñugpagrinsi,</vernac>
        <gloss>went-ahead-of</gloss>
      </word>
    </cell>

    <cell column-key="post" glossed="yes">
      <word>
        <vernac>huacaspa,</vernac>
        <gloss>weeping</gloss>
      </word>
    </cell>
```

```
      <!-- Another cell in the same column (goes to following row)
     -->
     <cell column-key="post" glossed="yes">
        <!-- Nested stretches: outer one indicates text that moved
           here. -->
       <wordStretch unitGroup="u9" featureID="move-dest" destID="m1">
          <!-- Inner stretch associates phrase with participant A (a
           prop). -->
         <wordStretch unitGroup="u10" featureID="partcpt-A">
           <word>
             <vernac>chilacuán</vernac>
             <gloss>wild-papaya</gloss>
           </word>
           <word>
             <vernac>piti</vernac>
             <gloss>piece</gloss>
           </word>
         </wordStretch>
         <word>
           <vernac>pambascama.</vernac>
           <gloss>to-where-had-buried</gloss>
         </word>
       </wordStretch>
     </cell>

   </sentence>  <!-- End tag of first sentence. -->

 …  <!-- Remaining sentences omitted. -->

 </body>
```

Fig. 19. Body of Inga abstract chart.

5.3 A rendering stylesheet

In this section, a rendering stylesheet for an abstract chart, `cc-to-html.xsl`, is described in general terms. The full code of the stylesheet can be found at (Huttar 2003). Other stylesheets could be used instead to render abstract charts into different presentation forms.

The stylesheet begins with the use of an auxiliary stylesheet, `validate-cc.xsl`, which performs validation of the abstract chart for constraints that cannot be specified in the DTD. For example, the DTD can specify that every stretch's `featureID` attribute must be

one of a list of valid `featureID`s, but it cannot express the constraint that every `column`'s `rank` attribute must hold a value one less than that of its parent. If the validation stylesheet finds that the abstract chart violates a constraint, a diagnostic message is printed describing where the error was found, and stylesheet processing terminates.

The rendering stylesheet proceeds by emitting an HTML header, including CSS styles code. Much of this CSS code is boilerplate, and could be moved out to a separate `cc.css` stylesheet, referenced by link from the HTML header, if desired. CSS styles for row and column borders are generated from the aesthetic configuration information in the abstract chart.

After generating the legend table and the title and attribution, the stylesheet generates the table which is a view of the chart itself. First, header rows are generated based on the columns and column-groups in the abstract chart's analytic configuration; then table body rows are generated based on sentences and cells in the annotated text portion of the abstract chart. HTML `span` elements with CSS `style` attributes are used to associate `wordStretch` and `morphemeStretch` styles with the text spanned by the stretches. CSS `style` attributes are also used to associate cells with the appropriate row-border and column-border styles.

5.4 Chart for Inga story

The HTML chart display for the Inga text, as rendered from its abstract chart, is shown in Fig. 20. (For comparison, the originally published version may be seen in Fig. 3.) As noted earlier in footnote 23 above, some information have not been preserved: the location tracking and temporal succession information indicated by the L_1/L_2 and arrow

Constituent chart: "Inga Text"

from Longacre, Robert E. and Stephen H. Levinsohn. 1978. Field analysis of discourse. In Current Trends in Textlinguistics, ed. by Wolfgang U. Dressler, 103-22. Berlin: de Gruyter. Used by permission. [With modifications based on Dooley and Levinsohn (2001) and personal communication with Levinsohn, 2003]

Sample	Meaning
sample	Moved (e.g. fronted) text
«sample»	Direct quotation
... 1	Participant 1: mother-in-law
... 2	Participant 2: father
... A	Participant A: piece of papaya
——	Zero anaphora

	Pre-nuclear elements	Independent Clauses			Post-nuclear elements
		S	(O)	P	
1a	*Chihora* that-time	*chi suégraca 1* that mother-in-law	—— 2 [Postposed Goal]	*ñugpagrinsi,* went-ahead-of	*huacaspa,* weeping
b					***chilacuán piti A pambascama.*** **wild-papaya piece to-where-had-buried**
2a	*Chayagríspaca,* going-and-arriving	—— 1	—— 2	*ninsi:* said	
b		« —— 3	*Caypimi* here	*pambarayá »* is-buried	
3a	*Chasa níspaca,* thus saying	—— 1	*carumalla* to-just-far	*sipirigrís* going-and-strangling-self	
b				*miticú.* fled	
4	*Chihora* that-time	*chi taytaca 2* that father	*chi pozótasi* that grave	*utcú.* dug	
5	*Alpa sitaspa,* earth removing	—— 2	*chilacuán pitíllasi A* wild-papaya just-piece	*tarí.* found	
6a		—— 2	*« Ajay » -si* oh-no	*ní.* said	
b	*« Cayhórami* now		—— 2	*yachahuanga. »* it-will-be-known-to-me	
7	*Chasa níspaca,* thus saying	—— 2	*rastrótasi* footprint	*catichí* followed	
8	***Riscata 1*** *catichí-spaca,* **to-one-who-had-gone** following	—— 2	[in Pre-nuc]	*tarigrinsi.* went-and-found	
9	*Timpo* already	*sipiríscasi 1* one-who-had-strangled-self		*huarcurayá* was-suspended	

Fig. 20. Display for Inga story rendered from abstract chart.

notations. Moreover, the dashed lines and circles marking participant reference chains have been removed, at the original chart author's suggestion.[26] (The latter does not constitute the omission of information, but merely of notation, since participant reference chains can still be inferred from the participant reference information that is shown.)

[26] Dooley and Levinsohn (2001:64) describe a separate method of participant reference charting.

It may also be noted that some aesthetic details have been changed: the vernacular text is rendered in italics rather than underlined, and quotations are enclosed in «guillemots» rather than dash-underlined. Moreover the gloss line text is smaller than the vernacular line. These are all configurable aesthetic changes, and could have been done in the same form as the original chart (except for the dashed underline, which is not supported in HTML). The essential point, however, is that the *information* in the original chart is preserved and is presented clearly and consistently.

A few other features[27] have been added in the HTML version of the chart. Bold formatting is used to make moved text more noticeable (as in the last column of line 1b). A legend is added at the upper right of the chart to ensure that a reader of the chart will have the information necessary to decode the chart. Another feature added, although not visible in the print version of this thesis, is that a different color was used for each participant to make participant reference tracking easier. Color features would be useful for on-screen viewing of charts (e.g., on the web). Another new feature is dashed intra-sentence lines to separate rows from each other. For example, in the original Inga chart (Fig. 3), if sentence 2 is compared with sentence 5, it is not immediately obvious that in sentence 2, *ninsi* precedes *Caypimi* because *Caypimi* is on a separate logical row; whereas in sentence 5, *tarí* comes after *pitíllasi* because the two are on the same logical row (but *pitíllasi* appears on the next physical line [in the same cell] because of word wrapping). This ambiguity would not occur in the HTML form of the chart; the use of dashed lines between rows in the HTML form of the chart

[27] "Features" in the sense of useful characteristics, not in the sense of linguistic features.

clarifies the difference between word wrapping (as in sentence 5) and logically separate rows (as in sentence 2).

This brings us to a limitation in the way this particular stylesheet uses HTML to render interlinear text. (This is not a limitation of the abstract chart model but of the presentation method.) The method used here guarantees that each vernacular word or morpheme and its gloss will be aligned properly, and will not wrap lines in such a way as to interpose other text between a vernacular word or morpheme and its gloss. However, this is accomplished at the cost of rigidity. In the HTML rendered chart, a word and its gloss cannot be wrapped together as a unit (for example, "*miticú* / fled" in sentence 3 cannot be rendered below "*sipirigrís* / going-and-strangling-self"[28]). Bow, Hughes, and Bird (2003:§5.3) note that this limitation is inherent in simpler formatting languages such as HTML and DocBook, but can be overcome using more sophisticated layout languages such as XSL Formatting Objects (XSL-FO) and TEX.

5.5 Chart for "Ordeal"

The next chart, "Ordeal" (Fig. 21), is a fairly straightforward application of the abstract chart model to the original chart. (The original is shown in Fig. 4.) It may be noted that the process of applying descriptive markup to the chart uncovered some inconsistencies in the original, with the result that the quality of the chart was improved. For example, the quotation "Hello" (sentence 3) is given different punctuation in the original than the quotation "How's it going?" (sentence 9). This was not due to any analyzed distinction

[28] Depending on the HTML browser, it may or may not be possible for a long, hyphenated gloss, such as "to-one-who-had-gone" (sentence 8), to be wrapped in the space below the vernacular.

Constituent chart: "Ordeal in the Winter Woods"

Sample	Meaning
text	moved (e.g. fronted) text
«text»	direct quotation
<text>	complement clause (of "merged sentence")
----------	verb gapping

	Preposed	S	V	O	Postposed
1	Early in the afternoon of the tenth day, a Friday,	Robin	suddenly heard	the swish of skis.	
2		He	saw	a cross-country skier	on a course that would bring him about 100 feet away.
3		Robin	shrieked,	«Hello!»	
4		The skier	was	22-year-old, John Steinmetz	a state-parks life-guard in Sta. Cruz.
5a	Also a lone skier,	he	was trying		
b			<to come to grips with	a problem of his own.>	
6a	A few weeks earlier,	he and others	had failed		
b			<to resuscitate	a drowned swimmer.>	
7		He	was ridden	with remorse.	
8a		Some quality in Robin's shout	made		
b		<John	swerve to a halt and pole his way back.>		
9a	«How **'s**	it	[in Pre] going?»		
b		he	asked	the injured skier.	
10		«I	have	a broken leg and ankle.»	
11	«How long **have**	you	[in Pre] been out here?»		
12			«----------	Ten days.»	
13a		«It	took	me	about 45 minutes
b			<to get here		from McCabe Lake,>»
c		John	said.		
14a	«Well,»	Robin	answered,		
b		«It	took	me	six days.»

Fig. 21: Display for "Ordeal" rendered from abstract chart.

(personal communication with author, 2003). Having to give each quotation or other unit a descriptive tag (in this case, a `featureID` attribute on a span) caused us to focus on the actual information encoded in the chart, which in the end resulted in a more consistent presentation form.

With any chart, however intuitively designed and consistently executed, some aspects of the visual "markup" may be unfamiliar to a given reader. To solve this problem, the author must spend time writing, and the reader must spend time finding and digesting, an explanation of what the various visual forms mean. In the case of the original "Ordeal" chart, for example, the meaning of the dashed lines and the arrows pointing across columns[29] was not at first clear to this reader. This situation can be ameliorated by use of a notation summary (legend) located directly adjacent to the chart itself. The legend shown here in the HTML version is automatically generated from the information supplied in the aesthetic configuration. Clearly not all types of notation fit well into a table cell, but this is a convenient format for many cases.

Upon investigation, it turned out that one linguistic phenomenon, text movement, was being rendered in three different ways in the three sample charts. In the Inga chart (Fig. 3), the word "fronted" in the (O) column (sentence 8) indicates that some text has moved leftward from the (O) column. The notation "(O)" under *Riscata* indicates that *Riscata* was the text that had moved from the (O) column. (Similarly with "(O)" in sentence 1, although "postposed," corresponding to "fronted," is not used in the (O) column in the original chart.) In the "Ordeal" chart (Fig. 4), text movement is shown by arrows: the arrow points from the

[29] The dashed lines indicate that the following text is the complement clause of a "merged sentence." The arrows are discussed below.

moved text to the column from which it has moved (sentence 9 for example). The moved text is placed in parentheses. And in the "Little Hans" chart (Fig. 5), for example in sentence 3, a note "[in O]", placed in the column from which the text moved, tells what column the text moved to. In "Little Hans," no special marking is used to highlight the moved text itself, or to distinguish it from any other text that may be in the same column.

For this sample implementation, the decision was made to render all three charts using one stylesheet, thus unifying their notation to a large degree (with the exception of the zero marker notation, which is configurable in the aesthetic configuration component of each abstract chart). A conflation and adjustment of the three notations was settled on, using "[in A]" in column B to indicate that text had moved from B to A, and rendering the moved text in bold for greater visibility.

In another instance of unified visualization, angle brackets < > are used to mark complement clauses in the "Ordeal" HTML chart as they are in the "Hans" chart, though the description of them shown in the legend is different: in the "Ordeal" chart they indicate a special kind of complement clause, that of a "merged sentence." The chart's author felt that this was an important distinction that needed to be preserved.

5.6 Chart for "Little Hans"

The "Little Hans" chart is the longest and most complex of the three. The first page of the rendered chart is shown in Fig. 22. See Fig. 5 for the original version. (The full HTML rendered version of this chart and the others, `*-cc.html`, can be found at [Huttar 2003].) Encoding the Hans chart with descriptive markup required some analysis of what information was actually intended in the way the original chart was formatted. Double vertical lines

Constituent chart: "Little Hans"

from Longacre, Robert E. and Shin Ja J.Hwang. n.d. Discourse Analysis/Textlinguistics: a Field Manual. Used by permission.

Sample	Meaning
text	moved (e.g. fronted) text
[in O]	where moved text "came from"
Note	analytic note
<text>	complement clause
{text}	relative clause
«text»	direct quotation

S#	Notes	Introducers		Preposed Dependent Clauses				Independent Clauses			Postposed Dependent Clauses			
		S-I	S-M	conj.	S	P	O, etc.	S	P	O, Comp, Others	conj.	S	P	O, etc.
1	Stage							The winter afternoon	was	dark and grey over Old Strasbourg.				
2a								Little flurries of snow	came whirling down	between the chimneys				
b			and					a biting wind	blew	in the narrow streets.				
3a	VS order	Above the roofs,			Ø	rising	high into the clouds,	[in O]	stood	**the great cathedral,**		its stones	Ø	dim in the gathering gloom,
b												its windows	catching	the lights within.
4								Fine people	were hurrying	up the broad steps —ladies with furs, gentlemen in splendid attire,		many of them	coming	in their carriages.
5								Little Hans	watched	them.				
6a					Ø	Ø	Perished with cold,							
b					Ø	Ø	ragged,							
c	Durative				Ø	Ø	an unwanted bit of humanity,	he	snuggled	between two buttresses— a retreat from the wind—				
d			and					Ø	wished					
e								< *he*	dare go	into the cathedral {where all was warm and bright, and where (as he could dimly hear) the organ was pealing loudly.}>				

Fig. 22. Display for "Little Hans" rendered from abstract chart (first page).

showed the boundaries of column groups, as in the Inga chart. This convention was carried over to the HTML chart rendering. But what was meant by the dashed line between the "Conj" and "S" columns under the "Preposed Dependent Clauses" group, and how did its meaning differ from that of the lack of vertical lines between the "S", "V", and "O, etc." columns in that group?[30]

It was determined in consultation with the chart author that the "Conj" column is simply a member of the "Preposed Dependent Clauses" column group, on equal standing with "S", "P" (or "V"), and "O, etc." These columns were then modeled as equals in the abstract chart, and for this reason the rendered HTML output uses the same style of vertical lines (single and solid) between these four columns, but surrounds the Preposed Dependent Clauses group with double lines. The decision was also made to separate the "#" and "Notes" columns into two columns.

Another aspect of the word-processor-created Hans chart is the use of text overflow across columns. For example, in the "Postposed Dependent Clauses" column group, the "S", "P", and "O, etc." columns are modeled as a single table column, in which constituents are aligned with the proper header by the insertion of spaces before the text. This allows longer constituents to wrap across two or all three columns, where desired, so that the material takes up fewer rows and/or does not require columns as wide as it would otherwise. This approach works in practice for a static chart, but if it becomes necessary to change the font size, adjust column widths, or edit the text, the line breaking and spacing has to be redone by hand in

[30] Some differences between the original Hans chart and the rendered HTML version are due to differences of preference between the original chart author (who also consulted in the modeling and rendering of the HTML version) and the colleague who first created the word processor table version of the chart. In some of the formatting and layout decisions, the need to conserve space played a role. The version shown in Fig. 5 was modified from the first word processor version to conform more closely to the original chart author's intent.

order to restore reasonable line wrapping and alignment of text with headers. The abstract chart for Little Hans, by contrast, models the “S”, “P”, and “O, etc.” columns as separate columns, and there is no provision for text to overflow across the columns. The result is that while the chart may take up more space in some configurations, the benefit of flexible rendering, and the freedom to make changes in the chart without time-consuming manual cleanup afterwards, outweigh the cost in space. Because of this flexibility, experimentation with different chart dimensions is greatly facilitated, and the chances of finding a configuration that suits the material and the space available are increased.

It may also be noted that horizontal lines are not used in the original “Hans” chart to separate rows within sentences (although one early version had them in some places). The abstract chart rendering stylesheet produces horizontal lines between all adjacent rows, eliminating the occasional ambiguity between local word wrapping and row succession as discussed in §5.4.

In sentence 1, in the “O, Comp, Others” column, the original chart has an intentional line break between “grey” and “over,” rather than allowing the text to wrap wherever it ran out of space to go on. The motivation for this is that where possible, it is visually preferable for each linguistic unit (in this case, the phrase “over old Strasbourg”) to be rendered on one line rather than being split over a line break. This kind of advanced layout control could be achieved with a rendering stylesheet that produces XSL-FO output. However, for the scope of this thesis, in which the stylesheet produces the simpler HTML output, it was decided that only ordinary word wrapping would be supported. In consultation with the chart’s author, it

was decided that keeping each phrase together on a line was not an essential feature of the presentation form of the chart.

None of the three original charts uses row numbering (a, b, etc. for multiple rows within a sentence). This feature was added to the rendering step to demonstrate that it could be done, since it can be useful for referring to text in the chart. Neither sentence nor row numbering requires any additional information in the abstract chart; the rendering step simply counts the sentences and rows that are there. A setting in the aesthetic configuration component allows row numbering to be turned on or off.

5.7 Summary

This chapter has demonstrated the use of the abstract chart model by presenting an encoding of the original charts into XML documents conforming to the abstract chart DTD, a description of an XSL stylesheet for rendering these documents, and the final HTML presentation forms. Chapter 6 summarizes our findings and suggests directions for future research.

CHAPTER 6.

CONCLUSIONS

This chapter enumerates some possible directions for future work, starting with the generation of an abstract chart from an annotated text. Then it summarizes the results of this thesis.

6.1 Generating an abstract chart from annotated text

Rendering an abstract chart into a presentation form is only part of the task of helping a linguist produce a visually appealing and reusable display. There is also a need for a convenient way to produce the abstract chart in the first place. The abstract chart instances described so far were created by hand in XML. This requires a knowledge of XML and of XML editing tools, and is time-consuming and error-prone. In section 1.3 it was mentioned that an abstract chart might be created using a spreadsheet-like chart creation program, or it might be generated automatically from an annotated text. In this section we look briefly at how an abstract chart was generated from a sample annotated text by means of an XSL stylesheet.

An *annotated text* (see footnote 12 on linguistic annotation) is a document containing text along with linguistic annotations that encode descriptive or analytic information provided by a linguist. Annotations can include grammatical tagging, parse trees, links from words to a lexicon, freeform notes, participant identification, and other items. The

information in an annotated text is chart-neutral. That is, many charts (and other views) could be generated from a given annotated text: one emphasizing the tense/mode/aspect of verbs, another aligning the constituents of clauses to highlight variations in order, and another tracing the occurrence of "mystery particles."[31] Each of these desired views could be a basis for generating a distinct abstract chart from the same annotated text. Though the source text and annotations would be the same, each abstract chart would contain a different subset of the information in the annotated text.

As part of this thesis project, sample annotated text DTDs were designed (in two parts: `text.dtd` and `annotations.dtd`[32]), and a small annotated text was encoded in XML (`bala-tawip-text.xml` and `bala-tawip-annotations.xml`) using these DTDs. Then a prototype XSL stylesheet was written to extract the data from the annotated text relevant to a constituent chart and transform it into an abstract chart XML document (`bala-tawip-cc.xml`). These files are not presented here but can be found at Huttar (2003).

The principal annotations in the `bala-tawip-annotations.xml` document are constituent-type and participant-reference annotations. The former associate a range of text with a particular constituent type (from a fixed set of possible values), which allows the text to be mapped into a particular column as the abstract chart is created. The latter are mapped to spans with `featureID`s corresponding to the referenced participants. The following

[31] Particles "whose meanings at the sentence level are either unclear or variable" (Longacre and Hwang, n.d., §1.2).

[32] Annotations are modeled using *standoff* markup, i.e. markup separated from the text, mainly due to the desire to allow the text to be updated and corrected independently of the annotation without breaking the links between the two. See Thompson and McKelvie (1997:§1) for more on the advantages of standoff markup.

diagram (Fig. 23) shows the role of the annotations and text in the broader framework. The dashed rectangle at the right illustrates the process as described up to this point in the thesis.

Note that the generation of the abstract chart involves another configuration document, `cc-config-lars.xml`. This file consists mainly of mappings from annotations (particularly constituent-type annotations) to column IDs.

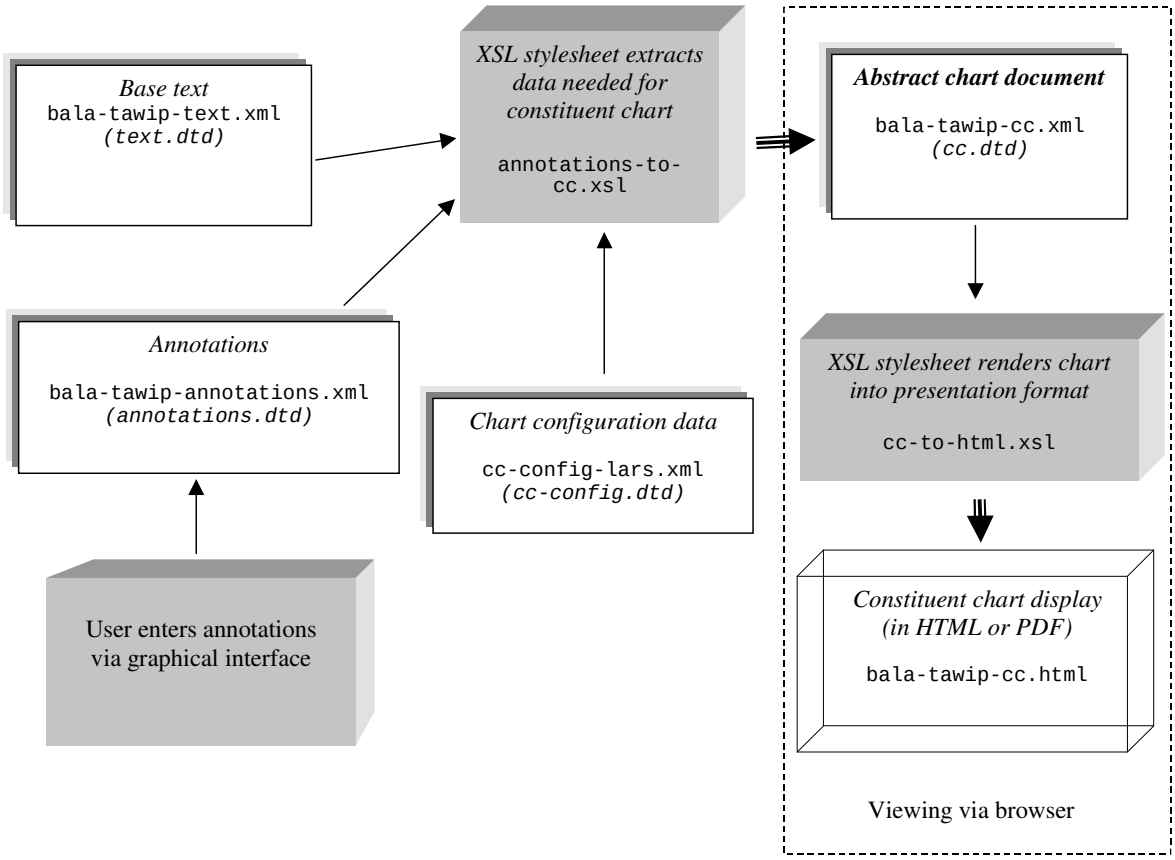

Fig. 23. Broader framework, including annotated text.

6.2 Future directions

There are a number of ways that the abstract chart information model and presentation model could be developed further or built upon. In the next three subsections, ways to improve the information model are presented, followed by possibilities for

improving the presentation model, and finally, tasks that go beyond the abstract constituent chart information and presentation models. These items are numbered as a continuation of the requirements list (§3.3).

6.2.1 Enhancements to the information model.

21. **Normalization of DTD.** Currently the DTD for the abstract chart information model includes several redundant attributes that make processing of the chart simpler or more efficient (e.g. `cell/@glossed`, `column/@position`, and `column/@rank`). It would be desirable for the DTD to be normalized by getting rid of this redundant data. The result would be a simpler DTD that would make it easier for people or software to create abstract charts. Then an XSLT stylesheet could be created to transform a normalized abstract chart into a denormalized one (i.e. to add redundant attributes so that the chart is easier to process). The denormalized chart could then be rendered by the existing stylesheets into presentation forms. This design resembles the processing model described in Bow, Hughes, and Bird (2003:Fig. 29), in that the abstract document is transformed into an intermediate-level document before the rendering step occurs.
22. **Support for right-to-left scripts.** Since significant work has been and continues to be done on discourse analysis with regard to Hebrew[33], not to mention the many languages that use Arabic-based scripts, it would be important for a constituent chart model to support right-to-left (RTL) text and layout. Further study of this issue is

[33] E.g. Longacre (1989). Robert Longacre and Andrew Bowling are currently working on a discourse modular grammar of biblical Hebrew.

required, but in the simplest case, where the LTR/RTL directionality of all columns is uniform, the only required modification to the information model would be the addition of a chart-global parameter specifying the chart's directionality. Hopefully it would not be necessary for different groups of columns within the same chart to have different layout directions. For mixed LTR and RTL text (bidirectional text) within a cell, no extension to the information model is needed assuming that the directionality information associated with each Unicode character can be relied upon. (See also paragraph 27 below.)

23. **Column descriptions.** While the analytic configuration for a chart includes a caption for each column or column group, there may be a need for more extensive documentation of how each column or column group is used in a project. This need increases when, in the interests of saving space (a perennial pressure in discourse charting), a column is used for more than one purpose, but the caption does not have space to describe all of them. This would be especially important for a distributed project where several researchers, sharing an analytic configuration, are charting texts in ways meant to be directly comparable when the charting is done. If the analytic configuration included a specific description of the intended usage of each column and column-group, then constituent charting software could make use of this description to provide help during charting (e.g. via popup help windows) . The descriptions could also be included in a "Chart Notes" section of a detailed presentation view of the chart, helping a later reader understand what was intended by the placement of a constituent in a particular column.

24. **Analysis language.** In connection with the above, it may be desirable to encode somewhere in the abstract chart (perhaps in the metadata) the analysis language and the language under study in the chart. The Dublin Core `<language>` element, the OLAC Metadata standard, and XML's `xml:lang` attribute provide starting points for how to model this information. The analysis language could then be taken as the default for column captions, analytical notes, glosses, and so on; and the language under study would be the default for the vernacular text.
25. **Morpheme: affix vs. stem.** It may be desirable to model morphemes more richly, to distinguish stems from affixes, for example. This would enable the rendering process to be more intelligent about rendering morphemes in interlinear text (IT), e.g. to place a hyphen on the affix rather than on the stem in the following IT fragment:

    bala -si

    child 3Posd

    With the possible addition of a few small enhancements like the above, it seems likely that this vernacular/gloss model for IT would prove sufficient for most uses within a constituent chart, where IT use is typically limited. However if one were to require more complex IT, then rather than extend the information model for constituent charts to also fully encompass IT, it would be desirable to use an existing information model of IT such as those developed in Schmidt (2002) and Bow, Hughes, and Bird (2003).
26. **Styles configurable in data.** It would be helpful to be able to modify the appearance of styles to a large extent by modifying the aesthetic configuration, instead of

modifying the XSL stylesheet (in other words, the style data should be separated from the stylesheet design). This would apply to span styles and moved text (source and destination); such a separation is already supported for row and column styles and zero marker rendering. Note that formatting would have to be specified in an output-language-neutral way, i.e. a way that can be translated by a stylesheet into whatever output format language might be used. However it should also be possible to have different styles for the various kinds of output, e.g. for HTML vs. PDF, and for monochrome vs. color, and screen vs. paper. For this purpose, the style data in the aesthetic configuration might serve as defaults, which can be overridden by the various stylesheets. Or one could design a mechanism whereby a main, default aesthetic configuration is inherited and overridden by other aesthetic configuration documents.

6.2.2 Enhancements to the presentation model.

27. **Right-to-left rendering.** As mentioned in paragraph 22, RTL text would necessitate some changes in the models—most of them in the presentation model. When an abstract chart is identified as being RTL, the presentation model should map the logical order of columns in the analytic configuration to right-to-left order in the presentation form. A consequence of this column layout order is that cells too will be laid out right-to-left. Any mixture of LTR notes (e.g. “[in O]”) with RTL text will require somewhat more complex processing in order to be displayed correctly. Flexible rendering support for non-Roman text, such as that being worked on in Project SILA (Tang 2003)—an integration of the Graphite extensible text rendering

system (Lyons 2001) into the Mozilla browser—will make graceful rendering of RTL and bidirectional text using standard presentation languages considerably easier.

28. **Metadata in generated output.** Because the generated presentation output can be saved in a file that could get separated from its sources, it would be helpful to embed more metadata in the output presentation form. For example when generating HTML, one could embed metadata in the document using `<meta>` elements (which are usually invisible when viewing the document in a browser).
29. **Compatibility with more browsers.** The output HTML from the rendering stylesheet currently works only in Internet Explorer. It should be verified that the generated HTML is using only standard features, and if it is not, it should be fixed. Ideally, the output should work with most standards-compliant browsers.
30. **XSL-FO output.** Design rendering stylesheets to use XSL-FO for output instead of HTML. This would allow more flexible wrapping of interlinear text. (These stylesheets would complement the HTML stylesheets; HTML might still be preferable for web browsing and interactive rendering because of its relative rendering speed.) Using XSL-FO for output would provide greater control over layout in several other ways as well, such as controlling pagination, which would make it more suitable than HTML for rendering in print.

6.2.3 Beyond the abstract chart model.

31. **Abstract chart editor.** Develop a software tool for creating and editing abstract charts (a “discourse charting spreadsheet”). Such software should include templates

for analytic and aesthetic configurations, designed to be useful starting points for a charting project.

32. **Generate abstract chart from standard annotated text.** Following the prototype described in §6.1, instead of the provisional annotated text model, select a standard annotated text model such as XCES, that can accommodate the kinds of annotation needed for constituent charting; preferably one with good, free, existing tools that do the type of text annotation required. Develop a stylesheet that produces an abstract chart from this standard annotated text format.
33. **Support Thurman charts.** Explore how the abstract constituent chart model could be extended to produce an information model for Thurman charts (Grimes 1975:33), a.k.a. information charts (Hohulin 2001). A simple Thurman chart has been encoded and rendered using the abstract constituent chart DTD and XSL stylesheet. While the adequacy of this approach has not been verified for Thurman charts in general, this small experiment suggests that little more would be required in order to handle Thurman charts as well as constituent charts.

6.3 Results

This thesis has defined an information model for an abstract constituent chart. The model is expressed as an XML DTD. This model has been shown to be adequate for capturing the information communicated by three representative charts. These charts were encoded in XML using the abstract chart DTD. A presentation model for rendering the charts into presentation forms was developed and expressed as an XSL stylesheet. The rendered

charts are displayed in Fig. 20-Fig. 22. The information and presentation models were corrected and updated until the authors of the original charts were satisfied with the results.

By providing both an information model and a separate presentation model, this thesis lays the groundwork for specialized software that processes the actual information involved, not just a visual form of it, while providing multiple, configurable views of the data. This property greatly increases the potential to reuse, share, and repurpose both the software and the results of discourse analysis research.

REFERENCES CITED

Association for Linguistic and Literary Computing. 2003. Literary and Linguistic Computing. Oxford University Press. Available: http://www3.oup.co.uk/litlin/scope/. 6 July 2003.

Bański, Piotr. 2001. "The proposed encoding scheme for the IPI PAN corpus." *IPI PAN Reports 936, December 2001.* Available: http://dach.ipipan.waw.pl/CORPUS/banski_raport.rtf. 8 July 2003.

Bateson, M. C. 1964. "Morphological Continuity in Poetry." Paper presented at the summer meeting of the *Linguistic Society of America*.

Beekman, John, John C. Callow, and Michael F. Kopesec. 1981. *The Semantic Structure of Written Communication*. Dallas: SIL.

Benveniste, E. 1974. *Problemès de Linguistique Générale*, vol. 2. Paris: Gallimard.

Berners-Lee, Tim, and D. Connolly. 1995. "Hypertext Markup Language - 2.0. Request for Comments 1866." Internet Engineering Task Force. Available: http://www.ietf.org/rfc/rfc1866.txt. 5 July 2003.

Bird, Steven and Mark Liberman. 1999a. "Annotation graphs as a framework for multidimensional linguistic data analysis." *Towards Standards and Tools for Discourse Tagging, Proceedings of the Workshop*:1-10. Association for Computational Linguistics.

---. 1999b. "A formal framework for linguistic annotation." *Technical Report MS-CIS-99-01.* Linguistic Data Consortium. Philadelphia: University of Pennsylvania.

---. 5 Dec. 2001 [last update]. "Linguistic Annotation." Linguistic Data Consortium. Available: http://www.ldc.upenn.edu/annotation/.

Bird, Steven, Kazuaki Maeda, Xiaoyi Ma, Haejoong Lee, Beth Randall, and Salim Zayat. 2002. "TableTrans, MultiTrans, InterTrans and TreeTrans: Diverse Tools Built on the Annotation Graph Toolkit." *Proceedings of the Third International Conference on Language Resources and Evaluation,* Paris: European Language Resources Association.

Bird, Steven and Gary F. Simons. 2003. "Seven Dimensions of Portability for Language Documentation and Description." To appear in *Language* 79. Preprint available: http://www.language-archives.org/documents/portability.pdf. 15 July 2003.

Bos, Bert. 13 Nov. 2001 [last update]. "XML in 10 points." W3C. Available: http://www.w3.org/XML/1999/XML-in-10-points.

Bow, Cathy, Baden Hughes, and Steven Bird. 2003. "Towards a General Model of Interlinear Text." Paper presented at *EMELD Language Digitization Project Conference 2003, Workshop on Digitizing and Annotating Texts and Field Recordings*. LSA Institute, Michigan State University, July 11th-13th 2003. Available: http://saussure.linguistlist.org/cfdocs/emeld/workshop/2003/bowbadenbird-paper.html.

Bray, Tim, Jean Paoli, C. M. Sperberg-McQueen, and Eve Maler, eds. 6 Oct. 2000. Extensible Markup Language (XML) 1.0 (Second Edition). W3C. Available: http://www.w3.org/TR/REC-xml.

Burnard, Lou. 8 Oct. 1999. TEI Pizza Chef. TEI Consortium. Available: http://www.tei-c.org/pizza.html.

Carletta, Jean, David McKelvie, Amy Isard, Andreas Mengel, Marion Klein, and Morten Baun Møller. 2002. "A Generic Approach to Software Support for Linguistic Annotation Using XML." In G. Sampson and D. McCarthy (eds.), *Readings in Corpus Linguistics,* London and NY: Continuum International. Available: http://www.cogsci.ed.ac.uk/~jeanc/readings-in-corpling.final.webformat.pdf.

Chamberlin, Donald Dean, Helmut F. Hasselmeier, Dieter P. Paris. 1988. "Defining Document Styles for WYSIWYG Processing." In J.C. van Vliet (ed.), *Document Manipulation and Typography.* Proceedings of the *International Conference on Electronic Publishing, Document Manipulation and Typography, Nice (France), April 20-22 1988.* Cambridge Series on Electronic Publishing. Cambridge: Cambridge University Press. 121-137.

Clark, James, ed. 16 Nov. 1999. "XSL Transformations (XSLT) Version 1.0." W3C. Available: http://www.w3.org/TR/xslt.

Coombs, James H., Allen H. Renear, and Steven J. DeRose. 1987. "Markup Systems and the Future of Scholarly Text Processing." *Communications of the Association for Computing Machinery* 30: 933-947. Available: http://www.oasis-open.org/cover/coombs.html.

Cover, Robin. 10 June 2003 [last update]. "Patents and Open Standards." *The Cover Pages.* OASIS. Available: http://xml.coverpages.org/patents.html.

Cromack, Robert E. 1968. *Language Systems and Discourse Structure in Cashinawa. Hartford Studies in Linguistics* 23. Hartford: Hartford Seminary Foundation.

Dashofy, Eric M., André van der Hoek, and Richard N. Taylor. "A Highly-extensible, XML-Based Architecture Description Language." In *Proceedings of the Working IEEE/IFIP Conference on Software Architectures (WICSA 2001)*, Amsterdam, Netherlands. Available: http://www.ics.uci.edu/~edashofy/papers/wicsa2001.pdf.

Day, David, John Aberdeen, Lynette Hirschman, Robyn Kozierok, Patricia Robinson and Marc Vilain. 1997. "Mixed-Initiative Development of Language Processing Systems." In *Fifth Conference on Applied Natural Language Processing, Association for Computational Linguistics, 31 March-3 April 1997, Washington, D.C.* Available: http://www.mitre.org/tech/alembic-workbench/ANLP97-bigger.html

Dooley, Robert A. and Stephen H. Levinsohn. 2001. *Analyzing Discourse: a Manual of Basic Concepts*. Dallas: SIL International.

Dundes, Alan. 1962. "From Etic to Emic Units in the Structural Study of Folktales." *Journal of American Folklore* 75, 95-105.

---. 1963. "Structural Typology in North American Indian Folktales." *Southwestern Journal of Anthropology* 19, 121-30.

---. 1964. *The Morphology of North American Indian Folktales*. Helsinki: Academia Scientiarum Fennica.

Dybkjær, L. and N.O. Bernsen. 2000. "Towards Corpus Annotation Standards: the MATE Workbench." Proceedings of the *COCOSDA Workshop 2000, Beijing, China, 2000.* 31-55. Available: http://www.tsc.uvigo.es/~carmen/bego/data/COCOSDA-20.10.2000-F.pdf. 10 July 2003.

EAGLES. 20 March 2000 [last update]. "Corpus Encoding Standard, Version 1.5." Expert Advisory Group on Language Engineering Standards (EAGLES). Available: http://www.cs.vassar.edu/CES/CES1.html.

Fallside, David, ed. 2 May 2001. "XML Schema Part 0: Primer." W3C. Available: http://www.w3.org/TR/xmlschema-0/.

Givón, Talmy, ed. 1994. *Voice and Inversion*. Amsterdam: John Benjamins.

Gleason, H.A., Jr. 1964. "The Organization of Language: a Stratificational View." *Monograph Series on Languages and Linguistics* 17: 75-95. Washington, D.C.: Georgetown University Press.

---. 1968. "Contrastive Analysis in Discourse Structure." In J.E. Alatis (ed.), *Contrastive Linguistics and Its Pedagogical Implications. Report of the Nineteenth Annual Round Table Meeting on Linguistics and Language Studies. Monograph Series on Languages and Linguistics* 21:39-63. Washington, D.C.: Georgetown University Press.

Goldfarb, Charles F. 1973. *Design Considerations for Integrated Text Processing Systems*. Cambridge, MA: IBM Corporation Cambridge Scientific Research Center.

---. 1996. "The Roots of SGML—A Personal Recollection." In *SGML Source Home Page.* Available: http://www.sgmlsource.com/history/roots.htm. 4 July 2003.

Google. 2003. Google Search: sgml. Available: http://www.google.com/search?q=sgml. 6 July 2003.

Grimes, Joseph E. 1965. "Linguistic and Anthropological Projects Using the Computer." In D. Hymes (ed.), *The Use of Computers in Anthropology*, The Hague: Mouton. 515-516.

---. 1975. *The Thread of Discourse*. The Hague: Mouton.

Gromov, Gregory. 2002. "History of Internet and WWW: The Roads and Crossroads of Internet History." NetValley. Available: http://www.netvalley.com/intvalstat.html. 9 July 2003.

Harbin, Duane. 1998. “Fiat Lux: The Electronic Word.” In Valerie R. Hotchkiss and Charles C. Ryrie (eds.), *Formatting the Word of God: the Charles Caldwell Ryrie collection*. An exhibition at Bridwell Library, Perkins School of Theology, Southern Methodist University, October 1998 - January 1999. Available: http://www.smu.edu/bridwell/publications/ryrie_catalog/xiii_1.htm.

Harper, Kenneth. 1964. “Inter-Sentence Connectivity in Written Discourse.” Paper presented at the meeting of the *Association for Machine Translation and Computational Linguistics*.

Harris, Zellig. 1952. “Discourse Analysis.” *Language* 28: 1-30.

Hohulin, E. Lou. 2001. *Discourse Analysis: A Manual for a Linguistic-Translation Workshop*. Unpublished ms.

Hopper, Paul J. and Sandra Thompson. 1980. “Transitivity in Grammar and Discourse.” *Language* 56: 251-299.

Huttar, Lars. 18 August 2003 [last update]. “Constituent Charting for Discourse Analysis.” Available: http://purl.oclc.org/NET/huttar/lars/ma-thesis.

Hwang, Shin Ja J. 1989. “Recursion in the Paragraph as a Unit Discourse Development.” *Discourse Processes* 12: 461-477.

---. 1993. *Approaching a Narrative: Charting and Chunking*. Unpublished ms.

---. 1997. “A Profile and Discourse Analysis of an English Short Story.” *Language Research* 33 no. 2: 293-320.

Ide, Nancy and Laurent Romary. 2003. “Encoding Syntactic Annotation.” In Anne Abeillé (ed.), *Building and Using Parsed Corpora.* Dordrecht: Kluwer. Available: http://www.cs.vassar.edu/~ide/papers/ide-romary.ps.

Ide, Nancy and Keith Suderman. 2003. “XCES: Corpus Encoding Standard for XML, Version 0.2.” Available: http://www.cs.vassar.edu/XCES/.

InfoWorld. 2003. InfoWorld Search. Available: http://search.infoworld.com/servlet/search?q=xml&r=all&s=0&t=10. 6 July 2003.

Jacobson, S. N. 1964. “Paragraph Structure as an Approach to Mechanized Discourse Analysis.” Paper presented at the meeting of the *Association for Machine Translation and Computational Linguistics*.

Lagoze, Carl, et al., eds. 10 June 2002 [last update]. “Open Archives Initiative Frequently Asked Questions. Open Archives Initiative.” Available: http://www.openarchives.org/documents/FAQ.html.

Lakoff, George P. 1964. “Structure Above the Sentence Level.” Paper presented at the summer meeting of the *Linguistic Society of America*.

Laprun, Christophe, Jonathan G. Fiscus, John Garofolo, and Sylvain Pajot. 2002. “A Practical Introduction to ATLAS.” Paper presented at the *3rd International Conference on Language Resources and Evaluation (LREC),* Las Palmas.

Lasilla, Ora and Ralph R. Swick, eds. 22 February 1999. “Resource Description Framework (RDF) Model and Syntax Specification.” W3C. Available: http://www.w3.org/TR/REC-rdf-syntax/.

Leech, Geoffrey, Martin Weisser, Andrew Wilson and Martine Grice. 1998. “LE-EAGLES-WP4-4: Survey and Guidelines for the Representation and Annotation of Dialogue.” In D. Gibbon, I. Mertins, and R.K. Moore (eds.), *Handbook of Multimodal and Spoken Dialogue Systems: Resources, Terminology and Product Evaluation,* Boston: Kluwer. 1-101. Available: http://www.ling.lancs.ac.uk/eagles/delivera/wp4final.htm.

Lie, Håkon Wium and Bert Bos. 11 Jan. 1999 [last update]. “Cascading Style Sheets, Level 1.” W3C. Available: http://www.w3.org/TR/1999/REC-CSS1-19990111. 24 July 2003

Longacre, Robert E. 1964. *Grammar Discovery Procedures: A Field Manual. Janua Linguarum, series minor* 33. The Hague: Mouton.

---. 1965. “Some Fundamental Insights of Tagmemics.” *Language* 41: 65-76.

---. 1968. *Discourse, Paragraph, and Sentence Structure in Select Philippine Languages*, vol. 1. Santa Ana, CA: SIL.

---. 1972. *Hierarchy and Universality of Discourse Constituents in New Guinea Languages*, 2 vols. Washington, D.C.: Georgetown University Press.

---. 1979. “Why We Need a Vertical Revolution in Linguistics.” *LACUS Forum* 5.247-70.

---. 1981. “A Spectrum and Profile Approach to Discourse Analysis.” *Text* 1.337-59.

---. 1989. *Joseph: A story of divine providence: A text theoretical and textlinguistic analysis of Genesis 37 and 39-48.* Winona Lake: Eisenbrauns.

---. 1990. *Storyline Concerns and Word-order Typology in East and West Africa. Studies in African Linguistics,* supplement 10. Los Angeles: The James S. Coleman African Studies Center & Dept. of Linguistics, University of California.

--. 1992. “Natural Text Processing and Text Meaning.” In Maxim Stamenov (ed.), *Current Advances in Semantic Theory*. Current issues in linguistic theory, 73. Amsterdam: John Benjamins. 521-34.

---. 1993. “Paul Ricoeur’s Philosophy and Textlinguistic Analysis.” In Peter A. Reich (ed.), *The Nineteenth LACUS Forum* 1992. Illinois: Linguistic Association of Canada and the United States. 47-55.

---. 1996. *The Grammar of Discourse*. 2nd ed. New York: Plenum Press.

Longacre, Robert E. and Shin Ja J. Hwang. n.d. *Discourse Analysis/Textlinguistics: a Field Manual*.

---. 1994. “A Textlinguistic Approach to the Biblical Hebrew Narrative of Jonah.” In Robert D. Bergen (ed.), *Biblical Hebrew and discourse linguistics*. Dallas: SIL. 336-58.

Longacre, Robert E. and Stephen H. Levinsohn. 1978. “Field analysis of discourse.” In Wolfgang U. Dressler (ed.), *Current Trends in Textlinguistics.* Berlin: de Gruyter. 103-22.

Longacre, Robert E. and Frances M. Woods, eds. 1976-77. *Discourse Grammar: Studies in Indigenous Languages of Colombia, Panama, and Ecuador, 1-3.* SIL Publications in Linguistics and Related Fields, 52(1-3). Dallas: SIL and the University of Texas at Arlington.

Loriot, James, and Barbara Hollenbach. 1970. “Shipibo Paragraph Structure.” *Foundations of Language* 6: 43-66.

Lyons, Melinda. 2001. "Graphite and WorldPad: Tools for writing the World's Other Languages." TechKnowLogia 3 No. 6:51-54. Available: http://www.techknowlogia.org/TKL_active_pages2/CurrentArticles/main.asp?FileType=PDF&ArticleID=349

Mann, W. C., and S. A. Thompson. 1986. “Relational Propositions in Discourse.” *Discourse Processes* 9: 57-90.

Miller, Eric, ed. 12 May 2003. “Semantic Web Activity Statement.” W3C. Available: http://www.w3.org/2001/sw/Activity.

National Institute of Standards and Technology. 6 Feb. 2003 [last update]. “ATLAS project: ATLAS overview.” Available: http://www.nist.gov/speech/atlas/overview.html.

O’Donnell, Matthew B., Stanley E. Porter, and Jeffrey T. Reed. 2001. “OpenText.org: An Experiment in Internet-based Collaborative Humanities Scholarship.” Paper presented at *2001 Joint International Conference of the ACH and ALLC*, New York University, 15 June 2001.

O’Donnell, Michael. 2000. “RSTTool 2.4: A Markup Tool for Rhetorical Structure Theory.” *Proceedings of the International Natural Language Generation Conference (INLG'2000),* 13-16 June 2000, Mitzpe Ramon, Israel. 253-256.

Partridge, Kathryn J. 1995. “A Discourse Study of Tense-Aspect in Narrative Sections of Hebrew Poetry.” Master’s thesis, University of Texas at Arlington.

Pickett, Velma B. 1959. *The Grammatical Hierarchy of Isthmus Zapotec*, Dissertation. University of Michigan.

Pike, Kenneth L. 1954. *Language in Relation to a Unified Theory of the Structure of Human Behavior. Part I.* Glendale, CA: SIL.

---. 1964a. “Beyond the Sentence.” *College Composition and Communication* 15: 129-35.

---. 1964b. “Discourse Analysis and Tagmeme Matrices. “*Oceanic Linguistics* 3: 5-25.

---. 1967. *Language in Relation to a Unified Theory of the Structure of Human Behavior. Janua Linguarum, series maior,* 24. The Hague: Mouton.

Propp, Vladimir. 1968. *Morphology of the Folktale.* 2nd ed. Trans. Lawrence Scott. Austin: University of Texas Press.

Quick, Philip. 1996. “Multilinear Discourse Analysis Software Demonstration.” In H. Andrew Black, Alan Buseman, David Payne, Gary F. Simons (eds.), *Proceedings of the 1996 General CARLA Conference, November 14-15.* Waxhaw, NC/Dallas, TX: JAARS and SIL. 291-309.

Raymond, Eric S. 2003. “The Open Source Case for Customers.” *Open Source Initiative.* Available: http://www.opensource.org/advocacy/case_for_customers.php. 17 June 2003.

Reid, Aileen, Ruth Bishop, Ella Marie Button, and Robert E. Longacre. 1968. *Totonac: from Clause to Discourse. SIL Publications in Linguistics* 17. Norman: SIL of the University of Oklahoma.

Schiffrin, Deborah, Deborah Tannen, and Heidi E. Hamilton, eds. 2001. *The Handbook of Discourse Analysis.* Oxford: Blackwell Publishers.

Schmidt, Thomas. n.d. “Visualizing Linguistic Annotation as Interlinear Text.” To appear in *Arbeiten zur Mehrsprachigkeit (Working Papers in Multilingualism),* Serie B. Hamburg.

SGML Users’ Group. 11 June 1990. “A Brief History of the Development of SGML.” Available: http://www.sgmlsource.com/history/sgmlhist.htm.

Simons, Gary F. 1993. “Specifications for a User Requirements Document.” In *CELLAR Project Status Report #7.* Unpublished internal document.

---. 1997a. “Conceptual Modeling Versus Visual Modeling: a Technological Key to Building Consensus.” *Computers and the Humanities* 30, no. 4: 303-319. Available: http://www.sil.org/cellar/ach94/ach94.html.

---. 1997b. “PTEXT: A Format for the Interchange of Parsed Texts Among Natural Language Processing Applications.” *SIL Electronic Working Papers* 1997-008, December 1997. Available: http://www.sil.org/silewp/1997/008/silewp1997-008.html

---. 1998. “The Nature of Linguistic Data and the Requirements of a Computing Environment for Linguistic Research. In John M. Lawler and Helen Aristar Dry (eds.), *Using Computers in Linguistics: a Practical Guide*, London and New York: Routledge. 10-25.

Simons, Gary F. and Stephen Bird, eds. 11 Dec. 2002 [last update]. “OLAC Metadata.” Available: http://www.language-archives.org/OLAC/metadata.html.

Simons, Gary F. and John V. Thomson. 1995. "Multilingual Data Processing in the CELLAR Environment." Paper presented at *Linguistic Databases,* 23-24 March 1995, University of Groningen. Available: http://www.sil.org/cellar/mlingdp/mlingdp.html.

Sol, Selena. 8 March 1999. "History of XML." *Web Developer's Virtual Library.* Available: http://wdvl.internet.com/Authoring/Languages/XML/Tutorials/Intro/history.html.

Sperberg-McQueen, C. M. and L. Burnard, eds. 1994. *Guidelines for Electronic Text Encoding and Interchange (TEI P3).* Chicago and Oxford: ACH/ACL/ALLC.

---, eds. 2002. *TEI P4:* Guidelines for Electronic Text Encoding and Interchange. Oxford: TEI Consortium. Available: http://www.tei-c.org/P4X/.

Spielmann, Kent. 2000. *A Discourse Analysis Tool Wishlist.* Unpublished ms.

Stennes, Leslie. 1969. *Participant Identification in Adamawa Fulani.* Hartford, Connecticut: Hartford Seminary Foundation.

Stutzman, Verna. 2003. "Lexical Database and Interlinear Text Tools." Unpublished manuscript.

Syntext. 2003. "Syntext Serna: Portable True WYSIWYG XML Editor." Available: http://www.syntext.com/products/serna/index.htm. 25 June 2003.

Taber, Charles R. 1966. *The Structure of Sango Narrative.* Hartford, CT: Hartford Seminary Foundation.

Tang, Frank Yung-Fong. 2003. "Project SILA: SIL.ORG Graphite and Mozilla Integration Project." Available: http://sila.mozdev.org/. 22 July 2003.

TEI Consortium. 6 Jan. 2003a [revision date]. "Welcome to the TEI Website." Available: http://www.tei-c.org/.

---. 3 Feb. 2003b [revision date]. "What is the TEI Consortium?" Available: http://www.tei-c.org/Consortium/index.html.

Thompson, Henry S. and David McKelvie. 1997. "Hyperlink Semantics for Standoff Markup of Read-only Documents." In *Proceedings of SGML Europe '97, Barcelona, Spain, May 1997.* Available: http://www.ltg.ed.ac.uk/~ht/sgmleu97.html.

Thurman, Robert C. 1975. “Chuave medial verbs.” *Anthropological Linguistics* 17 no. 7: 342-52.

Vetch, P. H. 20 Sept. 2001. “Association for Literary and Linguistic Computing: Honorary Members.” Available: http://www.kcl.ac.uk/humanities/cch/allc/refdocs/honmems.htm.

Weber, David J. 2000. “E.g.” Paper presented at *Workshop on Web-based Language Documentation and Description, 12-15 December, 2000.* Philadelphia, USA. Available: http://www.ldc.upenn.edu/exploration/expl2000/papers/weber/weber.pdf.

Weinrich, H. 1964. *Tempus: Besprochene und Erzählte Zeit.* Stuttgart: Kohlhammer.

CURRICULUM VITAE

Lars Huttar was born in Paramaribo, Suriname on September 26, 1969, and is an American citizen. He graduated from Duncanville High School in 1987. He received his Bachelor of Arts degree in Computer Science from Oberlin College, Ohio in 1991. He was employed as a software developer in Osaka, Japan and Dallas, Texas for three years. Subsequently, he taught ESL in Central Asia for five years, serving as a Team and Country Administrator and starting new programs at two schools.

In 1999, he entered the M.A. program in Applied Linguistics (Translation) at the Graduate Institute of Applied Linguistics in Dallas, Texas. While studying he made contributions to SIL International's Language Software Development group in the area of software development and testing. He has presented papers on his thesis work in progress at the University of Texas at Arlington Student Conference in Linguistics, February 2003, and at the Discourse Analysis Workshop at the International Linguistics Center in Dallas, Texas in September 2001. He and his wife Kathryn were married on September 21, 2002.