University of the Arts London

Transforming structured descriptions to visual representations. An automated visualization of historical bookbinding structures

Alberto Campagnolo

Thesis for the degree of Doctor of Philosophy

Volume 2 - Appendices

May 2015

Table of contents

Volume 2

Appendix A. Human memory system....318

Appendix B. Shape library....321

B.1. Page markers....322
B.2. Board markers....325
B.3. Bookmarks....328
B.4. Endleaves....335
B.5. Sewing....340
B.6. Boards....347
B.7. Spine shape....355
B.8. Spine lining....359
B.9. Endbands....368
B.10. Coverings....383
B.11. Furniture - fastenings....408

Appendix C. Coding example: endleaves....413

C.1. Endleaf XSLT....414
C.2. Endleaf master SVG....473
C.3. Endleaf CSS....476

Appendix A. Human memory system

Data perceived through our senses (sight, hearing, smell, touch, taste) is stored in the sensory memory for a very short length of time (between 200 and 500 milliseconds). Usually this information is passed on to our working memory only if the conscious mind activates an attention behaviour (however, some sensed data can linger for longer in an unconscious manner[502]). In our working memory, different kinds of information (visuospatial, auditory, episodic) are processed by a series of subsidiary systems coordinated by an attentive central executive. It is in the working memory that we process information for then discard it or record it in the long-term memory system for later retrieval.[503]

In our working memory, information dissipates in a matter of seconds if not rehearsed, and its capacity is rather limited. Studies suggest that we can hold in our working memory only around seven chunks of information. This limit seems to be physiological, as it does not seem to change with expertise. Expert minds, in fact, are only apparently capable of holding greater amounts of information in their working memory buffer: they are capable of developing increasingly complex representations of familiar patterns with direct access to data previously encoded in the long-term memory system, thus chunking more familiar and related data within each slot available. Unfamiliar patterns, or random data limit the chunking capacities even of expert minds.[504]

Once processed and retained, information can be encoded into the long-term memory system, where it is stored for hours, days, or even a lifetime — e.g. pro-

502. Brandimonte *et al.* 1992a; Brandimonte *et al.* 1992b; Brandimonte & Gerbino 1996.
503. Baddeley & Hitch 1974; Baddeley 1992; Baddeley 2000; Baddeley 2007; Alloway & Alloway 2012; Baddeley 2013.
504. Miller 1956; Cowan 2001; Saaty & Ozdemir 2003; Gobet & Clarkson 2004; Ericsson & Moxley 2012.

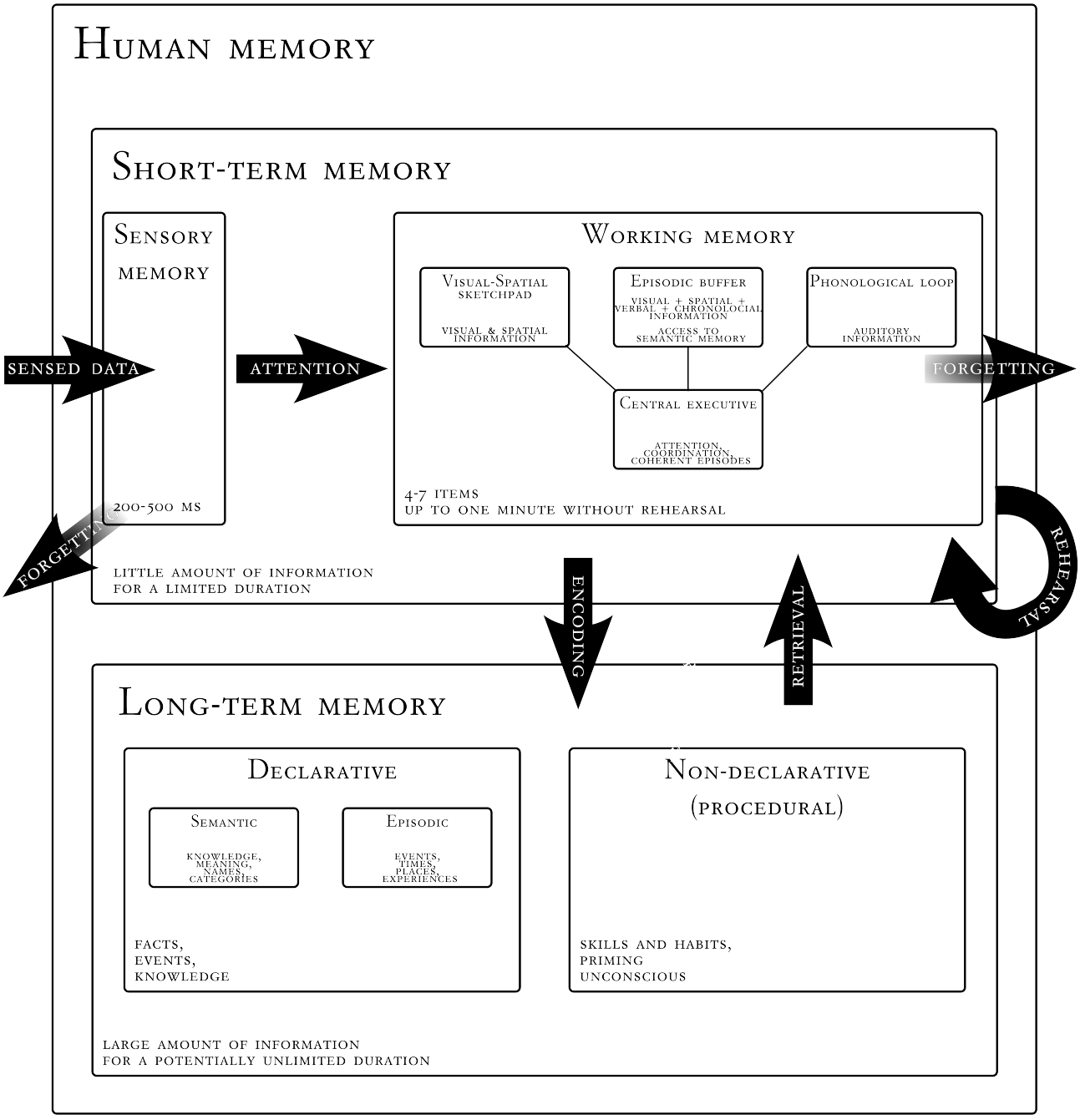


Figure 1. A model of the human memory system (after Baddeley & Hitch 1974; Baddeley 1992; Baddeley 2000; Baddeley 2007; Alloway & Alloway 2012; Baddeley 2013). Sensed data is processed in the sensory memory and, through attention, is passed on to the working memory system, which can store a limited amount of information for a limited amount of time before it is discarded or recorded in the long-term memory system. Our cognitive capabilities are limited by the capacity of our working memory.

cedural and semantic knowledge — and from where it can be retrieved, with more or less facility depending on the frequency in which a piece of information needs to be accessed: the more frequent the use, the easier and quicker the retrieval is.

Amongst the non-declarative kinds of long-term memory, an important phenomenon is that of *priming*. This refers to the implicit memory effect by which, exposure to a stimulus has measurable effects on the response to a later stimulus. This is of particular interest when considering visual stimuli. Once we have seen a visual pattern of some kind, it becomes much easier and quicker for our brain to identify it at a later appearance, and this priming effects can linger for minutes, or even hours. This means that recognition of visual patterns is much easier if the subjects are primed by visually similar images. This effect is strictly image-dependent and not based on high-level semantic information, but it can occur even if information is not consciously perceived.[505]

505. Ware 2013.

Appendix B. Shape library

This appendix contains the diagrams for the various components — and their possible variations — for each bookbinding structure.

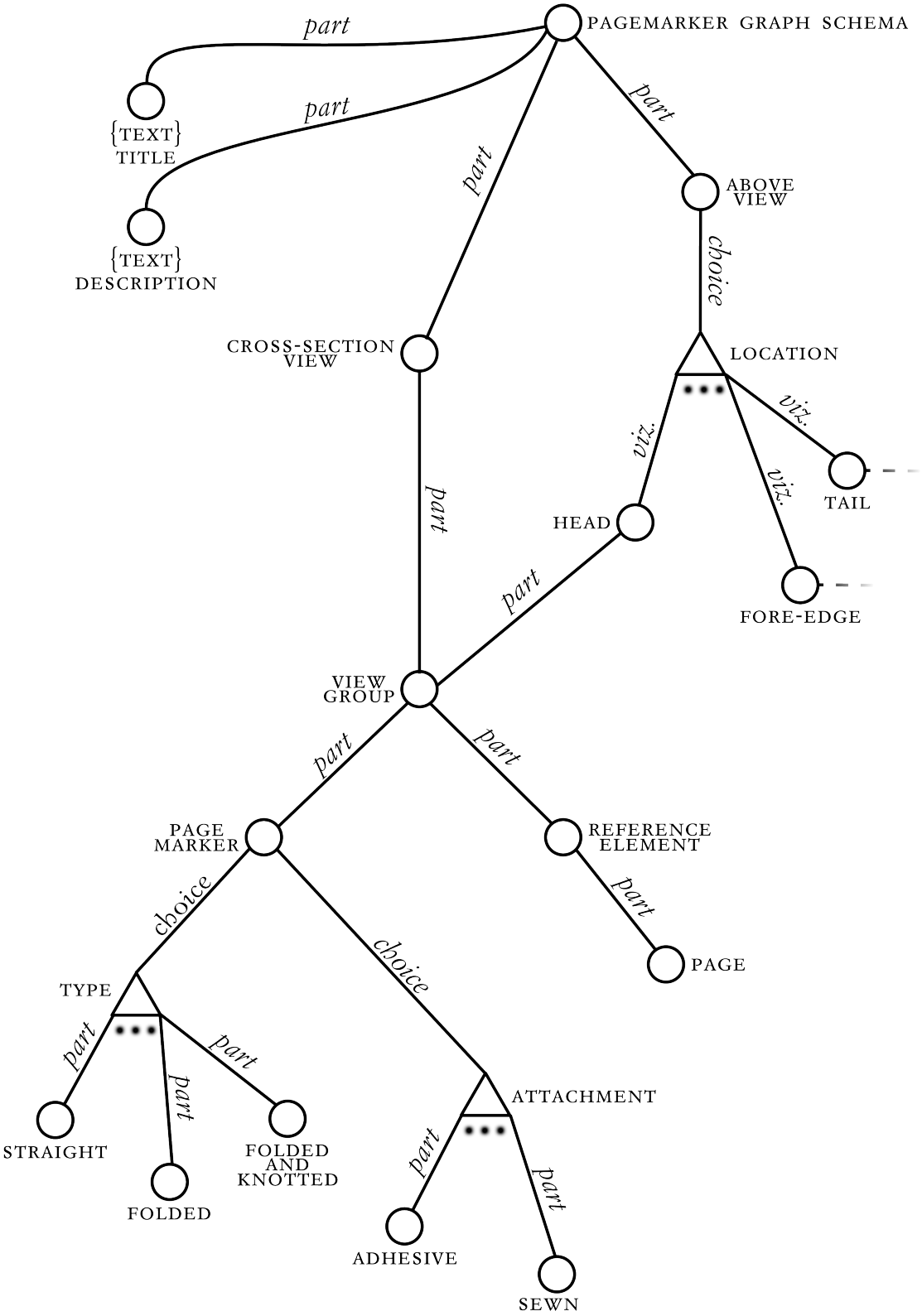
Each section begins with the graph schema for the relevant structure and its description/transformation. As covered in §6.2.1, graph schemas are a graphical representation of the information involved in each bookbinding structure description and transformation. At a glance, the reader can appreciate the complexity of a description/transformation, and graph schemas can also be followed, step by step, each node adding a new piece of information, each choice branching in parallel information paths. Graph schemas present all the information needed for a description/transformation in one page (or series of related pages for very complex structures).

Each graph schema is followed by a series of diagrams that can be generated from the information that it contains. All the diagrams shown have been automatically generated from XML descriptions. Titles and labels are used to relate the diagrams and their elements to nodes in the graph schema.

This appendix is subdivided into sections, one for each bookbinding structure, as follows: (i) page markers, (ii) board markers, (ii) bookmarks, (iv) endleaves, (v) sewing, (vi) boards, (vii) spine shape, (viii) spine lining, (ix) endbands, (x) coverings, (xi) furniture - fastenings

Page markers

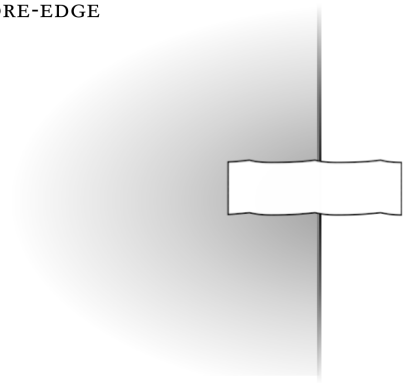
Page markers are those tabs attached to the edges of leaves in a book to indicate important places in the text. See §7.2.1.


PAGEMARKER GRAPH SCHEMA
part
part
part
part
{TEXT}
TITLE
{TEXT}
DESCRIPTION
ABOVE
VIEW
choice
CROSS-SECTION
VIEW
LOCATION
viz.
viz.
viz.
TAIL
HEAD
FORE-EDGE
part
part
VIEW
GROUP
part
part
PAGE
MARKER
REFERENCE
ELEMENT
part
PAGE
choice
choice
TYPE
part
part
part
STRAIGHT
FOLDED
FOLDED
AND
KNOTTED
ATTACHMENT
part
part
ADHESIVE
SEWN

PAGE MARKERS

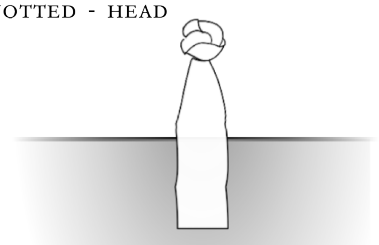
STRAIGHT - FORE-EDGE

ADHESIVE



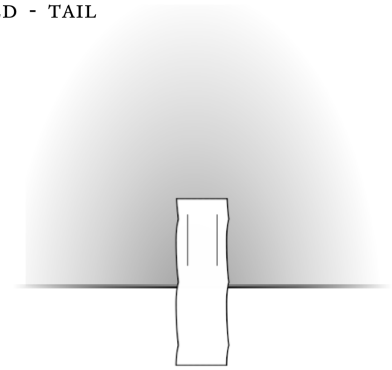
FOLDED & KNOTTED - HEAD

ADHESIVE



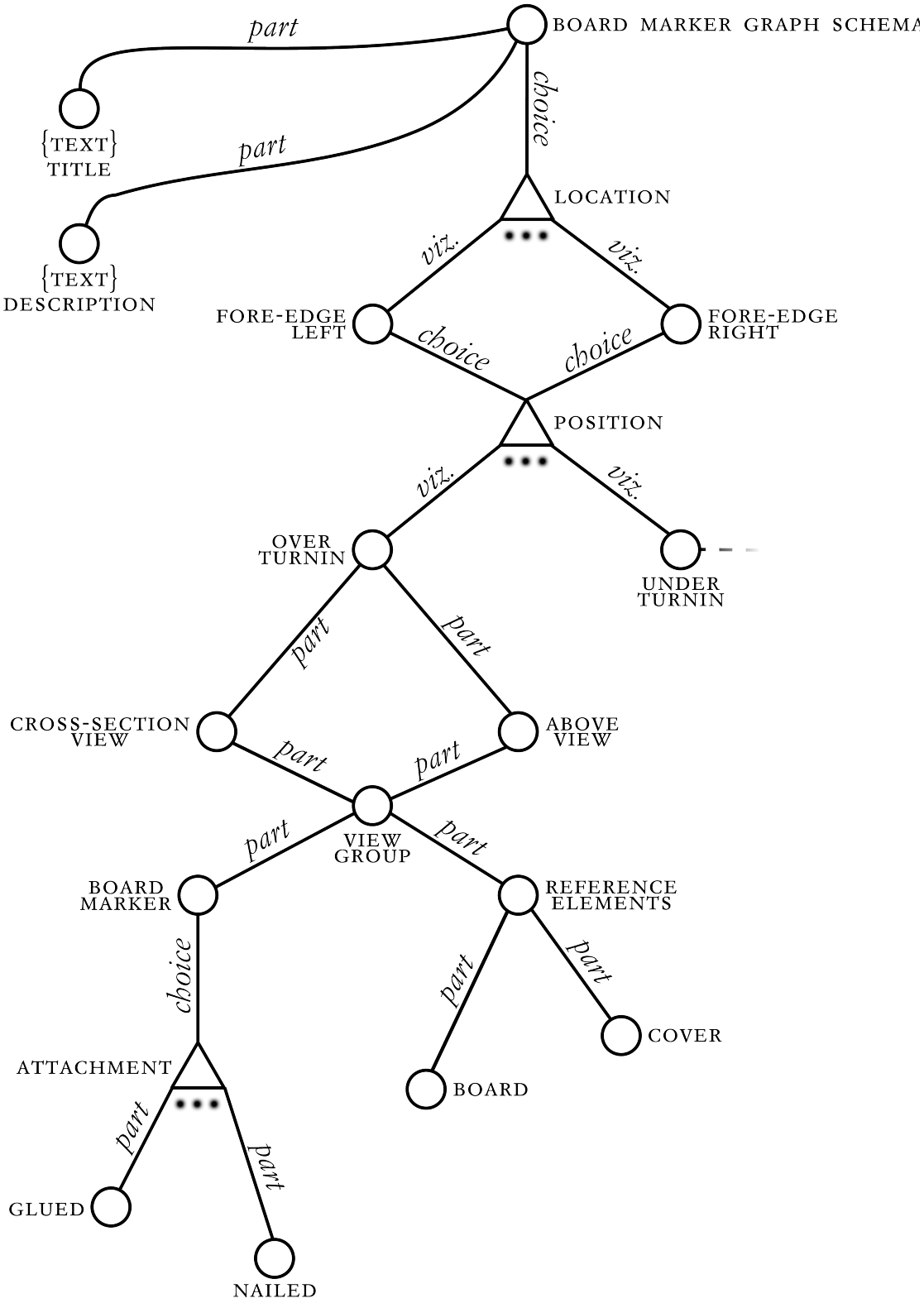
FOLDED - TAIL

SEWN



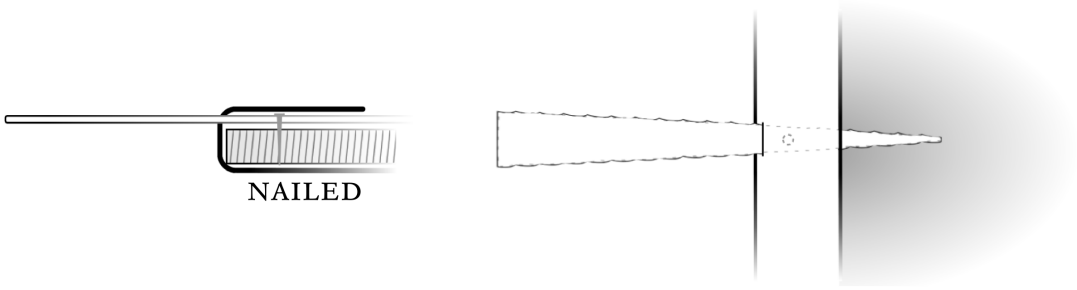
Board markers

Board markers are lengths of animal skin pasted to the inside of the board, projecting from the edge and most probably used as bookmarks. See §7.2.1.


BOARD MARKER GRAPH SCHEMA
part
{TEXT}
TITLE
part
{TEXT}
DESCRIPTION
choice
LOCATION
viz.
viz.
FORE-EDGE
LEFT
FORE-EDGE
RIGHT
choice
choice
POSITION
viz.
viz.
OVER
TURNIN
UNDER
TURNIN
part
part
CROSS-SECTION
VIEW
ABOVE
VIEW
part
part
part
part
VIEW
GROUP
BOARD
MARKER
REFERENCE
ELEMENTS
choice
part
part
COVER
ATTACHMENT
BOARD
part
part
GLUED
NAILED

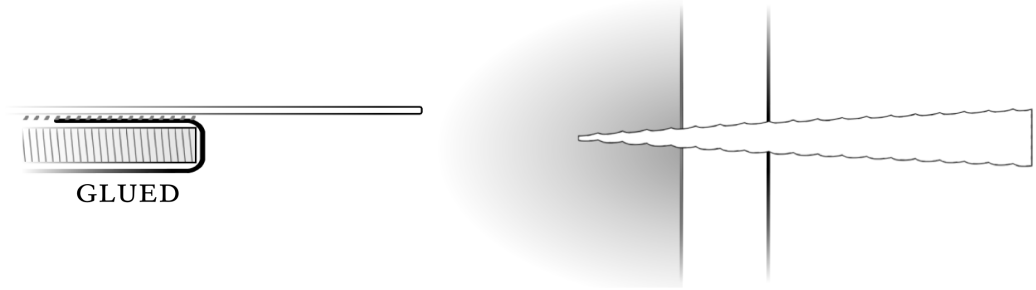
BOARD MARKERS

UNDER TURNIN - FORE-EDGE LEFT



NAILED

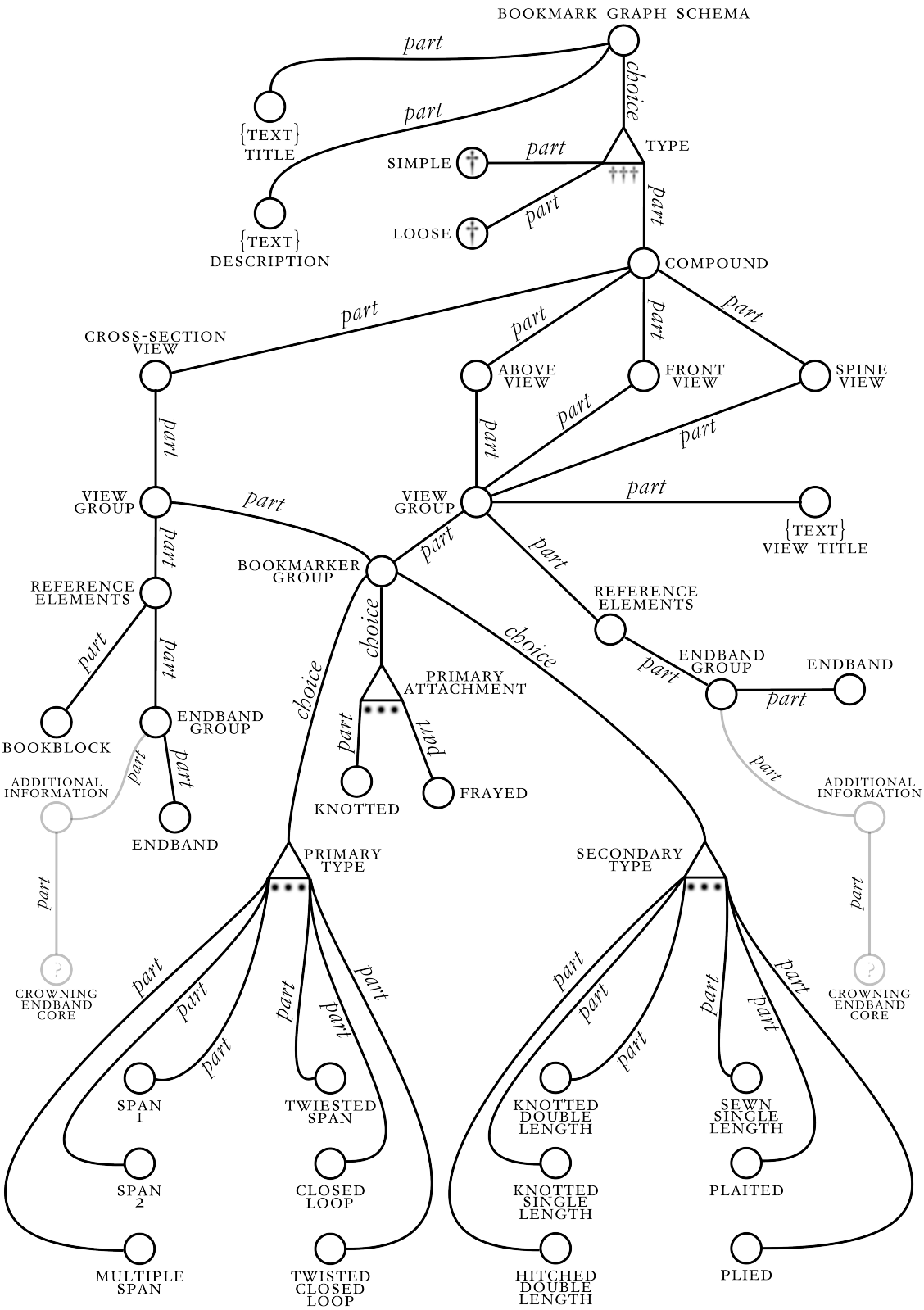
OVER TURNIN - FORE-EDGE RIGHT



GLUED

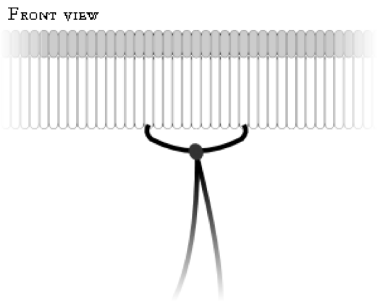
Bookmarks

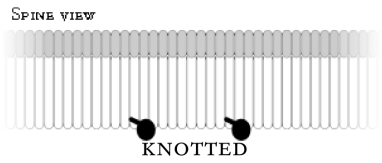
A bookmark is a device used to mark temporarily particular leaves or passages of a book; its formation can range from simple lengths of thread or textile ribbon to multiple elements attached to bars or disks fixed to an endband. See §7.2.1.


BOOKMARK GRAPH SCHEMA
part
choice
{TEXT}
TITLE
part
TYPE
SIMPLE
part
{TEXT}
DESCRIPTION
LOOSE
part
part
COMPOUND
part
part
part
part
CROSS-SECTION
VIEW
ABOVE
VIEW
FRONT
VIEW
SPINE
VIEW
part
part
part
part
part
VIEW
GROUP
part
VIEW
GROUP
part
{TEXT}
VIEW TITLE
part
part
part
BOOKMARKER
GROUP
REFERENCE
ELEMENTS
REFERENCE
ELEMENTS
part
part
choice
choice
choice
part
ENDBAND
GROUP
ENDBAND
part
PRIMARY
ATTACHMENT
BOOKBLOCK
ENDBAND
GROUP
part
part
part
part
part
part
ADDITIONAL
INFORMATION
ADDITIONAL
INFORMATION
KNOTTED
FRAYED
ENDBAND
part
part
PRIMARY
TYPE
SECONDARY
TYPE
CROWNING
ENDBAND
CORE
CROWNING
ENDBAND
CORE
part
part
part
part
part
part
part
part
part
part
part
part
SPAN
1
TWIESTED
SPAN
KNOTTED
DOUBLE
LENGTH
SEWN
SINGLE
LENGTH
SPAN
2
CLOSED
LOOP
KNOTTED
SINGLE
LENGTH
PLAITED
MULTIPLE
SPAN
TWISTED
CLOSED
LOOP
HITCHED
DOUBLE
LENGTH
PLIED

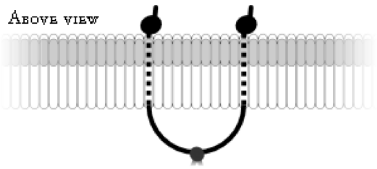
BOOKMARKS (COMPOUND) - I

PRIMARY: SPAN I
SECONDARY: KNOTTED DOUBLE LENGTH

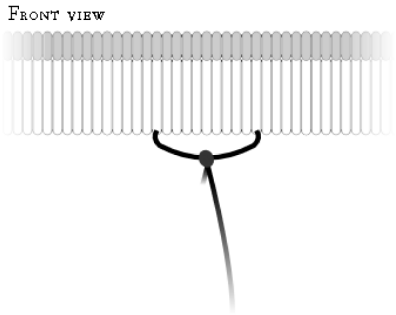




KNOTTED



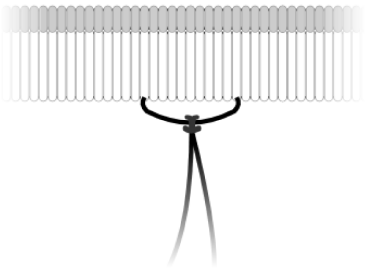
PRIMARY: SPAN I
SECONDARY: KNOTTED SINGLE LENGTH



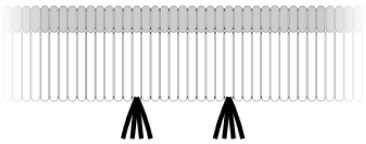
BOOKMARKS (COMPOUND) - 2

PRIMARY: SPAN 1
SECONDARY: HITCHED DOUBLE LENGTH

Front view

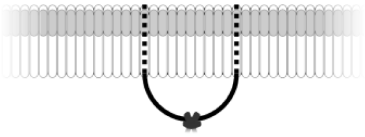


Spine view



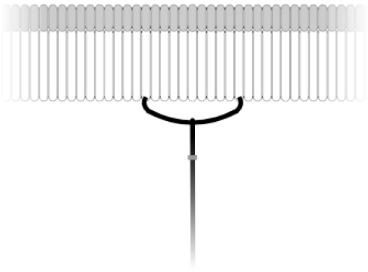
FRAYED

Above view

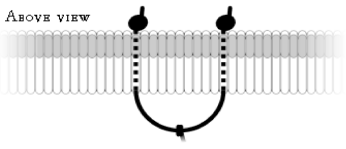


PRIMARY: SPAN 1
SECONDARY: SEWING SINGLE LENGTH

Front view

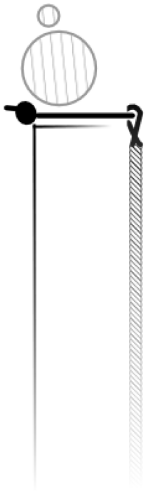


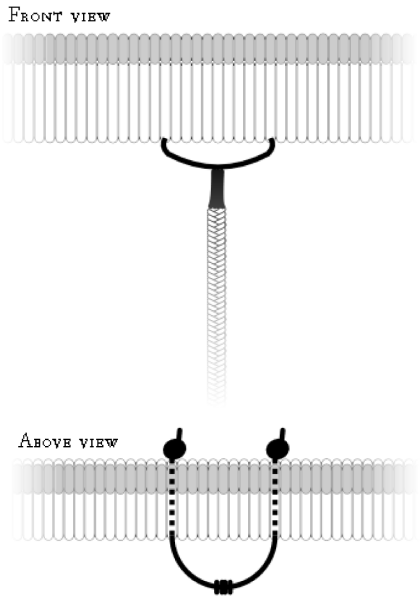
Above view



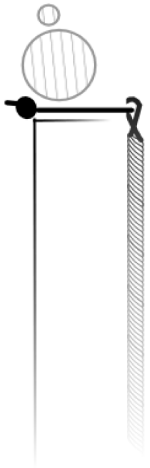
BOOKMARKS (COMPOUND) - 3

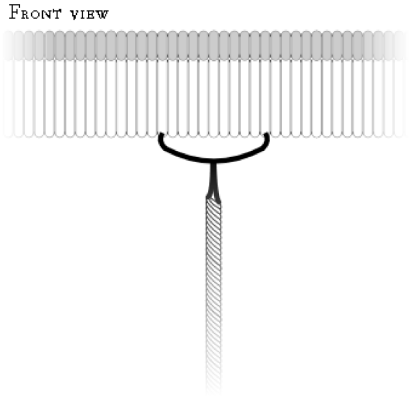
PRIMARY: SPAN 1
SECONDARY: PLAITED





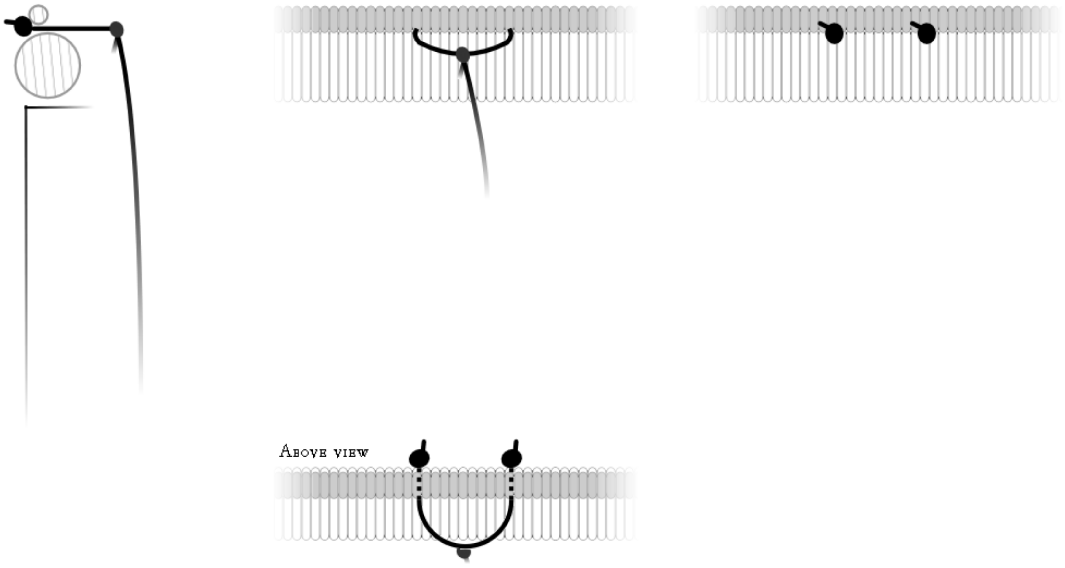
PRIMARY: SPAN 1
SECONDARY: PLIED



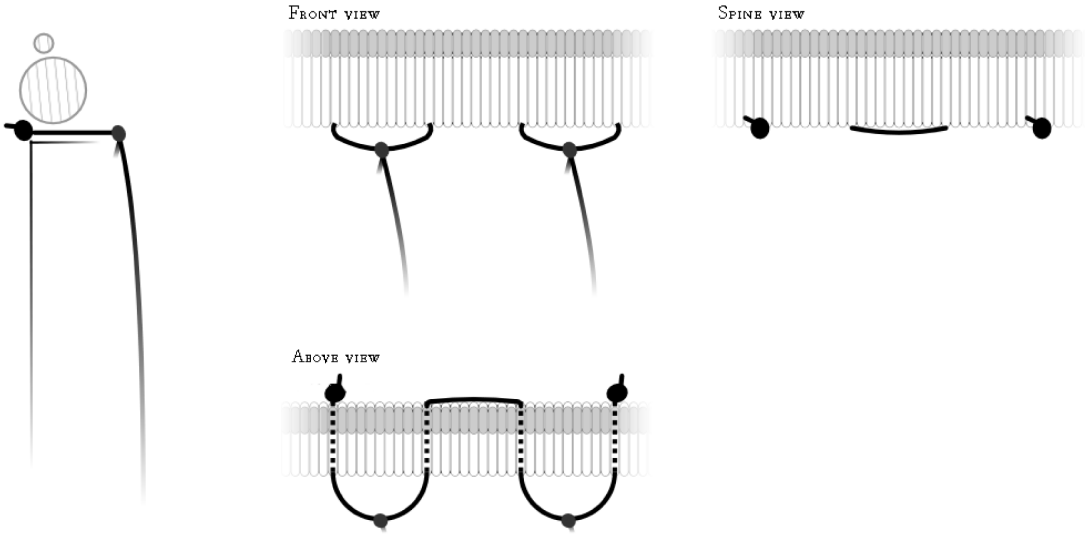


BOOKMARKS (COMPOUND) - 4

PRIMARY: SPAN 2
SECONDARY: KNOTTED SINGLE LENGTH

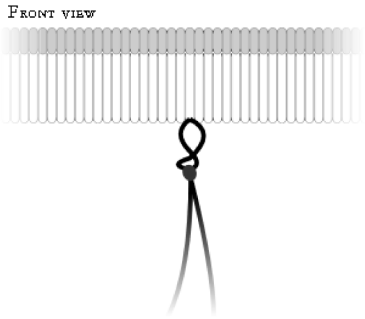


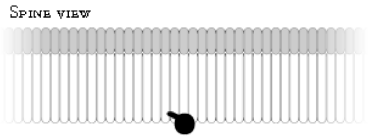
PRIMARY: MULTIPLE SPAN
SECONDARY: KNOTTED SINGLE LENGTH

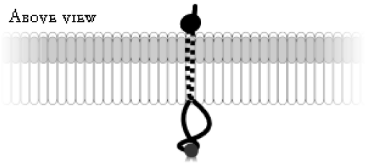


BOOKMARKS (COMPOUND) - 5

PRIMARY: TWISTED CLOSED LOOP
SECONDARY: KNOTTED DOUBLE LENGTH

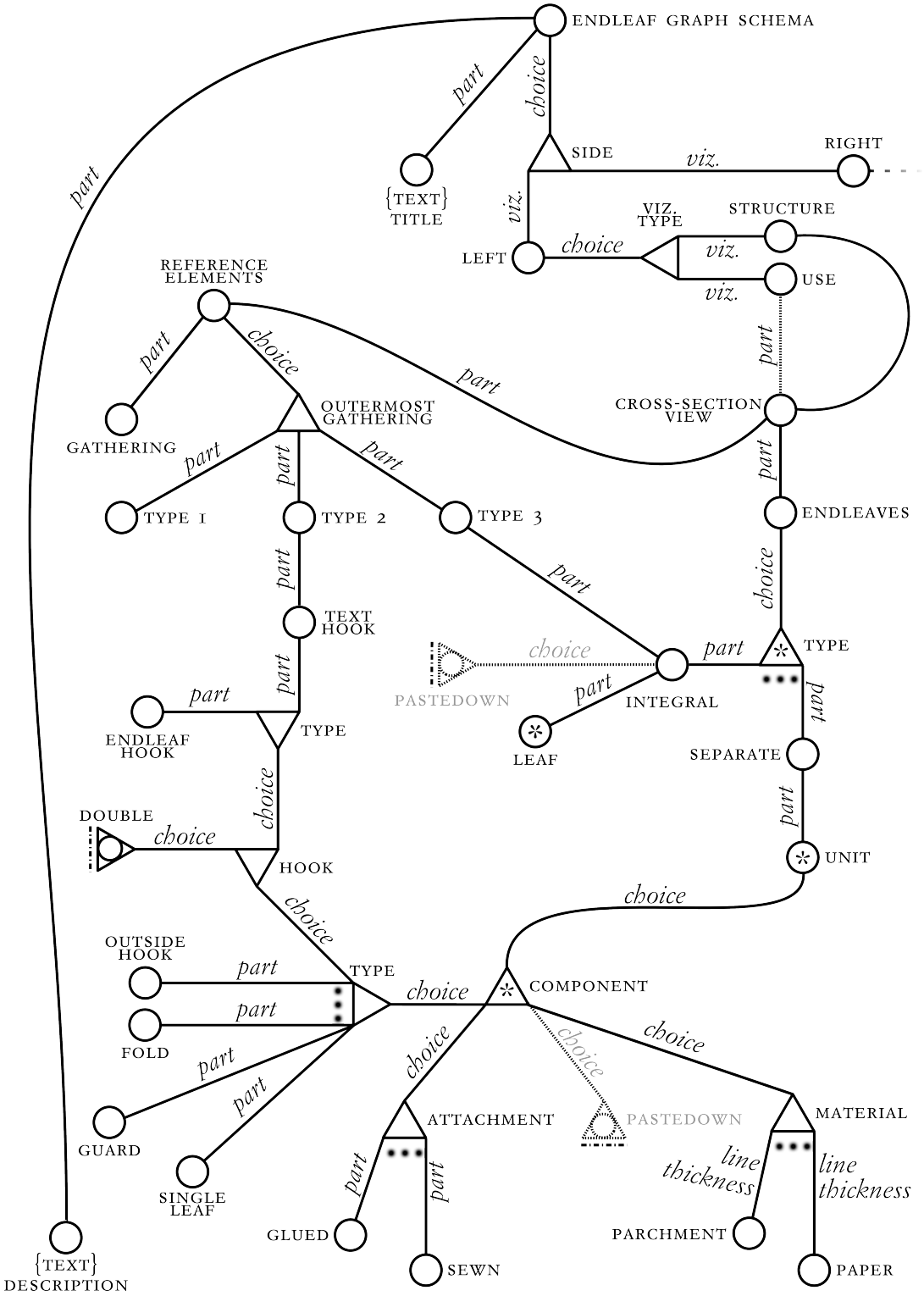




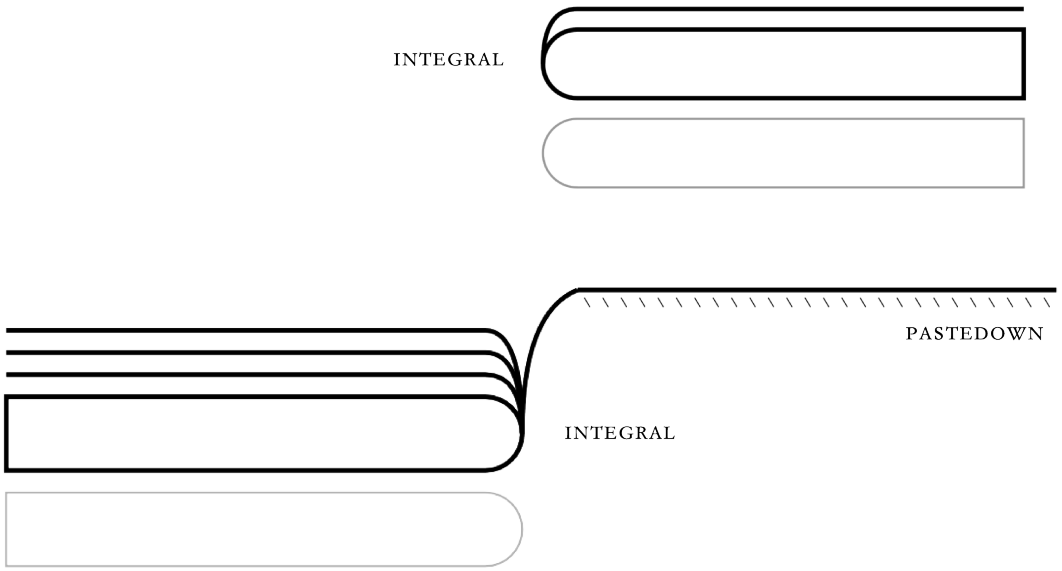


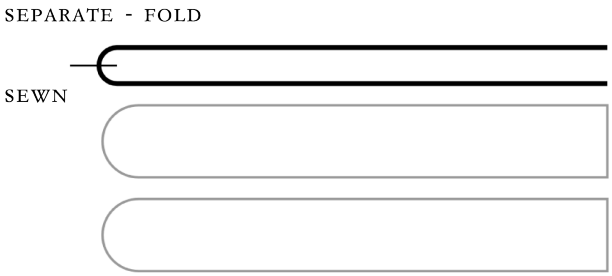
Endleaves

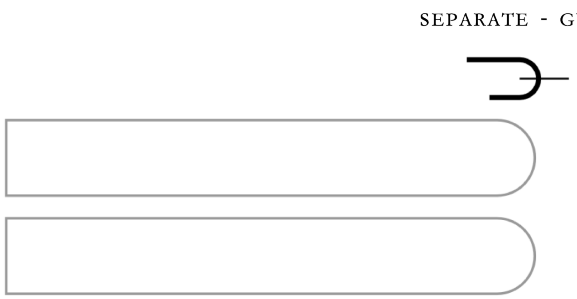
Endleaves are all the groups of leaves found at each side of a bookblock and are intended to give protection to the text leaves. They come in two basic types: those added by the binder before the book is sewn (separate endleaves) and blank leaves at the front and/or the back of the textblock (integral endleaves) which are used as endleaves, and may or may not also be pasted to the boards as pastedowns, even though they form part of text gatherings. Separate endleaves and integral endleaves can be combined at either the left or right ends of the textblock, and the endleaves at left and right can be made up differently. See §7.1.


ENDLEAF GRAPH SCHEMA
part
choice
{TEXT}
TITLE
SIDE
viz.
RIGHT
viz.
LEFT
choice
VIZ. TYPE
STRUCTURE
viz.
USE
viz.
REFERENCE ELEMENTS
part
choice
part
part
GATHERING
OUTERMOST GATHERING
CROSS-SECTION VIEW
part
part
part
part
part
TYPE 1
TYPE 2
TYPE 3
ENDLEAVES
part
part
choice
TEXT HOOK
part
choice
part
TYPE
PASTEDOWN
part
INTEGRAL
part
ENDLEAF HOOK
part
TYPE
LEAF
SEPARATE
part
DOUBLE
choice
choice
HOOK
UNIT
choice
choice
OUTSIDE HOOK
part
TYPE
choice
COMPONENT
part
FOLD
part
choice
choice
choice
part
part
ATTACHMENT
choice
PASTEDOWN
MATERIAL
GUARD
SINGLE LEAF
part
part
line thickness
line thickness
GLUED
SEWN
PARCHMENT
PAPER
{TEXT}
DESCRIPTION

ENDLEAVES - I


INTEGRAL
PASTEDOWN
INTEGRAL

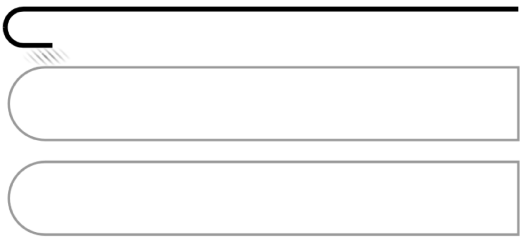

SEPARATE - FOLD
SEWN


SEPARATE - GUARD

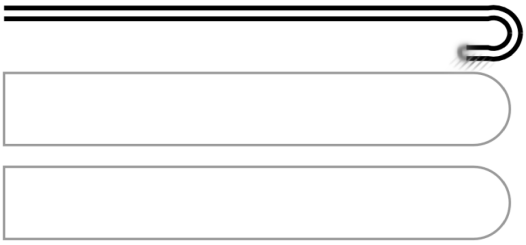
ENDLEAVES - 2

SEPARATE - ENDLEAF HOOK (NOT DOUBLE)

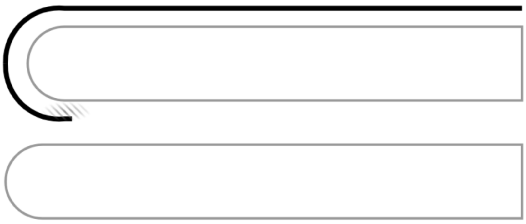
GLUED



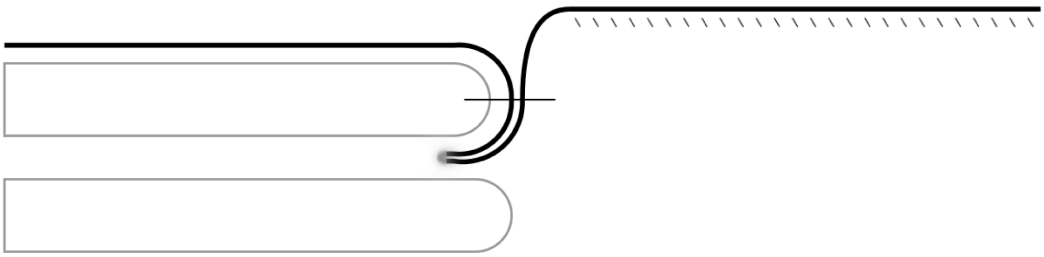
SEPARATE - ENDLEAF HOOK (DOUBLE)



SEPARATE - TEXT HOOK (NOT DOUBLE)

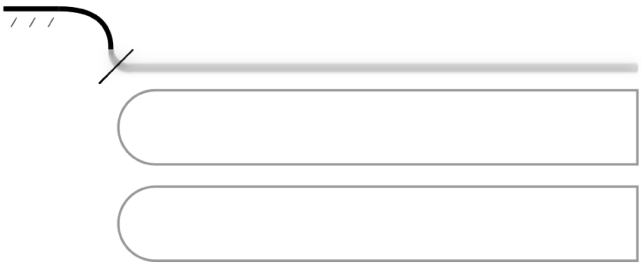


SEPARATE - TEXT HOOK (DOUBLE)

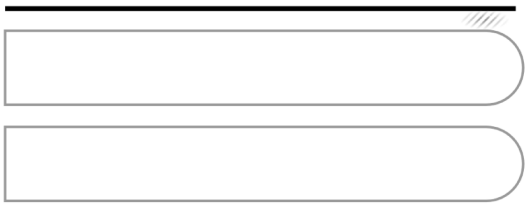


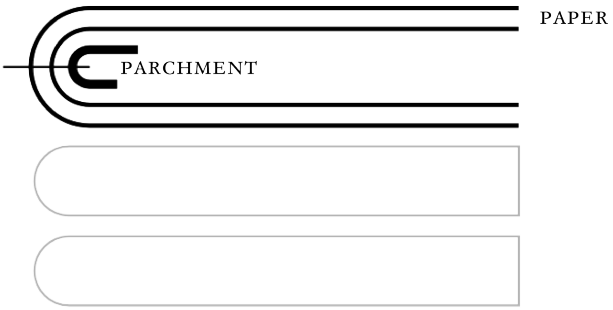
ENDLEAVES - 3

SEPARATE - OUTSIDE HOOK



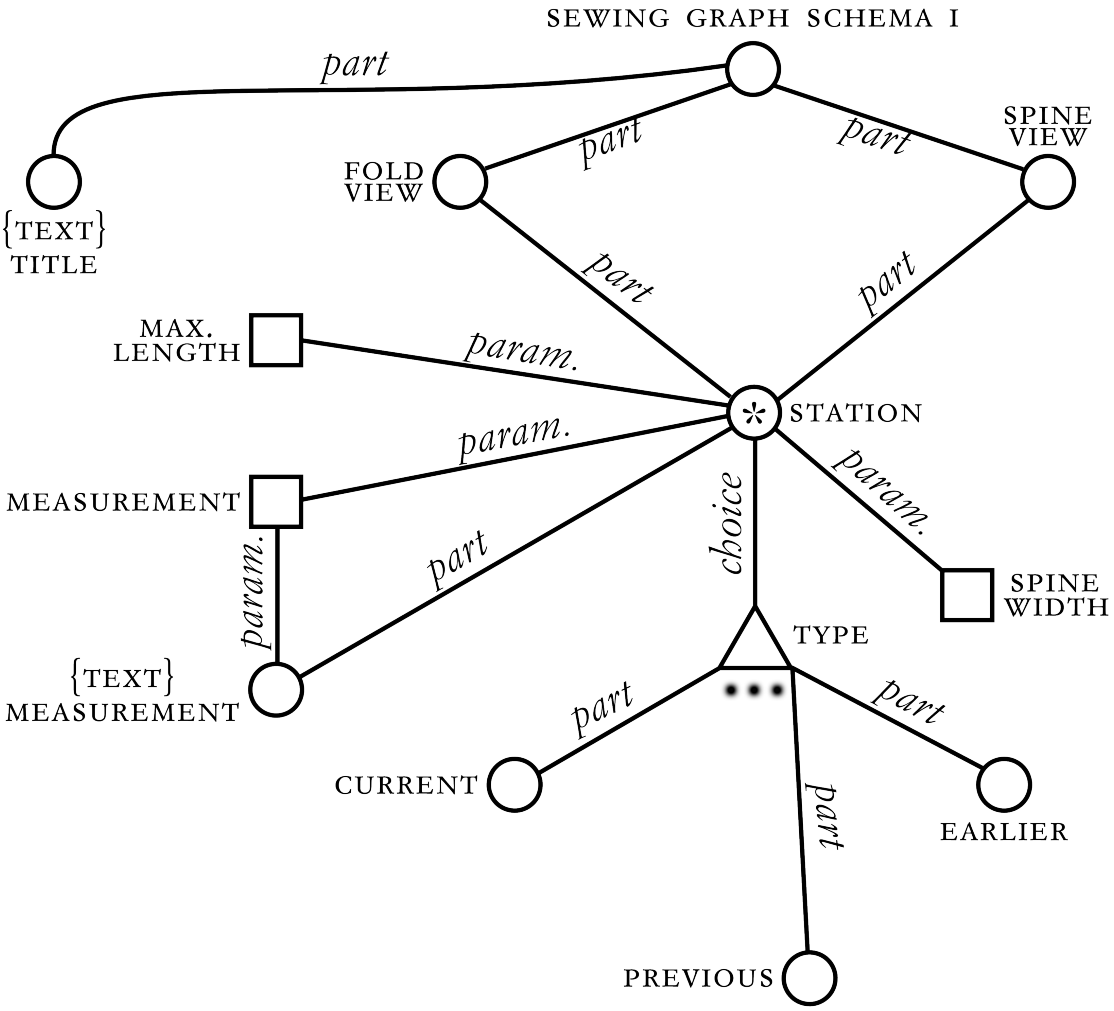
SEPARATE - SINGLE LEAF





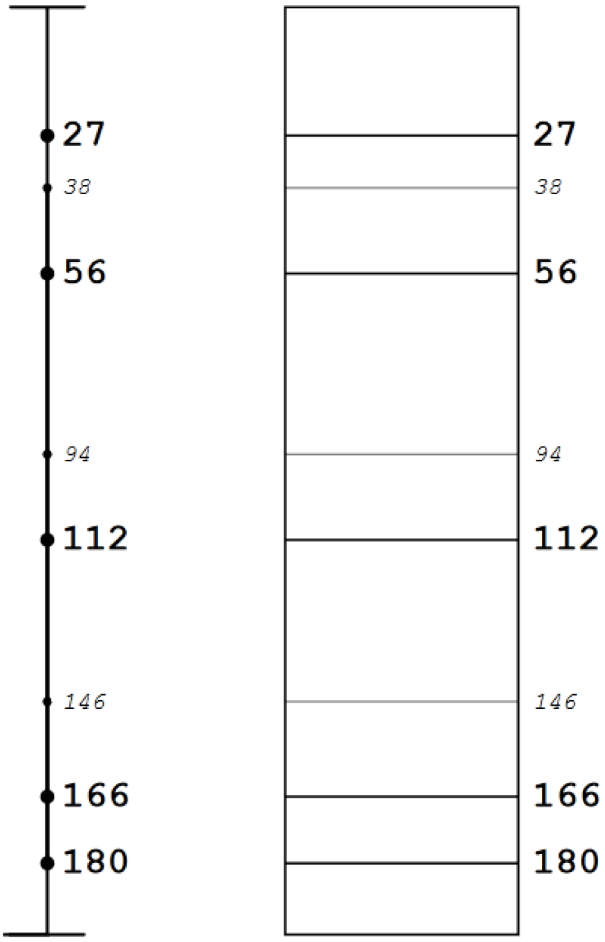
Sewing

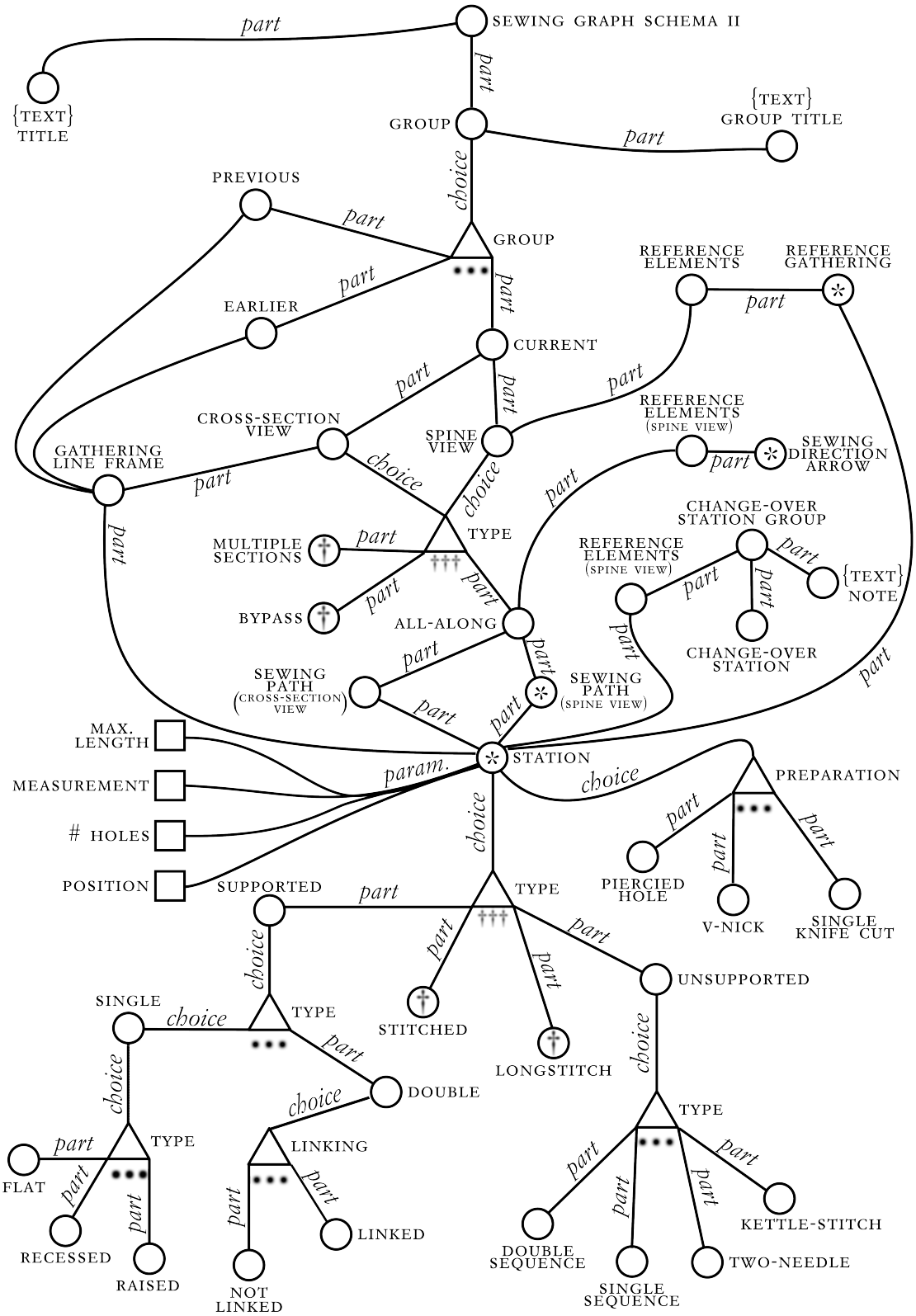
Sewing indicates the process through which the sections or leaves of a book are secured by means of thread, thus forming a consecutive and permanent unit. The basic description element of each sewing is the sewing station, i.e. the individual structural point on the spine-fold used in the creation of the book structure. See §7.2.2.


SEWING GRAPH SCHEMA I
part
SPINE VIEW
part
part
FOLD VIEW
{TEXT} TITLE
part
part
MAX. LENGTH
param.
STATION
param.
MEASUREMENT
param.
choice
param.
part
SPINE WIDTH
TYPE
{TEXT} MEASUREMENT
part
part
CURRENT
part
EARLIER
PREVIOUS

SEWING - I

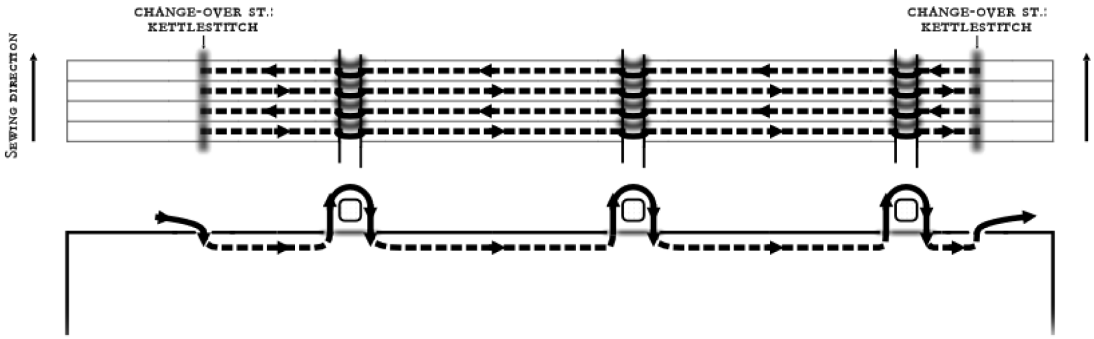
(MEASUREMENTS IN MM)



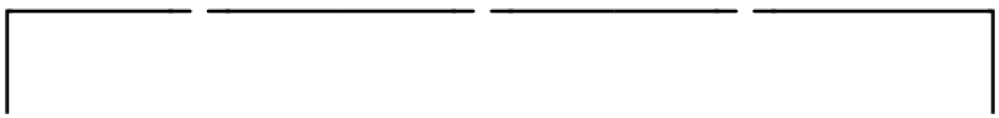

SEWING GRAPH SCHEMA II
{TEXT} TITLE
GROUP
{TEXT} GROUP TITLE
PREVIOUS
GROUP
EARLIER
CURRENT
REFERENCE ELEMENTS
REFERENCE GATHERING
CROSS-SECTION VIEW
SPINE VIEW
REFERENCE ELEMENTS (SPINE VIEW)
SEWING DIRECTION ARROW
GATHERING LINE FRAME
CHANGE-OVER STATION GROUP
MULTIPLE SECTIONS
TYPE
REFERENCE ELEMENTS (SPINE VIEW)
{TEXT} NOTE
BYPASS
ALL-ALONG
CHANGE-OVER STATION
SEWING PATH (CROSS-SECTION VIEW)
SEWING PATH (SPINE VIEW)
MAX. LENGTH
MEASUREMENT
HOLES
POSITION
STATION
PREPARATION
PIERCIED HOLE
V-NICK
SINGLE KNIFE CUT
SUPPORTED
TYPE
UNSUPPORTED
SINGLE
TYPE
STITCHED
LONGSTITCH
DOUBLE
FLAT
TYPE
LINKING
TYPE
KETTLE-STITCH
RECESSED
RAISED
LINKED
NOT LINKED
DOUBLE SEQUENCE
SINGLE SEQUENCE
TWO-NEEDLE
part
choice
param.

SEWING - 2

Current sewing (type: allAlong)

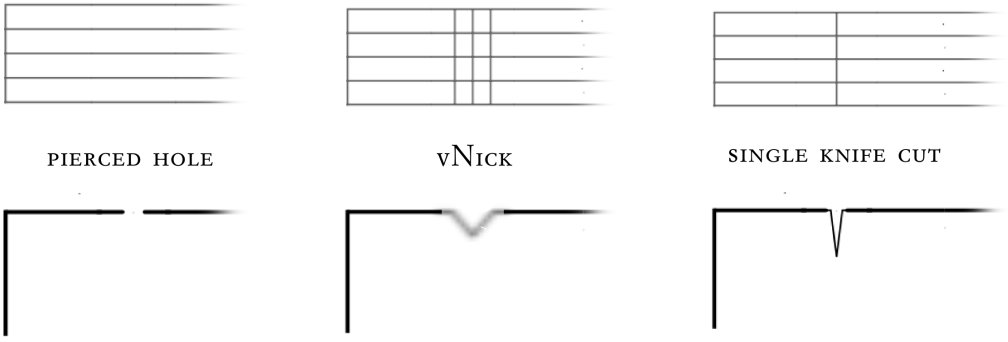


Previous sewing



SEWING - 3

PREPARATION

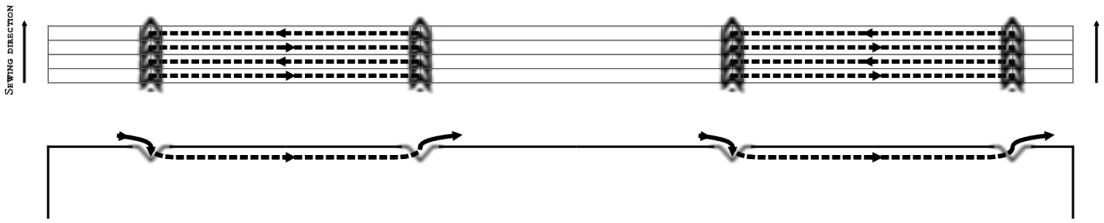


PIERCED HOLE

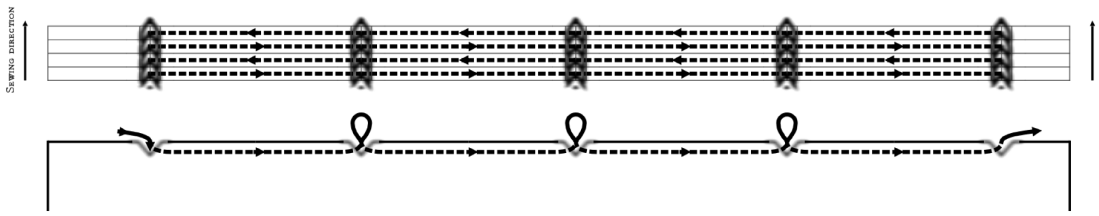
VNICK

SINGLE KNIFE CUT

UNSUPPORTED SEWING - TWO NEEDLE

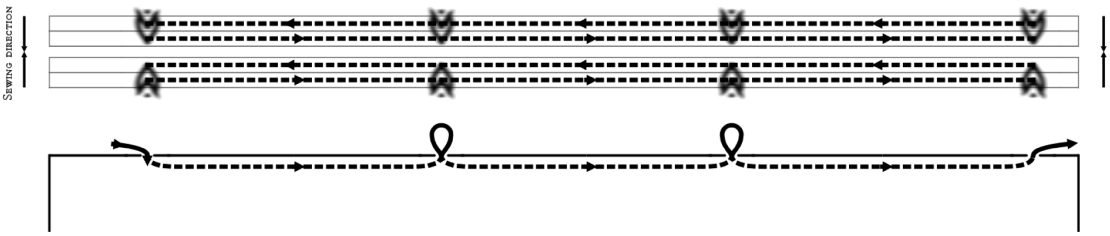


UNSUPPORTED SEWING - SINGLE SEQUENCE



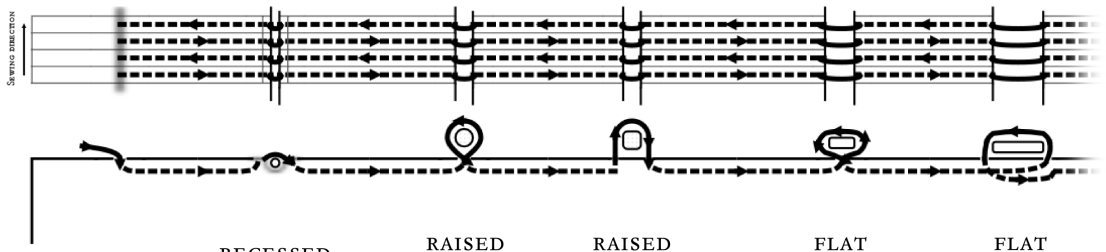
SEWING - 4

UNSUPPORTED SEWING - DOUBLE SEQUENCE



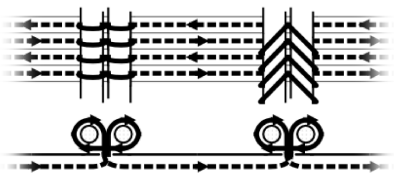
SUPPORTED SEWING - SINGLE

CHANGE-OVER ST.:
KETTLESTITCH



RECESSED | RAISED 1 HOLE | RAISED 2 HOLES | FLAT 1 HOLE | FLAT 2 HOLES

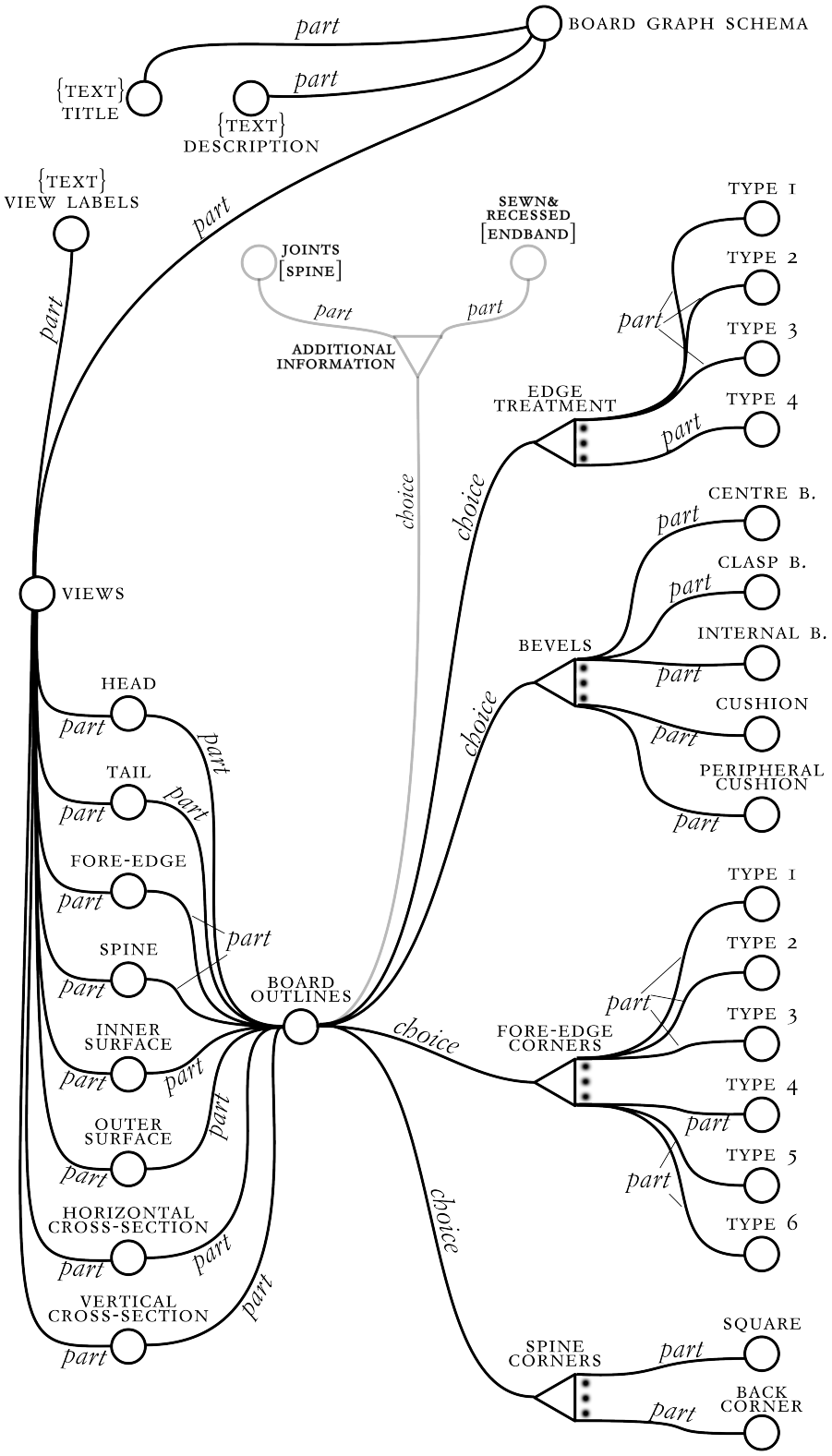
SUPPORTED SEWING - DOUBLE



NOT LINKED LINKED

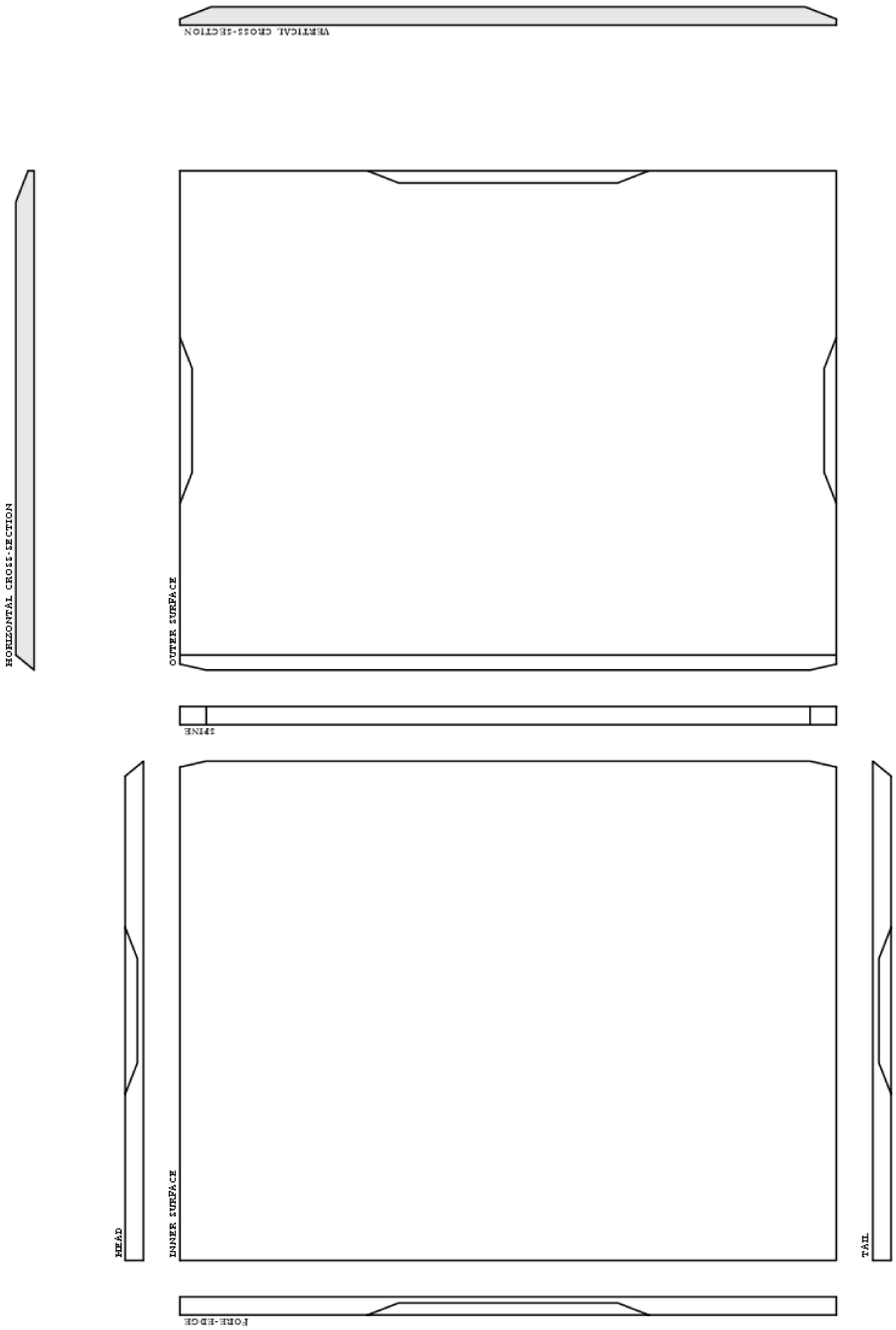
Boards

The wood, pasted paper, single- or multiple-ply sheets, or other material, used for the covers of bound books to protect the leaves. Boards are usually used in pairs and they never extend around the spine of the book. See §7.2.3.


BOARD GRAPH SCHEMA
part
part
part
{TEXT}
TITLE
{TEXT}
DESCRIPTION
{TEXT}
VIEW LABELS
part
part
JOINTS
[SPINE]
SEWN&
RECESSED
[ENDBAND]
part
part
ADDITIONAL
INFORMATION
choice
choice
choice
choice
choice
VIEWS
EDGE
TREATMENT
TYPE 1
TYPE 2
TYPE 3
TYPE 4
part
part
part
part
BEVELS
CENTRE B.
CLASP B.
INTERNAL B.
CUSHION
PERIPHERAL
CUSHION
part
part
part
part
part
HEAD
TAIL
FORE-EDGE
SPINE
INNER
SURFACE
OUTER
SURFACE
HORIZONTAL
CROSS-SECTION
VERTICAL
CROSS-SECTION
part
part
part
part
part
part
part
part
part
part
part
part
BOARD
OUTLINES
FORE-EDGE
CORNERS
TYPE 1
TYPE 2
TYPE 3
TYPE 4
TYPE 5
TYPE 6
part
part
part
part
part
SPINE
CORNERS
SQUARE
BACK
CORNER
part
part

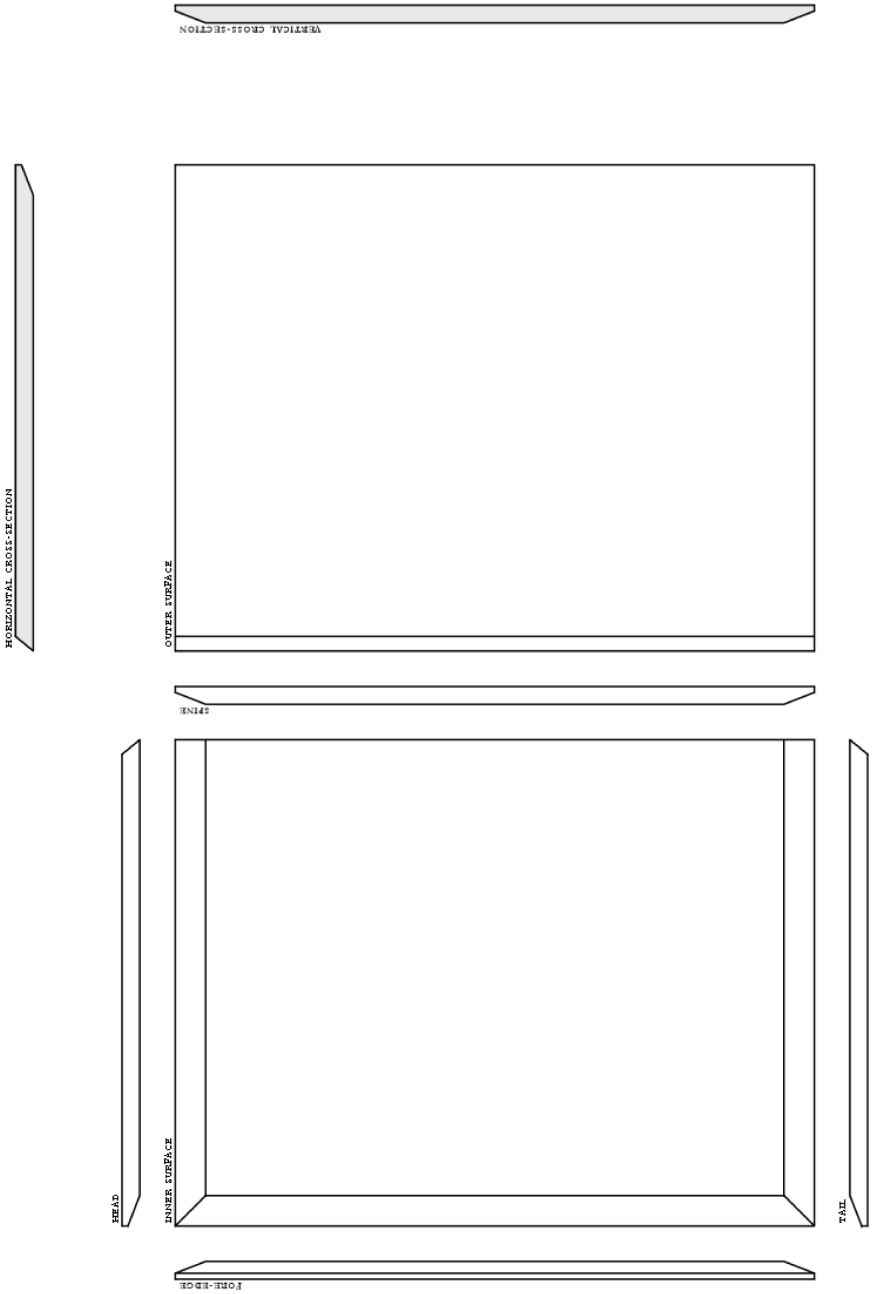
BOARDS - I

CENTRE BEVELS



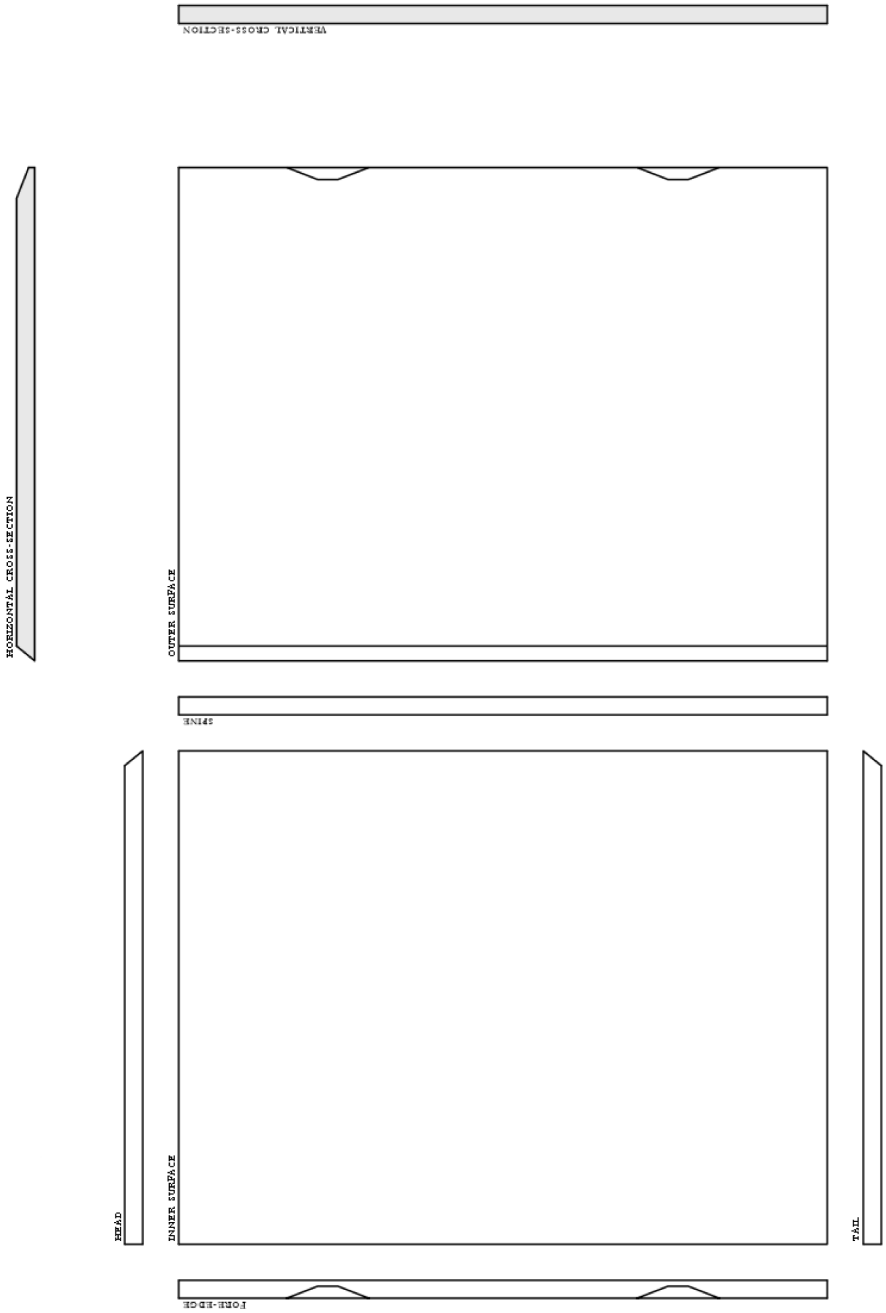
BOARDS - 2

INTERNAL BEVELS



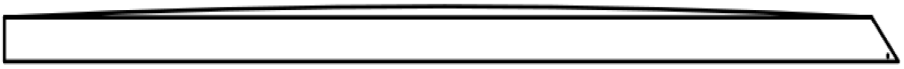
BOARDS - 3

CLASP BEVELS

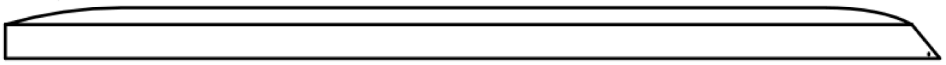


BOARDS - 4

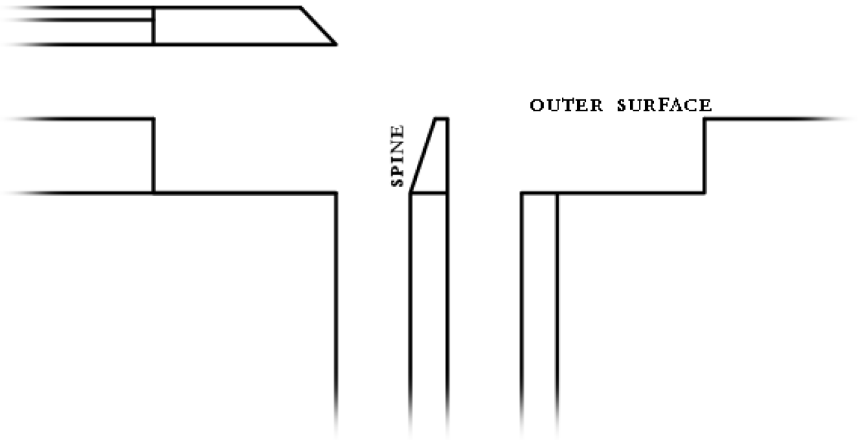
CUSHION



PERIPHERAL CUSHION



(ENDBAND CORE)
SEWN & RECESSED



BOARDS - 5

EDGE TREATMENT

TYPE 1

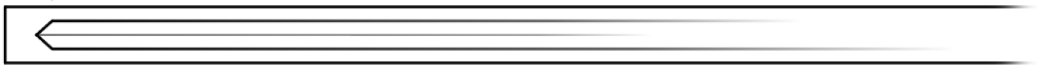
TYPE 2

TYPE 3

TYPE 4

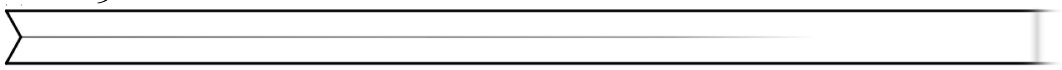
FORE-EDGE CORNER

TYPE 1

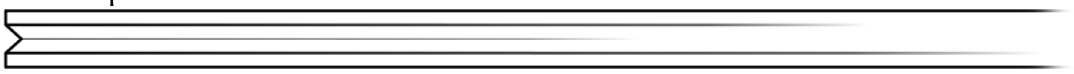


TYPE 2

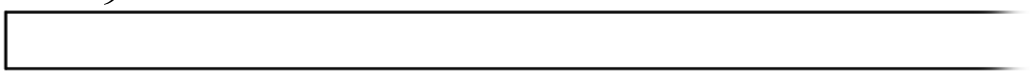
TYPE 3



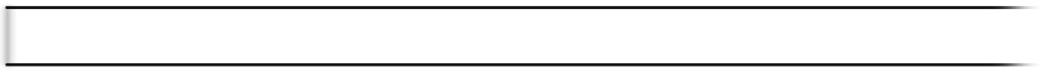
TYPE 4



TYPE 5

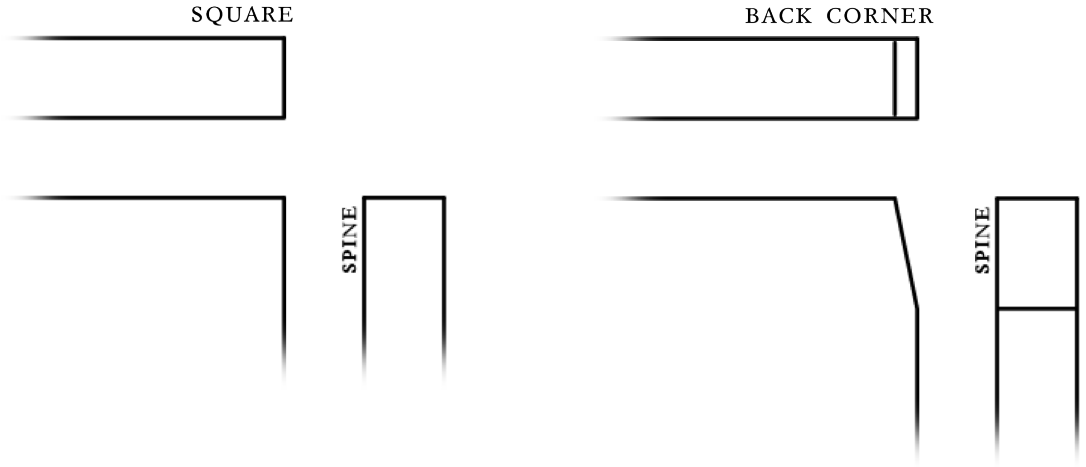


TYPE 6

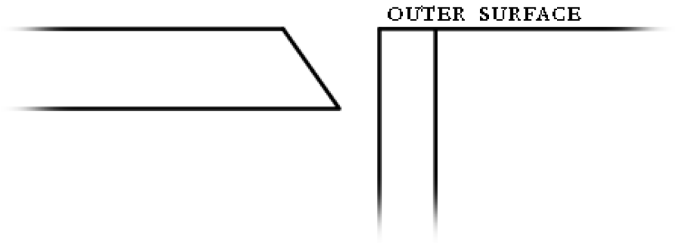


BOARDS - 6

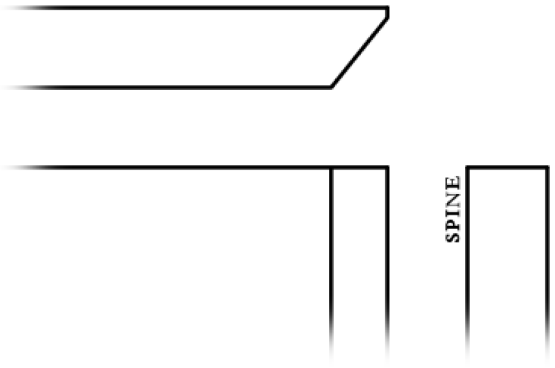
SPINE CORNER



JOINT SHAPE
ANGLED

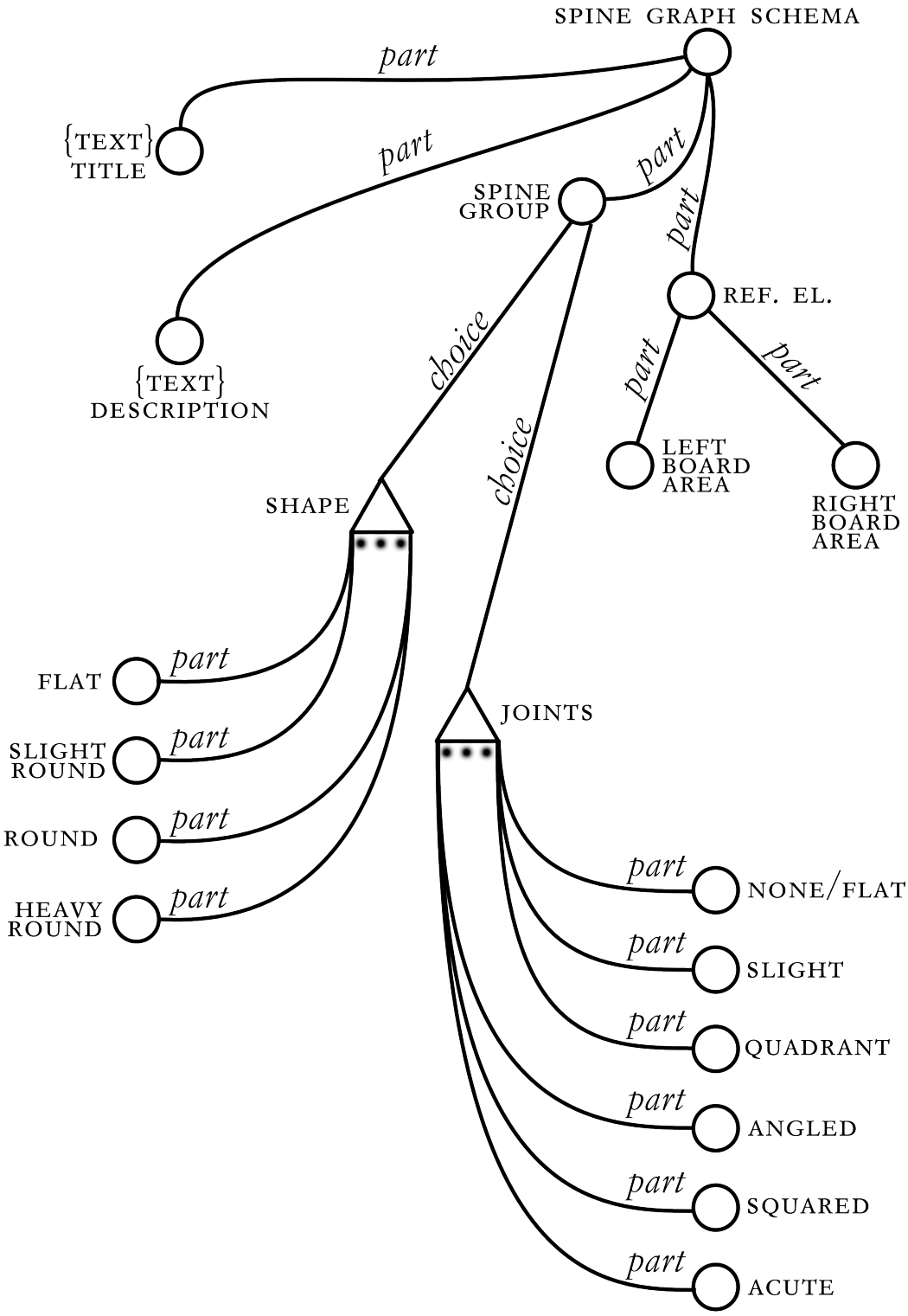


JOINT SHAPE
ACUTE

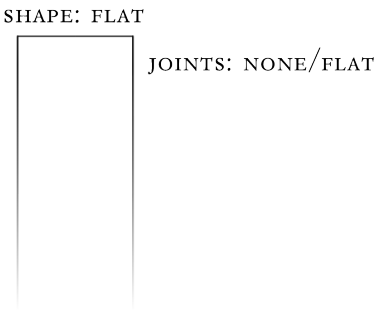


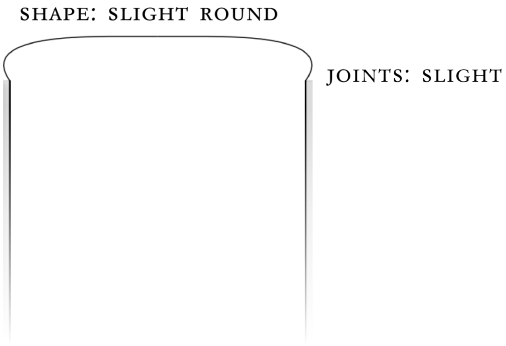
Spine shape

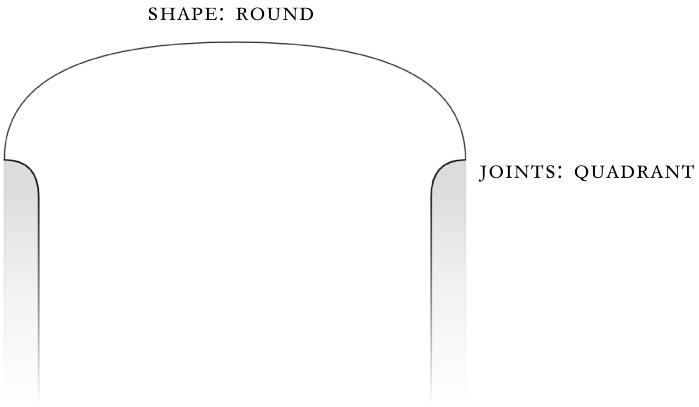
The schema describes the shape of the spine of the bookblock in an approximate and categorized manner distinguishing between its degree of roundness (flat, slight round, round, heavy round) or the shape of its joints (flat, slight, quadrant, angled, square, acute). This allows inputting a record of the shape of the spine in the database, and to then be able, for example, to search for different values of roundness for bookblock spines. See §7.2.4.


SPINE GRAPH SCHEMA
part
{TEXT}
TITLE
part
part
part
SPINE
GROUP
REF. EL.
{TEXT}
DESCRIPTION
choice
choice
part
part
LEFT
BOARD
AREA
RIGHT
BOARD
AREA
SHAPE
JOINTS
FLAT
SLIGHT
ROUND
ROUND
HEAVY
ROUND
part
part
part
part
part
part
part
part
part
part
NONE/FLAT
SLIGHT
QUADRANT
ANGLED
SQUARED
ACUTE

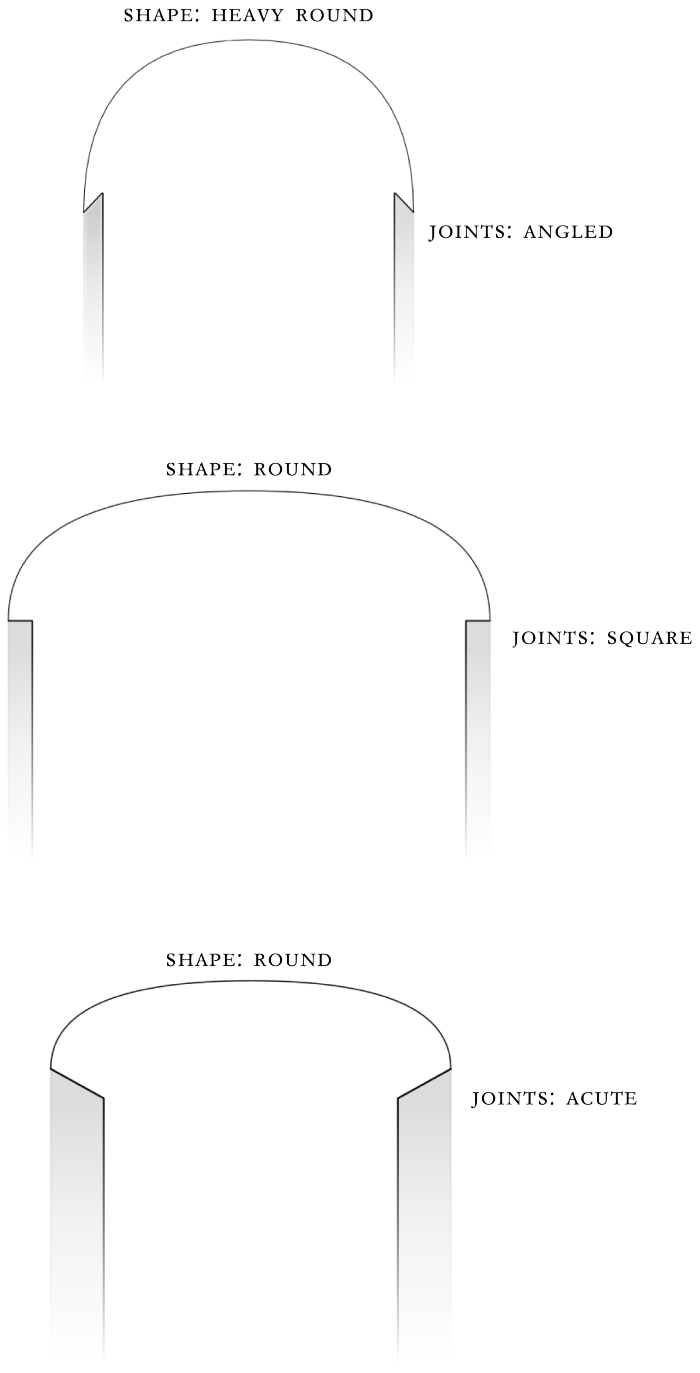
SPINE: SHAPE & JOINTS - I





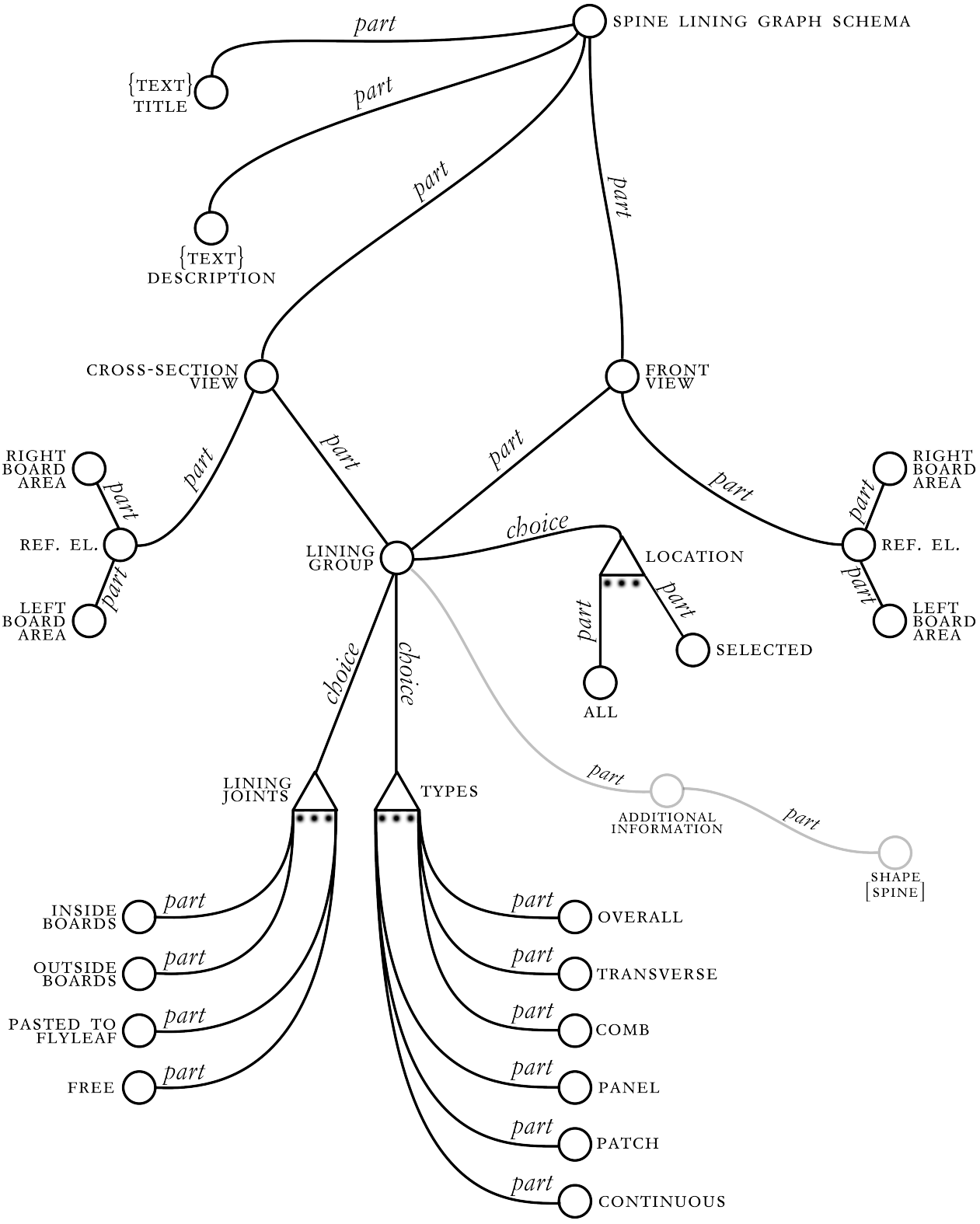


SPINE: SHAPE & JOINTS - 2



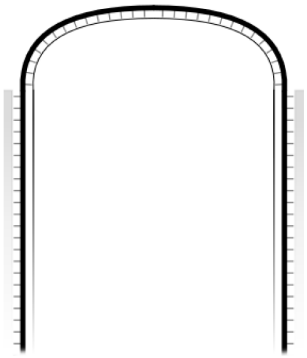
Spine lining

The spine lining is composed of one or a series of pieces of sheet material placed on the spine of the assembled bookblock as reinforcement, and either adhered to it or held in place without adhesive. See §7.2.4.

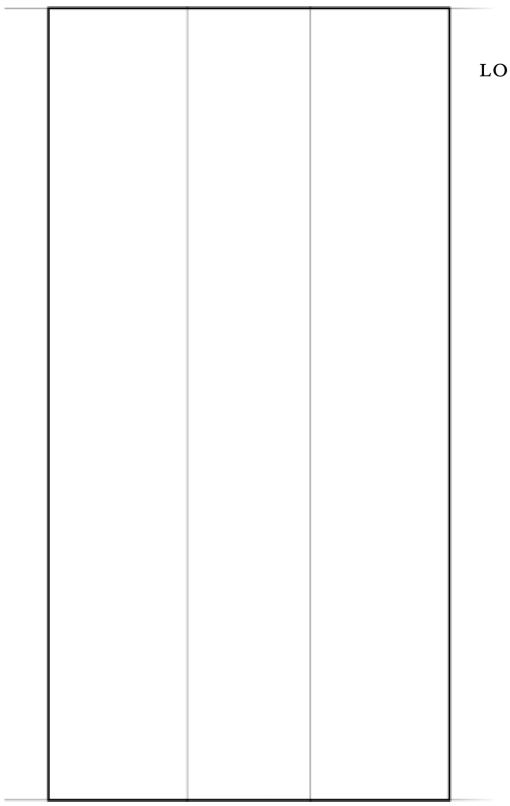

SPINE LINING GRAPH SCHEMA
{TEXT} TITLE
{TEXT} DESCRIPTION
CROSS-SECTION VIEW
FRONT VIEW
RIGHT BOARD AREA
REF. EL.
LEFT BOARD AREA
LINING GROUP
LOCATION
ALL
SELECTED
RIGHT BOARD AREA
REF. EL.
LEFT BOARD AREA
LINING JOINTS
TYPES
INSIDE BOARDS
OUTSIDE BOARDS
PASTED TO FLYLEAF
FREE
OVERALL
TRANSVERSE
COMB
PANEL
PATCH
CONTINUOUS
ADDITIONAL INFORMATION
SHAPE [SPINE]
part
choice

SPINE LINING - I

TYPE: OVERALL



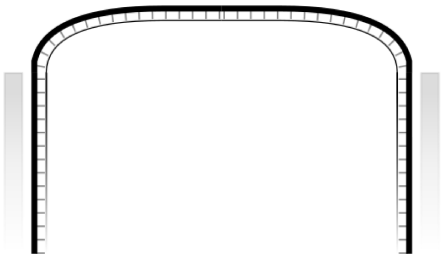
PASTED INSIDE BOARDS



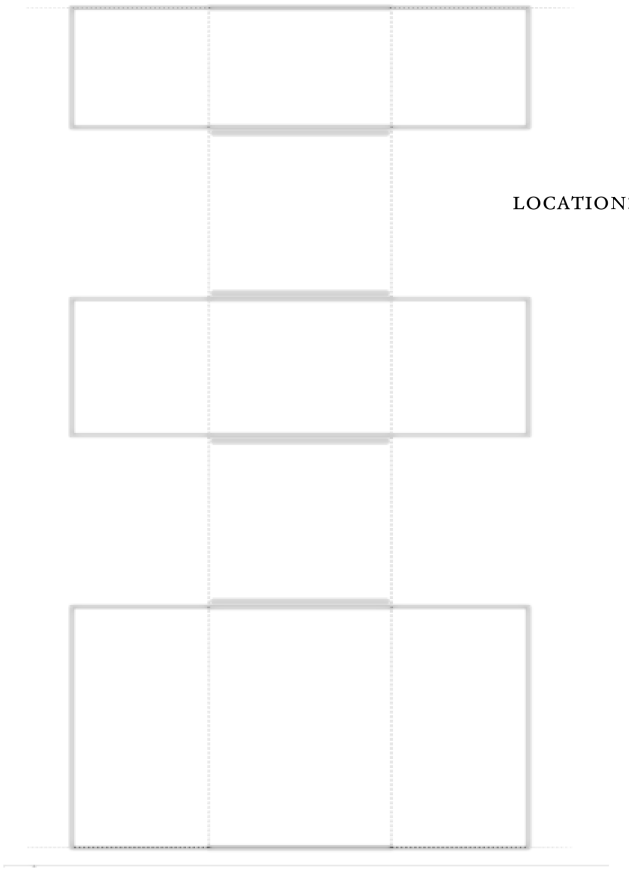
LOCATION: ALL

SPINE LINING - 2

TYPE: TRANSVERSE



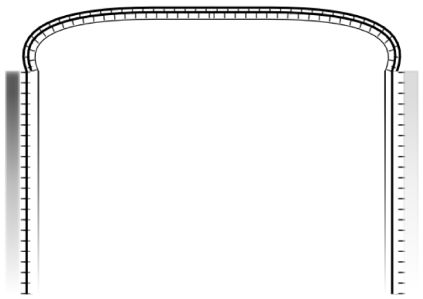
PASTED TO FLYLEAF



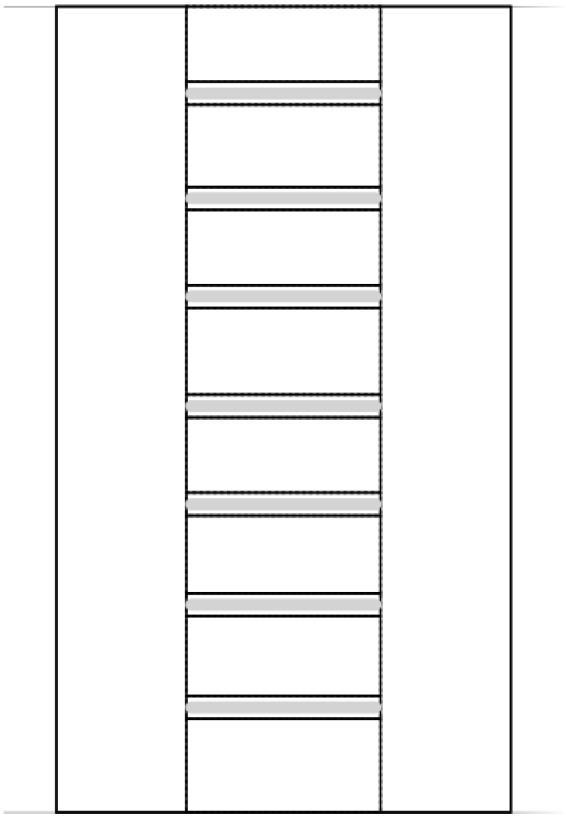
LOCATION: SELECTED

SPINE LINING - 3

TYPE: COMB



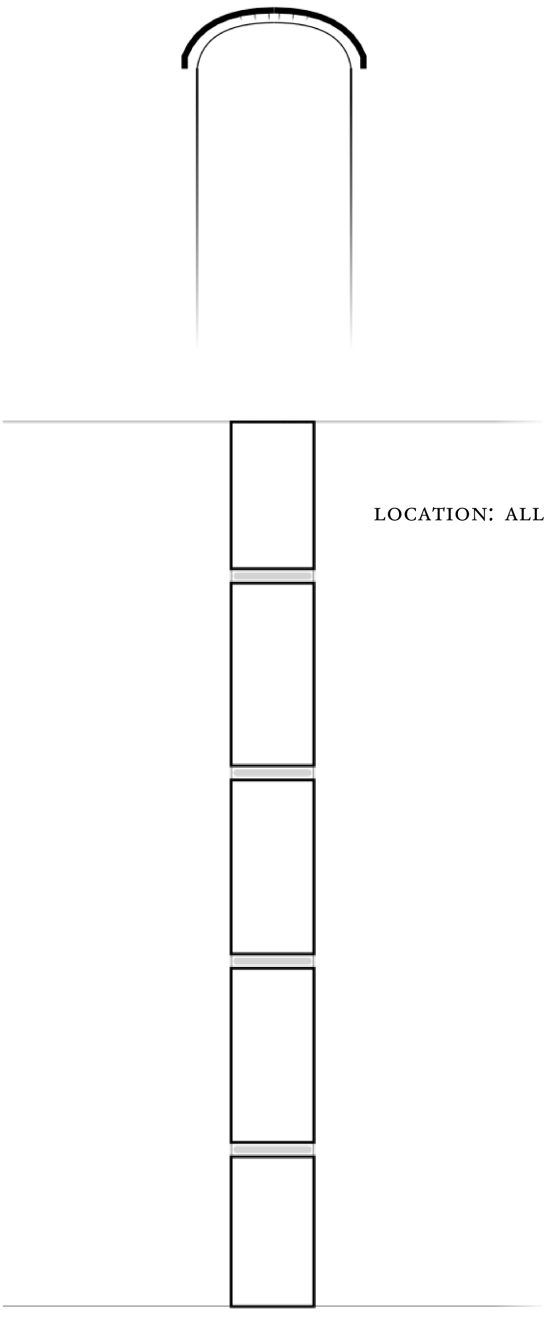
PASTED INSIDE BOARDS



LOCATION: ALL

SPINE LINING - 4

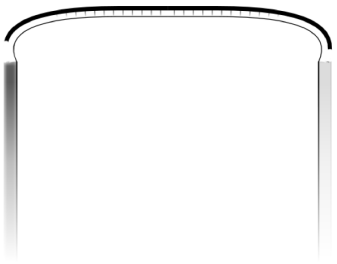
TYPE: PANEL



LOCATION: ALL

SPINE LINING - 4

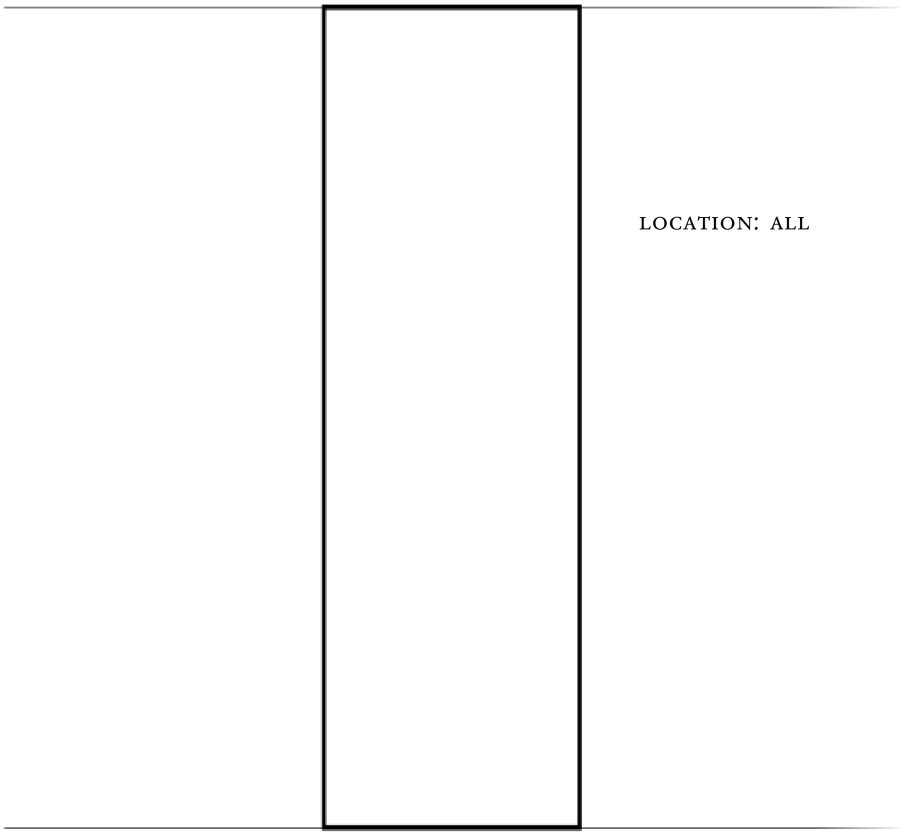
TYPE: PATCH



LOCATION: SELECTED

SPINE LINING - 6

TYPE: CONTINUOUS

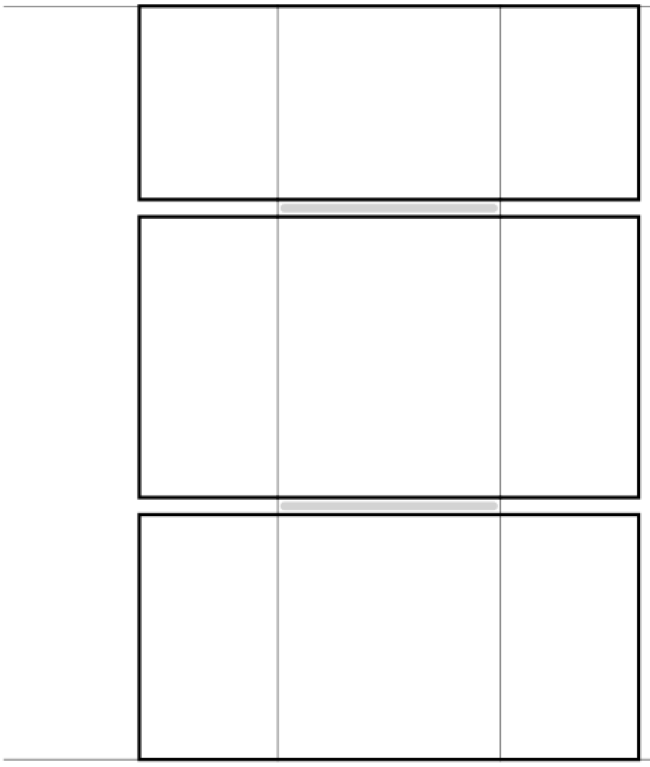


LOCATION: ALL

SPINE LINING - 7

TYPE: TRANSVERSE

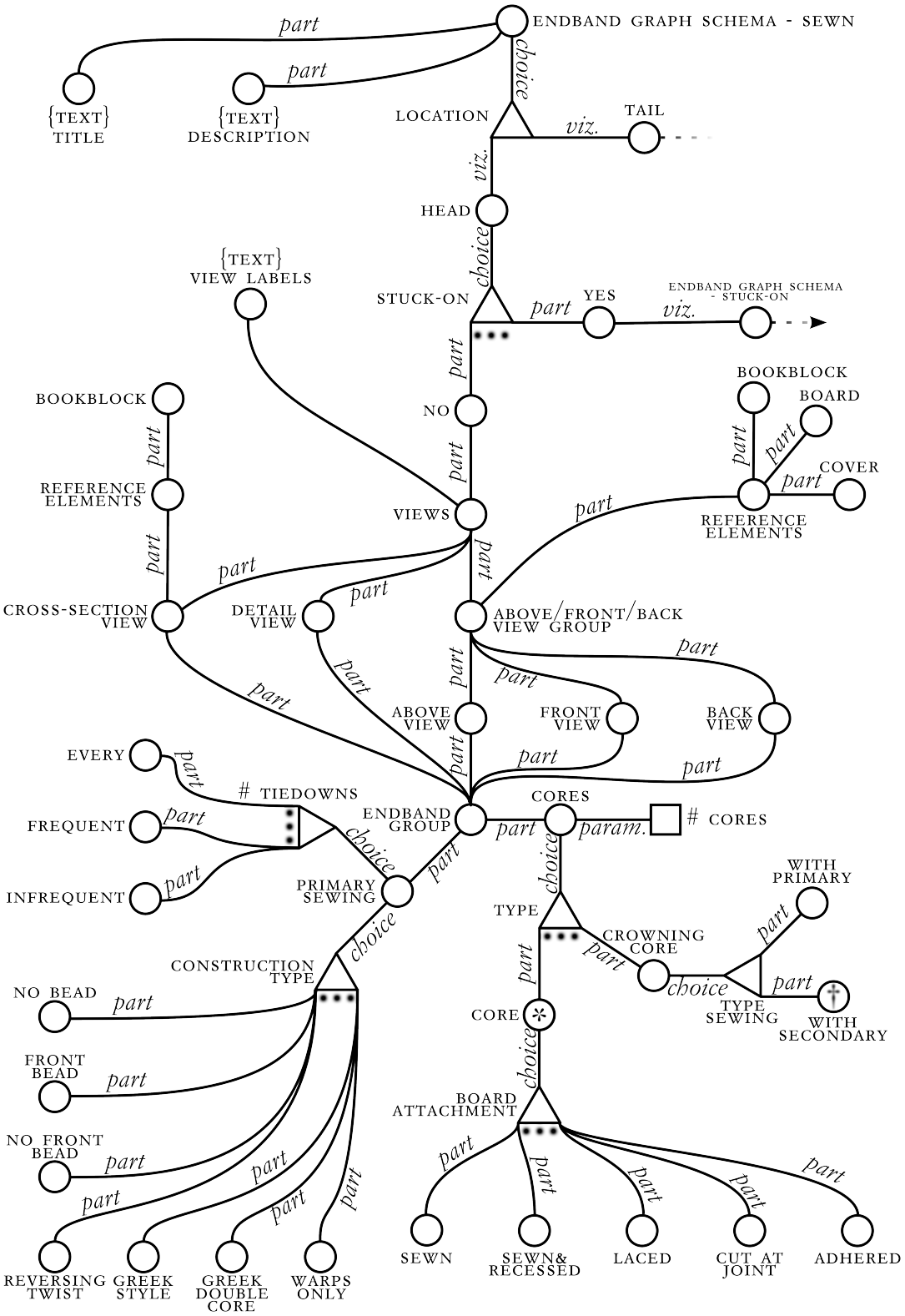
FREE



LOCATION: ALL

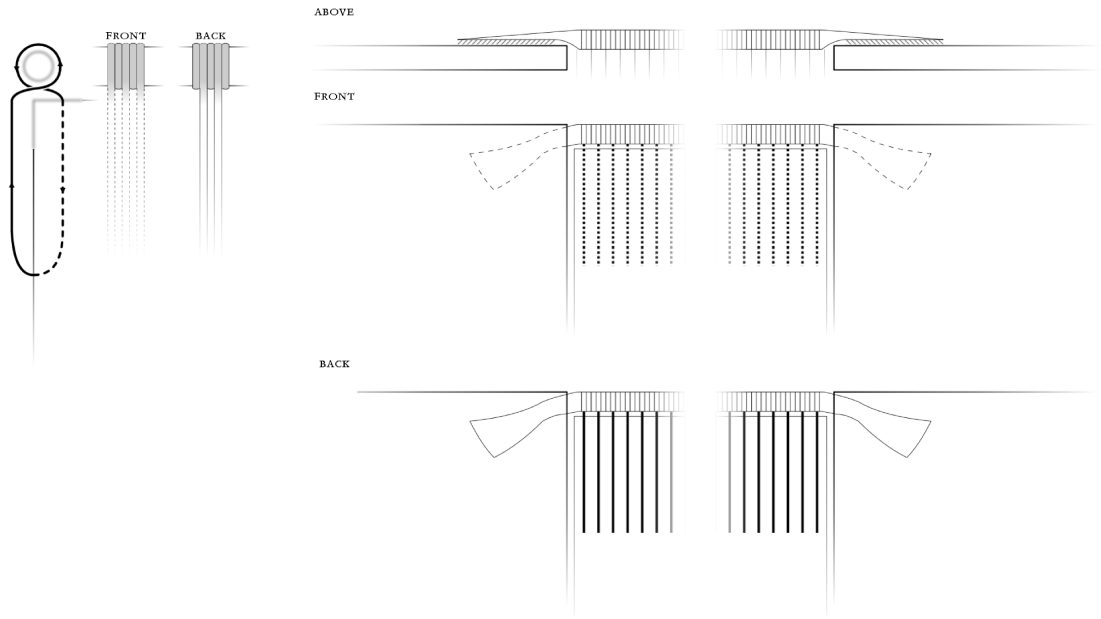
Endbands

Endbands are cores sewn or pasted across the head and tail of a bookblock spine. When sewn, these are attached to the bookblock by a primary sewing through the fold of the gatherings which loops around the core. When pasted, these are essentially transverse spine lining attached to the head and tail panels. See §7.2.5.


ENDBAND GRAPH SCHEMA - SEWN
{TEXT} TITLE
{TEXT} DESCRIPTION
LOCATION
TAIL
HEAD
{TEXT} VIEW LABELS
STUCK-ON
YES
ENDBAND GRAPH SCHEMA - STUCK-ON
NO
BOOKBLOCK
REFERENCE ELEMENTS
VIEWS
BOOKBLOCK
BOARD
COVER
REFERENCE ELEMENTS
CROSS-SECTION VIEW
DETAIL VIEW
ABOVE/FRONT/BACK VIEW GROUP
ABOVE VIEW
FRONT VIEW
BACK VIEW
EVERY
TIEDOWNS
FREQUENT
INFREQUENT
ENDBAND GROUP
CORES
CORES
PRIMARY SEWING
CONSTRUCTION TYPE
NO BEAD
FRONT BEAD
NO FRONT BEAD
REVERSING TWIST
GREEK STYLE
GREEK DOUBLE CORE
WARPS ONLY
TYPE
CROWNING CORE
WITH PRIMARY
TYPE SEWING
WITH SECONDARY
CORE
BOARD ATTACHMENT
SEWN
SEWN& RECESSED
LACED
CUT AT JOINT
ADHERED

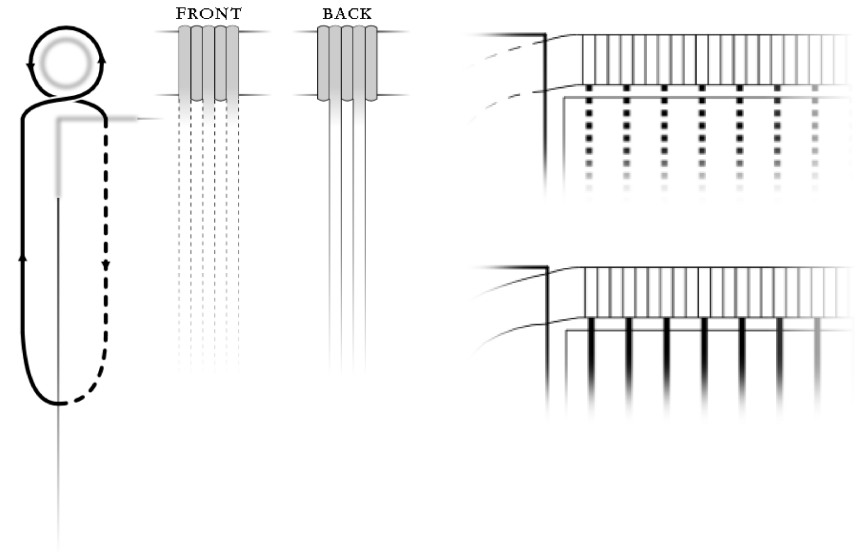
ENDBANDS - I

SEWN

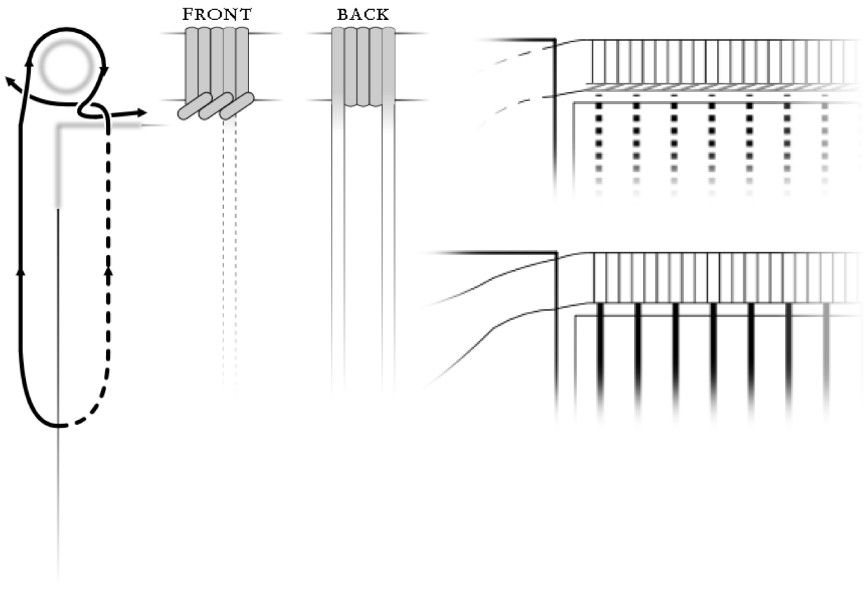


ENDBANDS - 2

NO BEAD

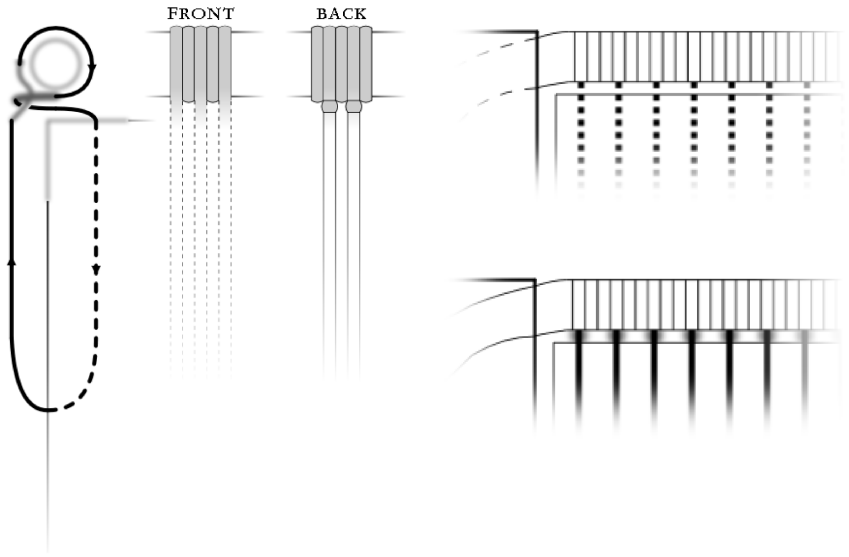


FRONT BEAD

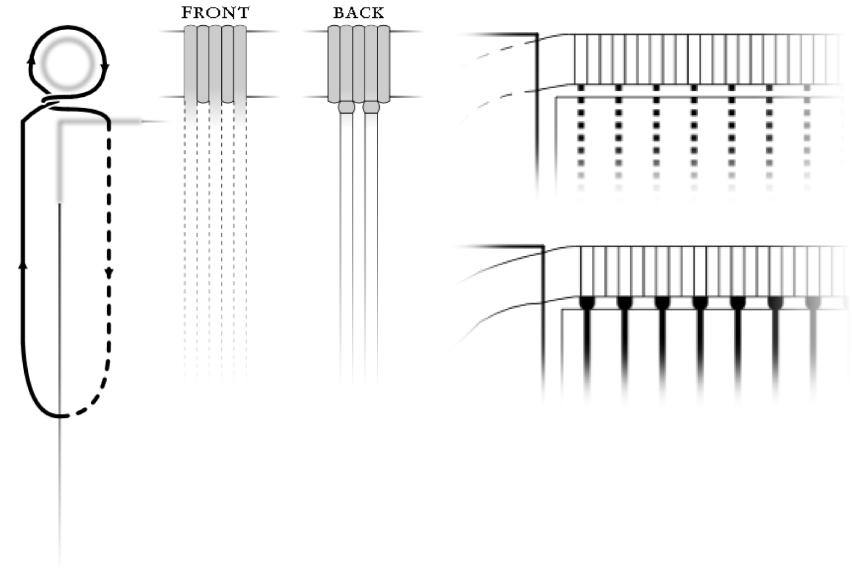


ENDBANDS - 3

NO FRONT BEAD

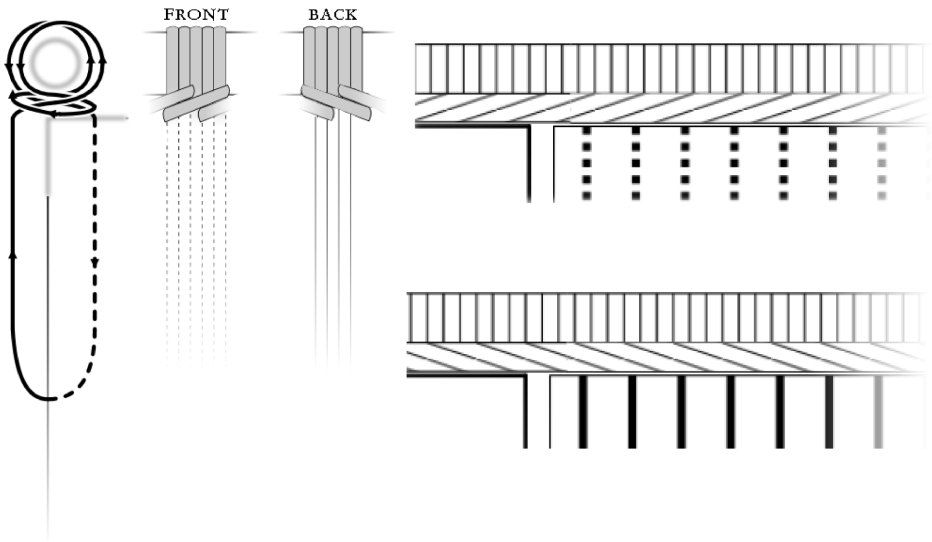


REVERSING TWIST

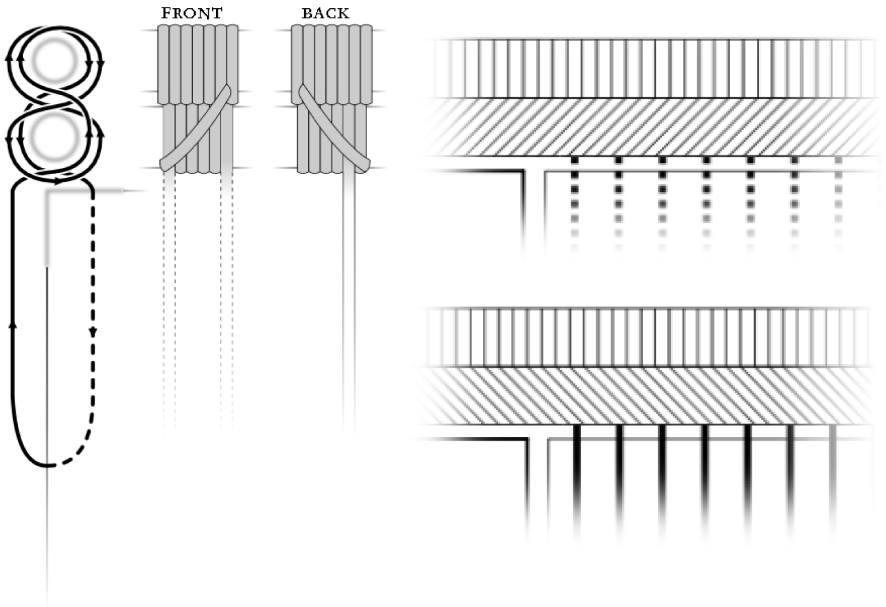


ENDBANDS - 4

GREEK SINGLE CORE

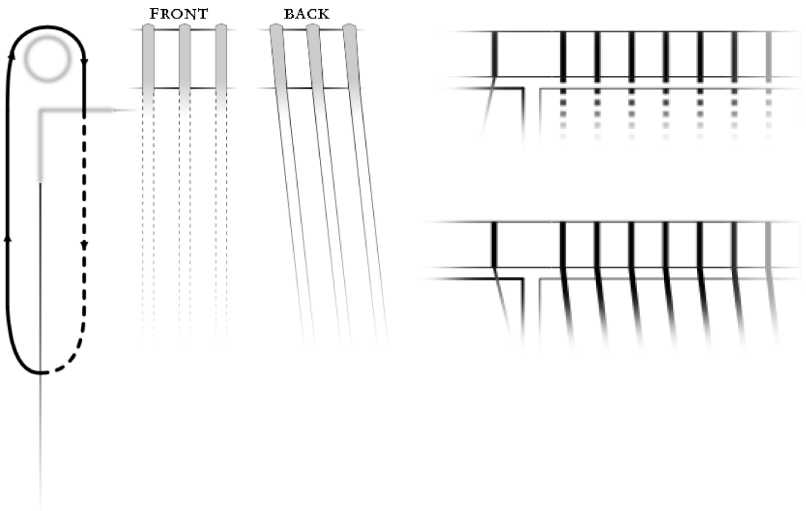


GREEK DOUBLE CORE

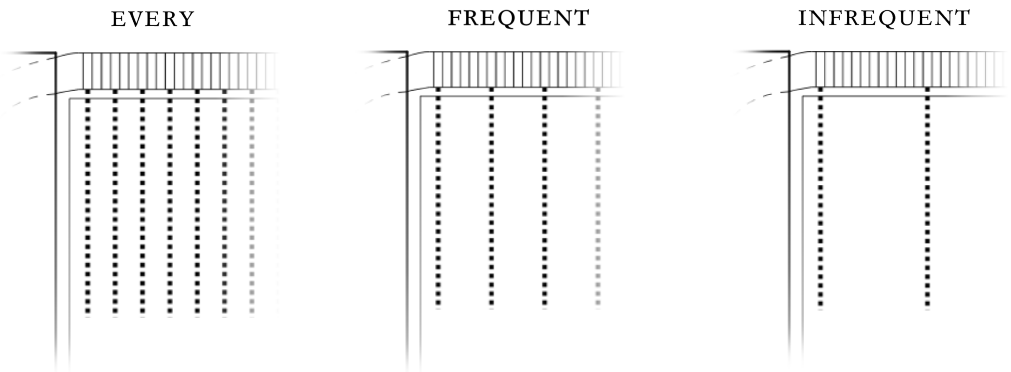


ENDBANDS - 5

WARPS ONLY

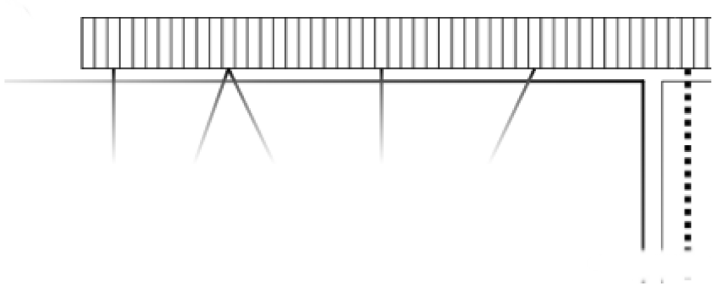


TIEDOWNS

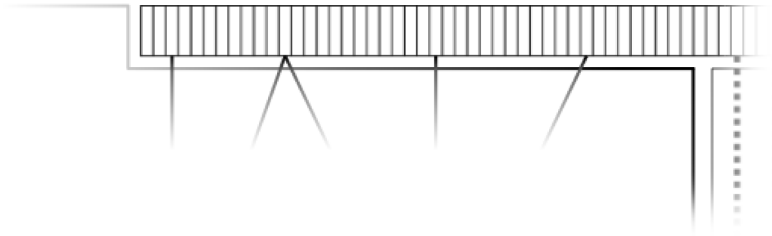


ENDBANDS - 6

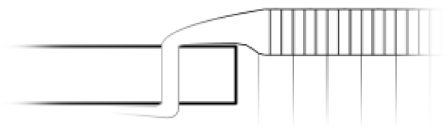
BOARD ATTACHMENT - SEWN

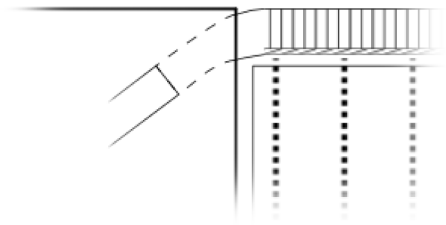


BOARD ATTACHMENT - SEWN AND RECESSED



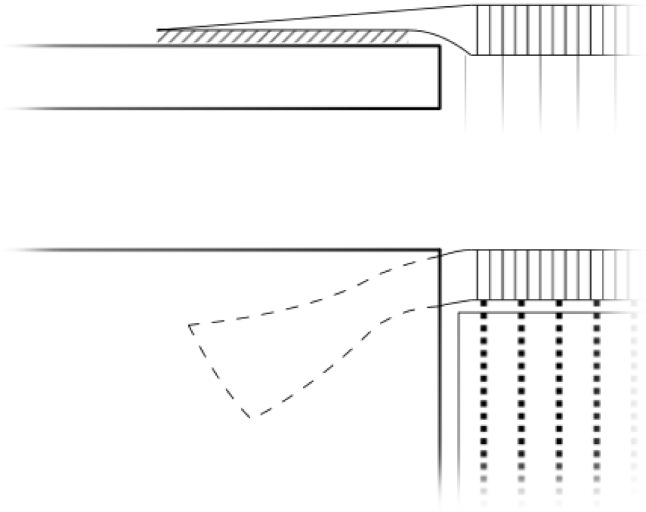
BOARD ATTACHMENT - LACED



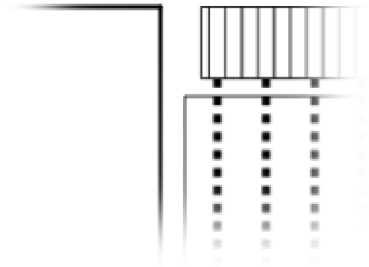


ENDBANDS - 7

BOARD ATTACHMENT - ADHERED

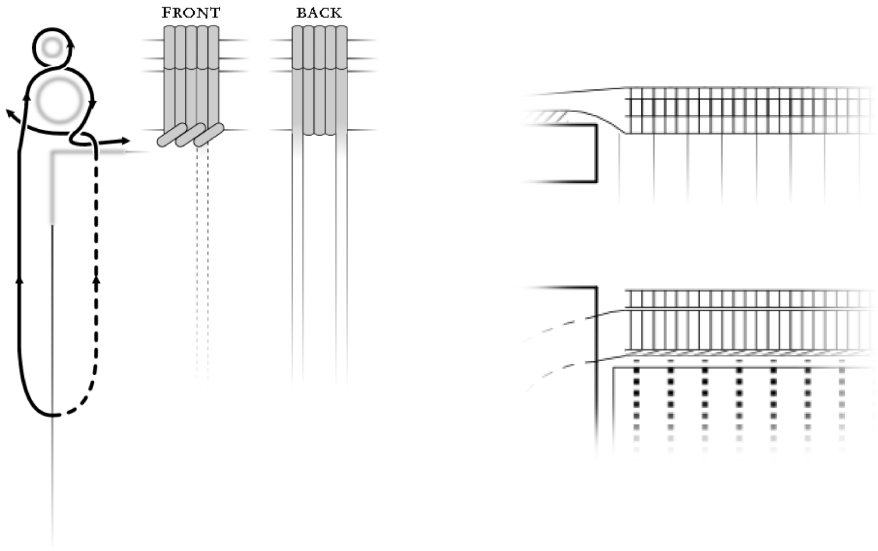


BOARD ATTACHMENT - CUT AT JOINT

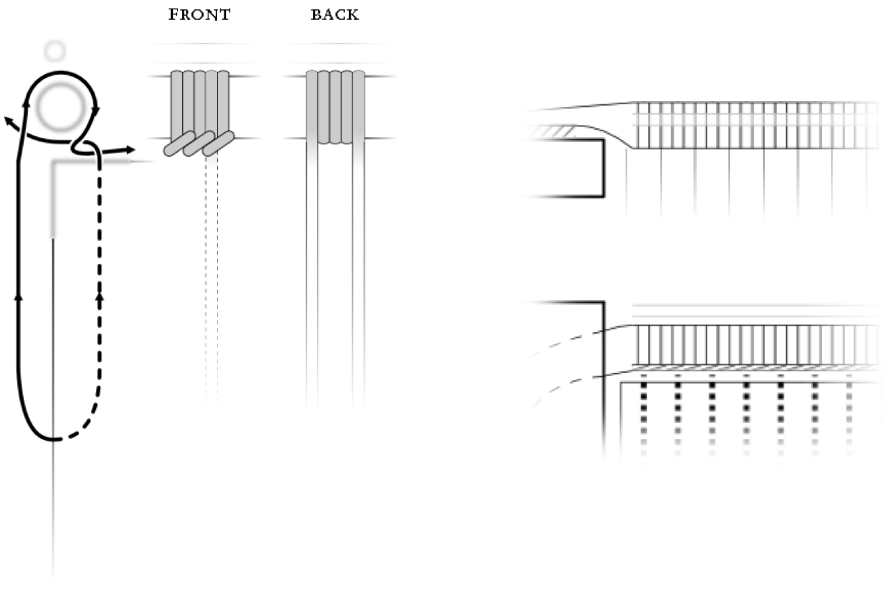


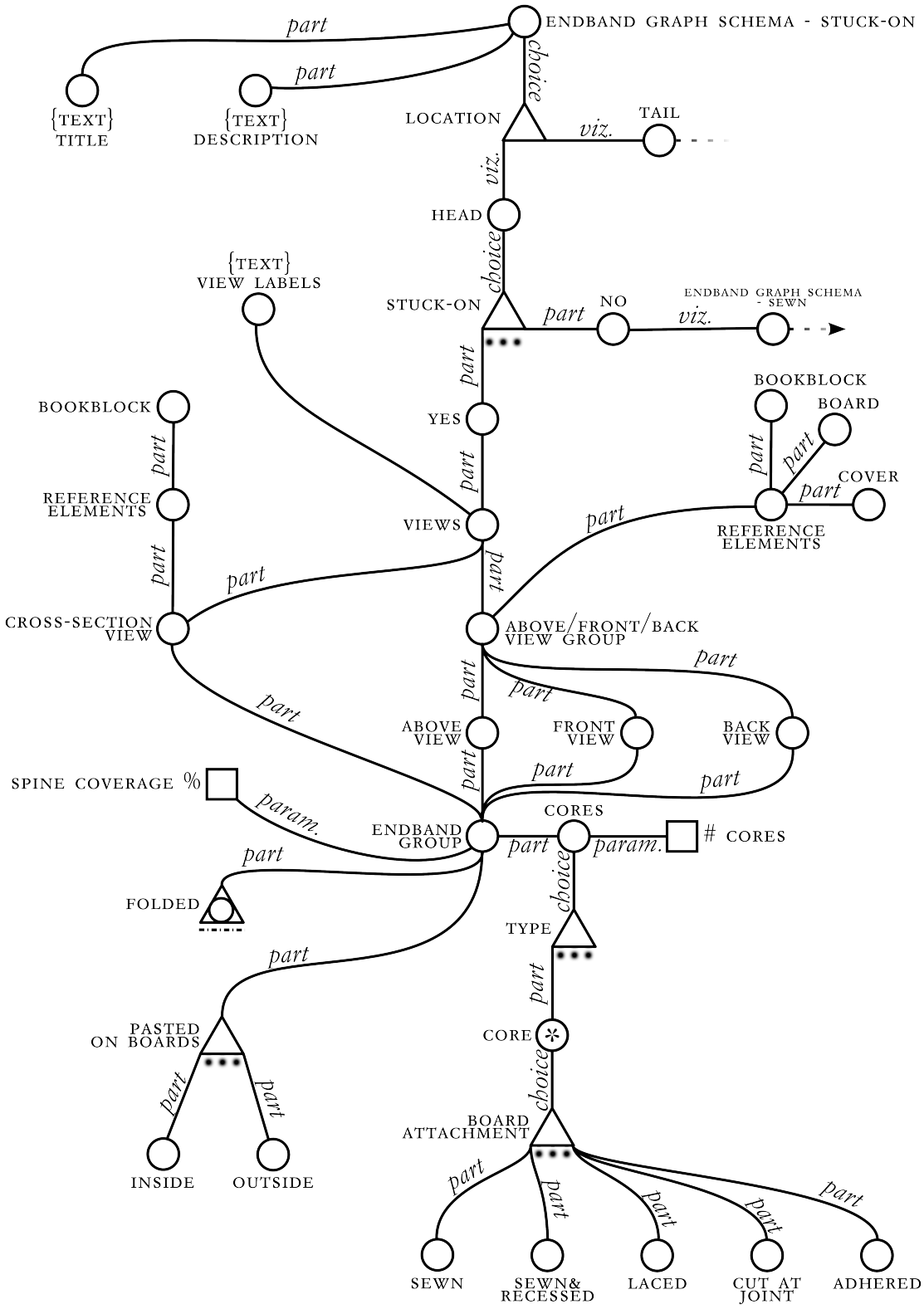
ENDBANDS - 8

CROWNING CORE - SEWN WITH PRIMARY



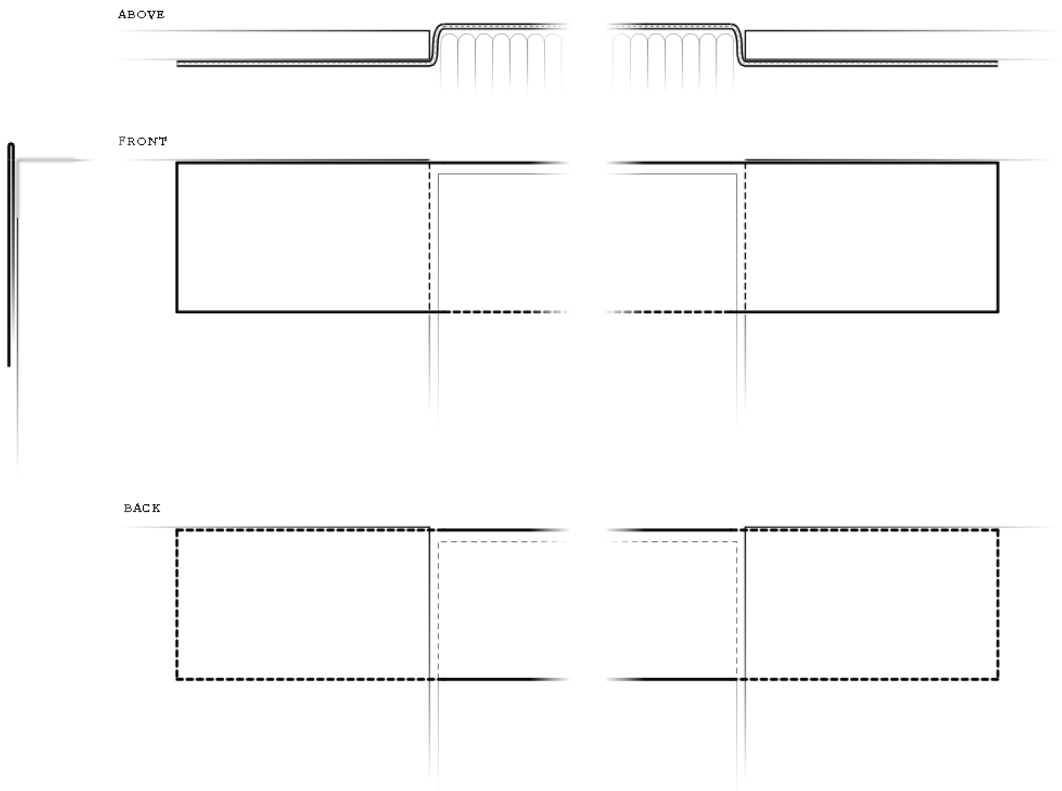
CROWNING CORE - SEWN WITH SECONDARY




ENDBAND GRAPH SCHEMA - STUCK-ON
{TEXT} TITLE
{TEXT} DESCRIPTION
LOCATION
TAIL
HEAD
{TEXT} VIEW LABELS
STUCK-ON
NO
ENDBAND GRAPH SCHEMA - SEWN
YES
BOOKBLOCK
BOARD
COVER
REFERENCE ELEMENTS
VIEWS
REFERENCE ELEMENTS
CROSS-SECTION VIEW
ABOVE/FRONT/BACK VIEW GROUP
ABOVE VIEW
FRONT VIEW
BACK VIEW
SPINE COVERAGE %
ENDBAND GROUP
CORES
CORES
FOLDED
TYPE
PASTED ON BOARDS
CORE
INSIDE
OUTSIDE
BOARD ATTACHMENT
SEWN
SEWN& RECESSED
LACED
CUT AT JOINT
ADHERED

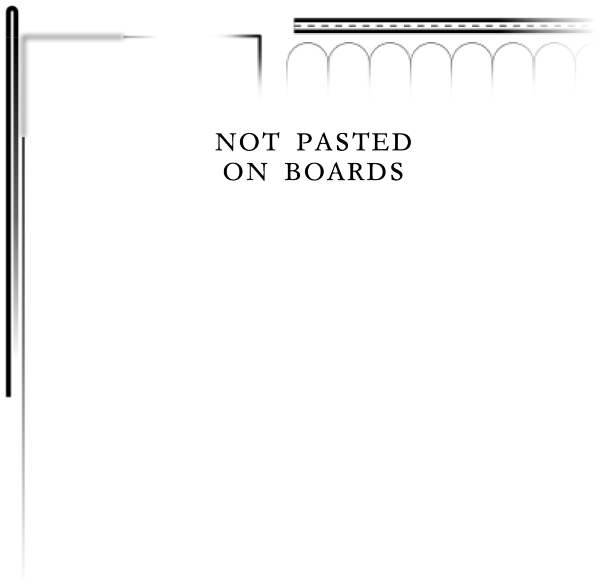
ENDBANDS - 9

STUCK-ON



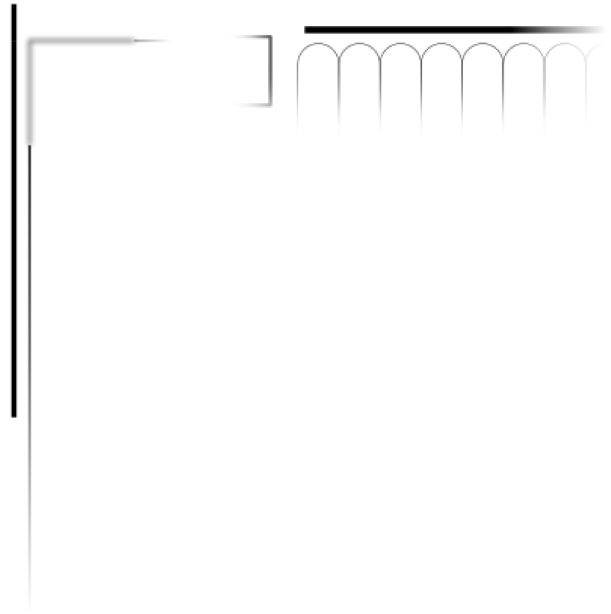
ENDBANDS - IO

FOLDED



NOT PASTED
ON BOARDS

NOT FOLDED



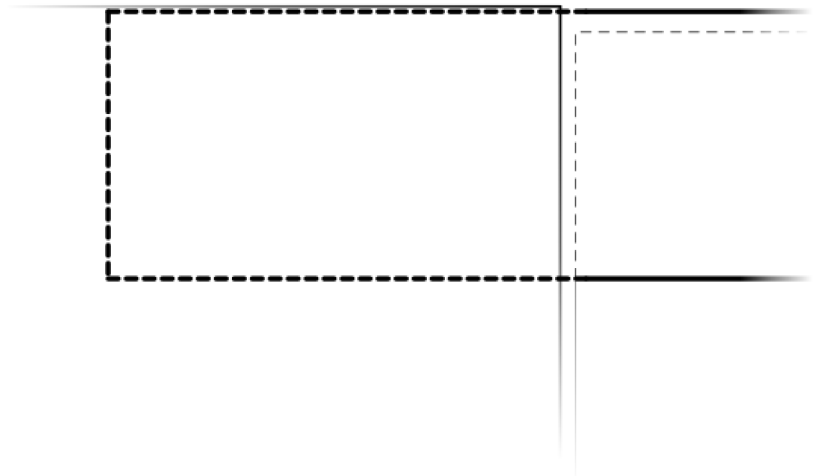
ENDBANDS - II

PASTED INSIDE BOARDS

ABOVE

FRONT

BACK

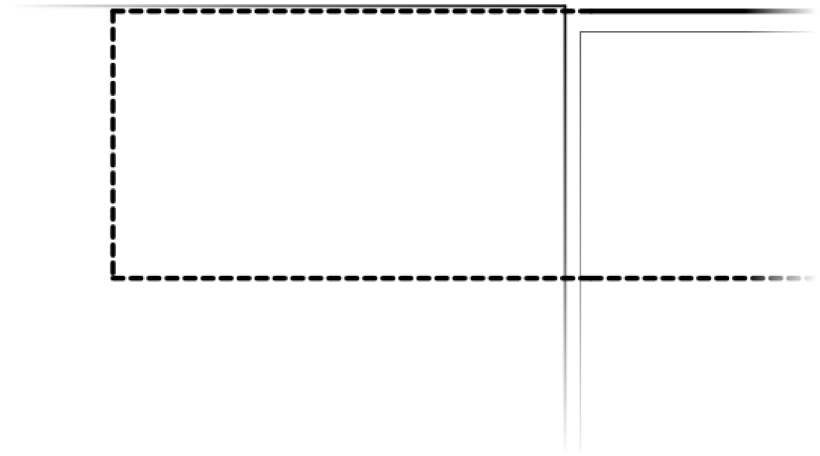


ENDBANDS - 12

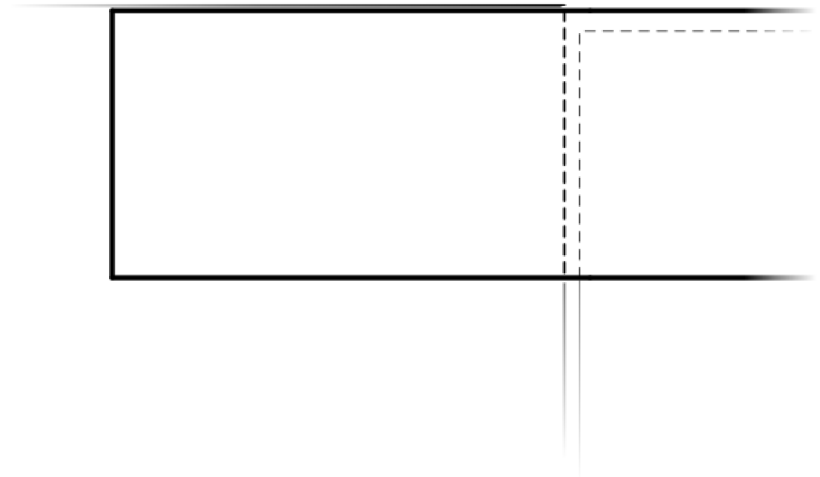
PASTED OUTSIDE BOARDS

ABOVE

FRONT

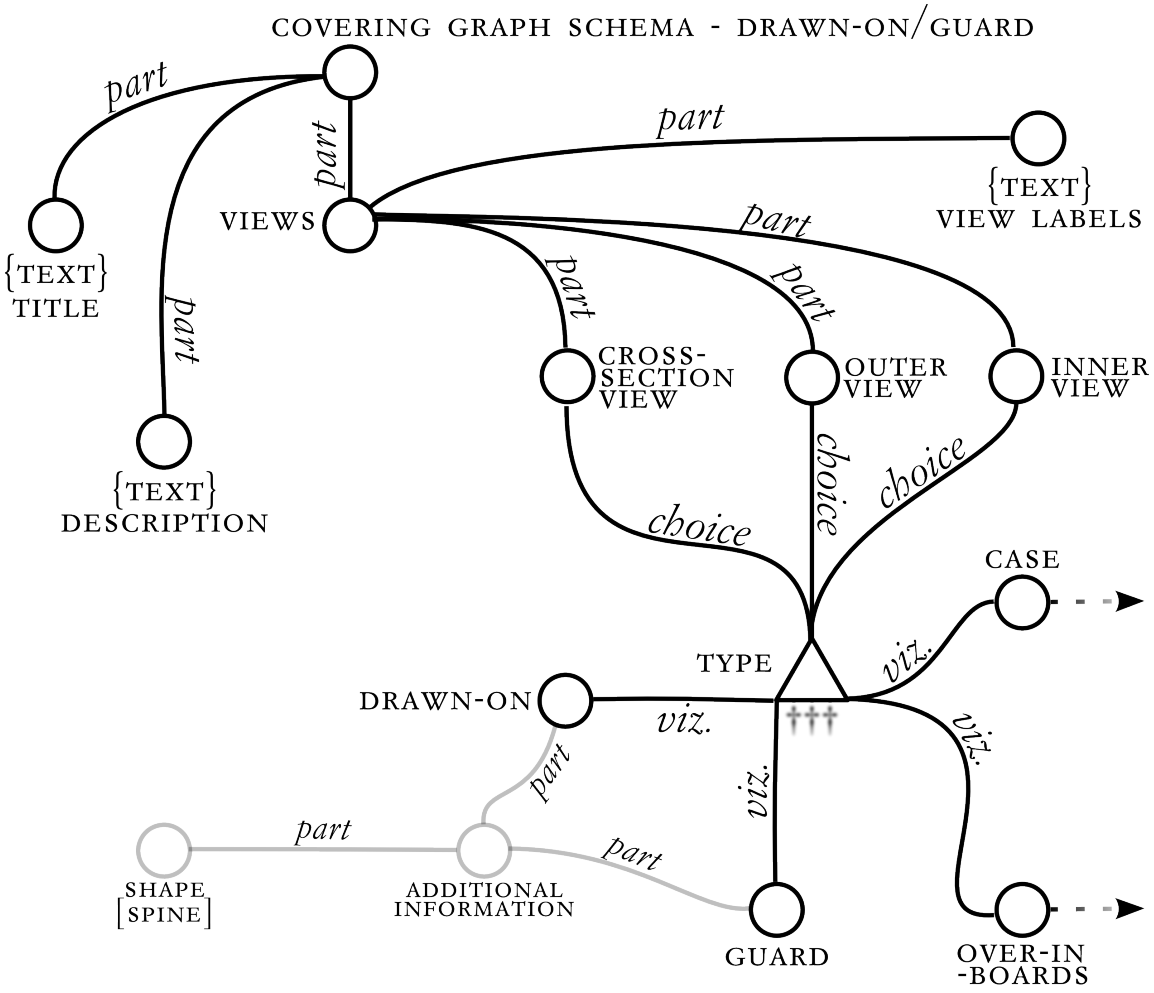


BACK



Coverings

The term *covering* indicates those materials put over a book, with the effect of protecting it. The schema distinguishes between four main kinds of coverings (guard, drawn-on, case, and over-in-boards) indicating rather different constructions. The two more simple types, guard and drawn-on, are not further specified by other parameters. Case and over-in-boards covers, instead, are further specified by means of a range of typologies and parameters. Because of the diversity in features and the wide range of options, the graph schema is presented here as divided in three related schemas: guard and drawn-on together, over-in-board, and case covering respectively. See §7.2.6.

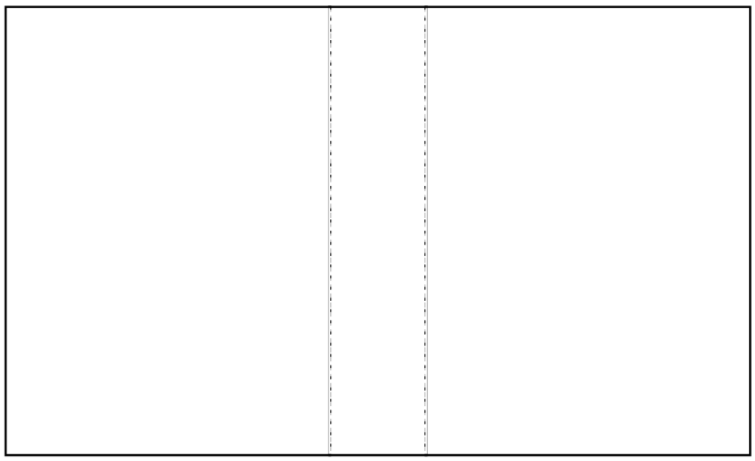

COVERING GRAPH SCHEMA - DRAWN-ON/GUARD
part
part
part
part
part
part
part
part
{TEXT}
TITLE
{TEXT}
DESCRIPTION
VIEWS
{TEXT}
VIEW LABELS
CROSS-
SECTION
VIEW
OUTER
VIEW
INNER
VIEW
choice
choice
choice
CASE
TYPE
DRAWN-ON
viz.
viz.
viz.
viz.
SHAPE
[SPINE]
ADDITIONAL
INFORMATION
GUARD
OVER-IN
-BOARDS

COVERING - I

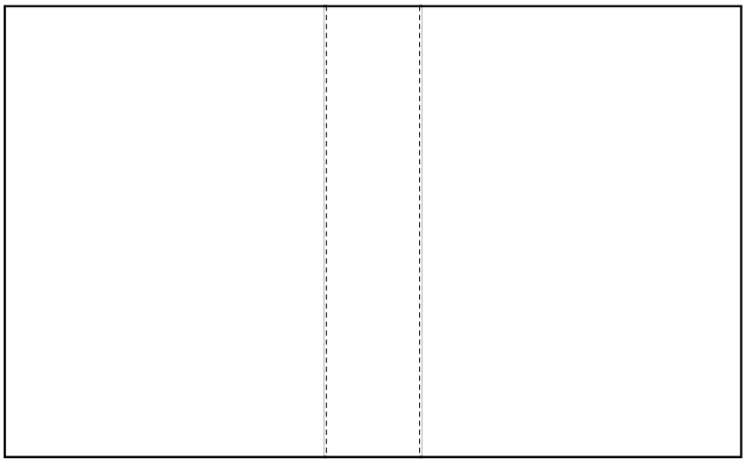
DRAWN-ON

CROSS SECTION

OUTER VIEW



INNER VIEW

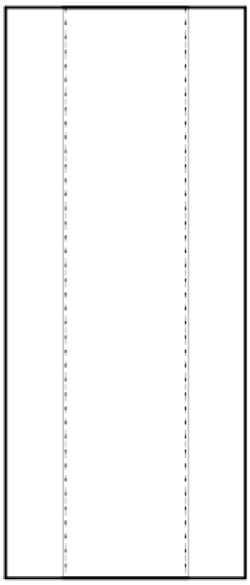


COVERING - 2

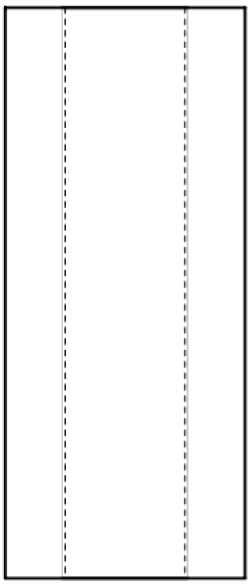
GUARD

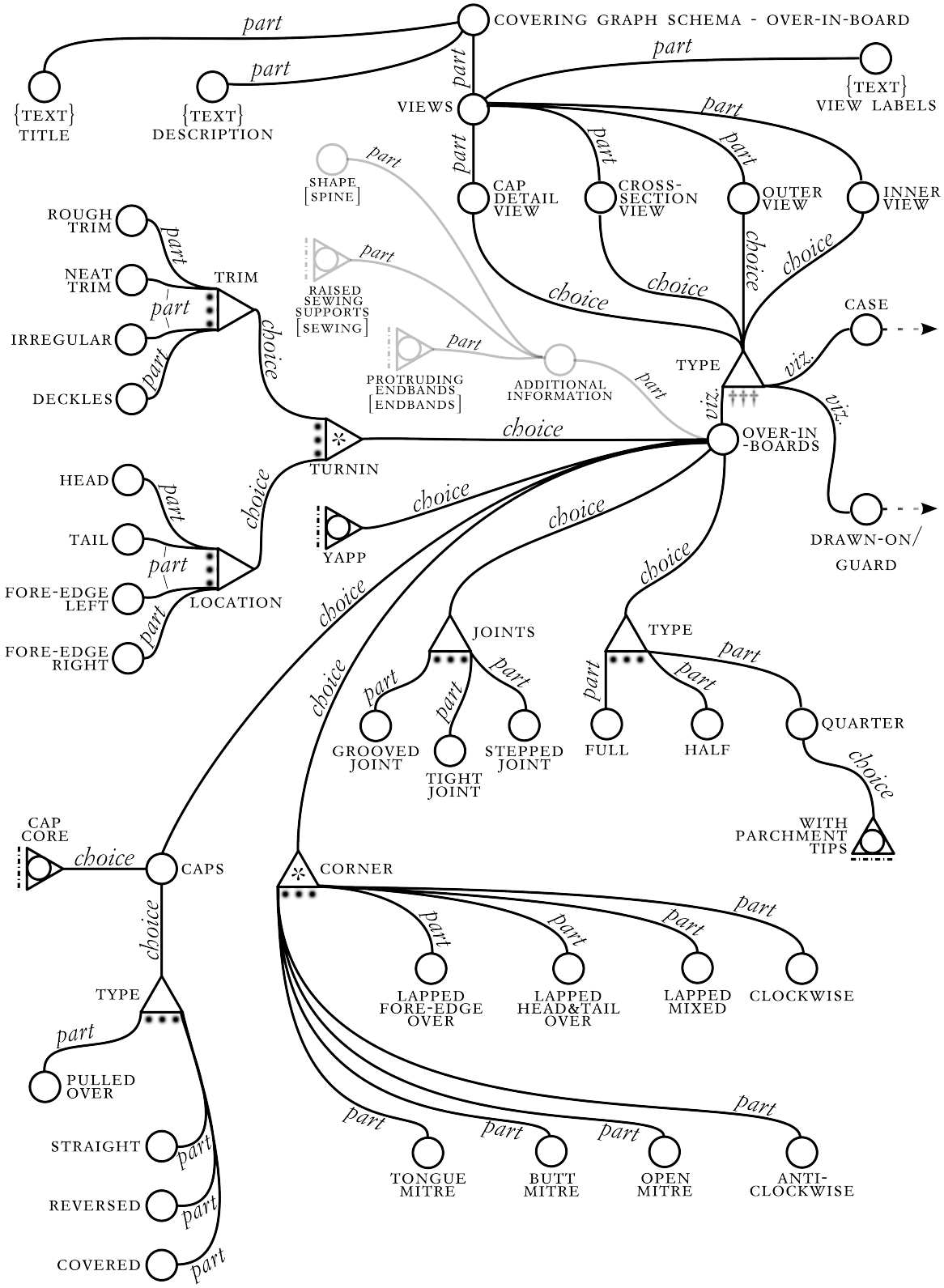
CROSS SECTION

OUTER VIEW



INNER VIEW




COVERING GRAPH SCHEMA - OVER-IN-BOARD
{TEXT} TITLE
{TEXT} DESCRIPTION
VIEWS
{TEXT} VIEW LABELS
CAP DETAIL VIEW
CROSS-SECTION VIEW
OUTER VIEW
INNER VIEW
SHAPE [SPINE]
RAISED SEWING SUPPORTS [SEWING]
PROTRUDING ENDBANDS [ENDBANDS]
ADDITIONAL INFORMATION
ROUGH TRIM
NEAT TRIM
IRREGULAR
DECKLES
TRIM
TURNIN
CASE
TYPE
OVER-IN-BOARDS
DRAWN-ON/ GUARD
HEAD
TAIL
FORE-EDGE LEFT
FORE-EDGE RIGHT
LOCATION
YAPP
JOINTS
GROOVED JOINT
TIGHT JOINT
STEPPED JOINT
TYPE
FULL
HALF
QUARTER
WITH PARCHMENT TIPS
CAP CORE
CAPS
CORNER
TYPE
PULLED OVER
STRAIGHT
REVERSED
COVERED
LAPPED FORE-EDGE OVER
LAPPED HEAD&TAIL OVER
LAPPED MIXED
CLOCKWISE
TONGUE MITRE
BUTT MITRE
OPEN MITRE
ANTI-CLOCKWISE
part
choice
viz.

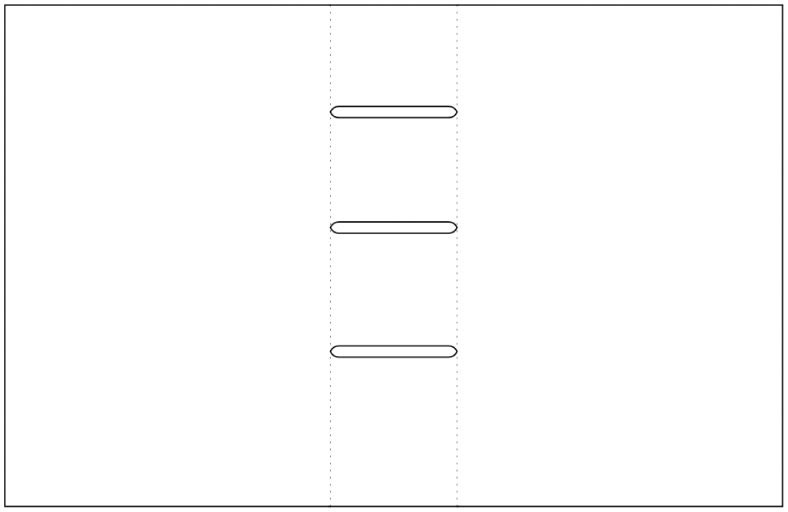
COVERING - 3

OVER IN BOARD - FULL

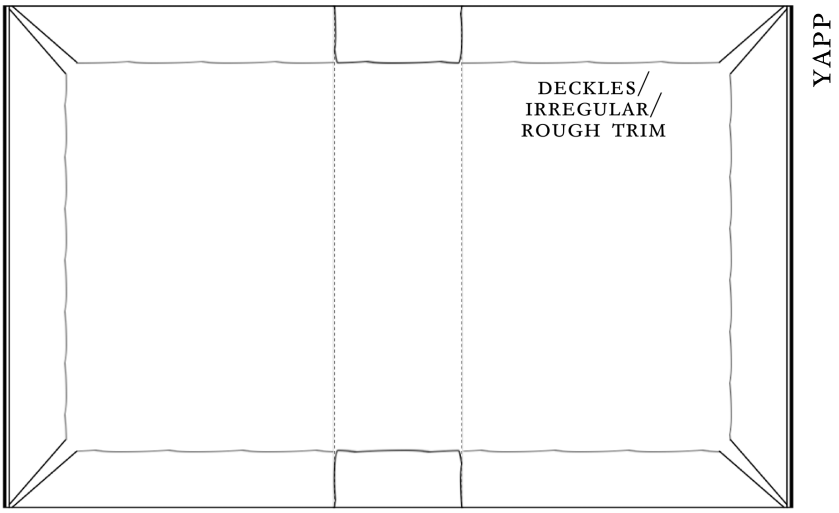
CAP DETAIL:
STRAIGHT

CROSS SECTION

OUTER VIEW



INNER VIEW



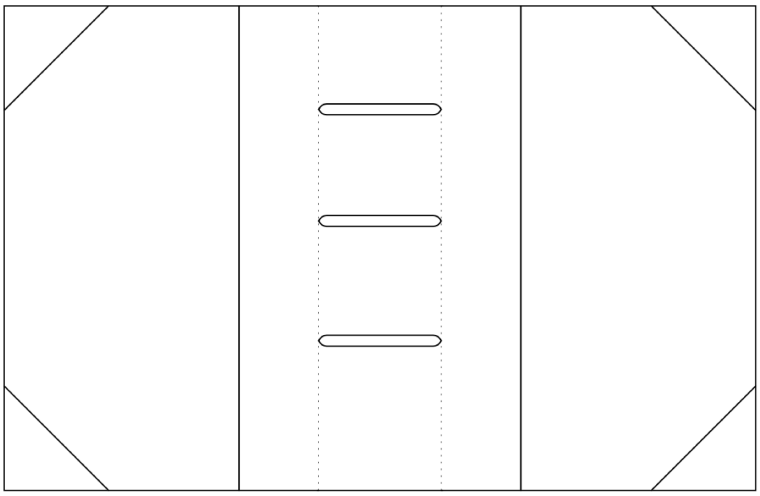
COVERING - 4

OVER IN BOARD - HALF

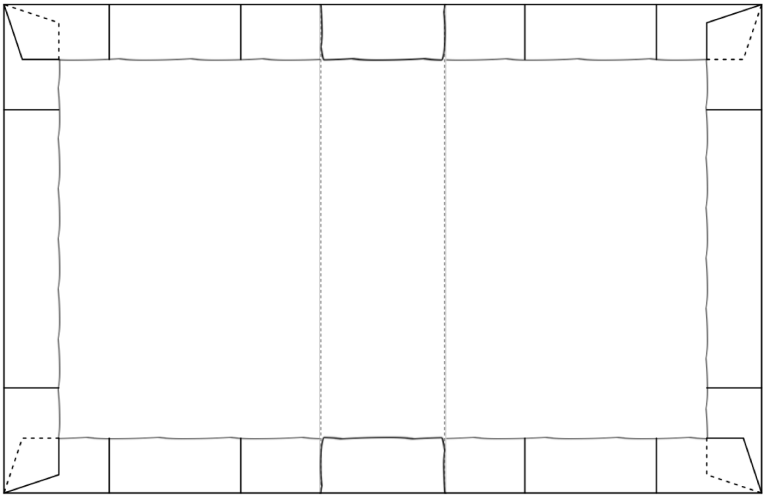
CAP DETAIL:
STRAIGHT

CROSS SECTION

OUTER VIEW

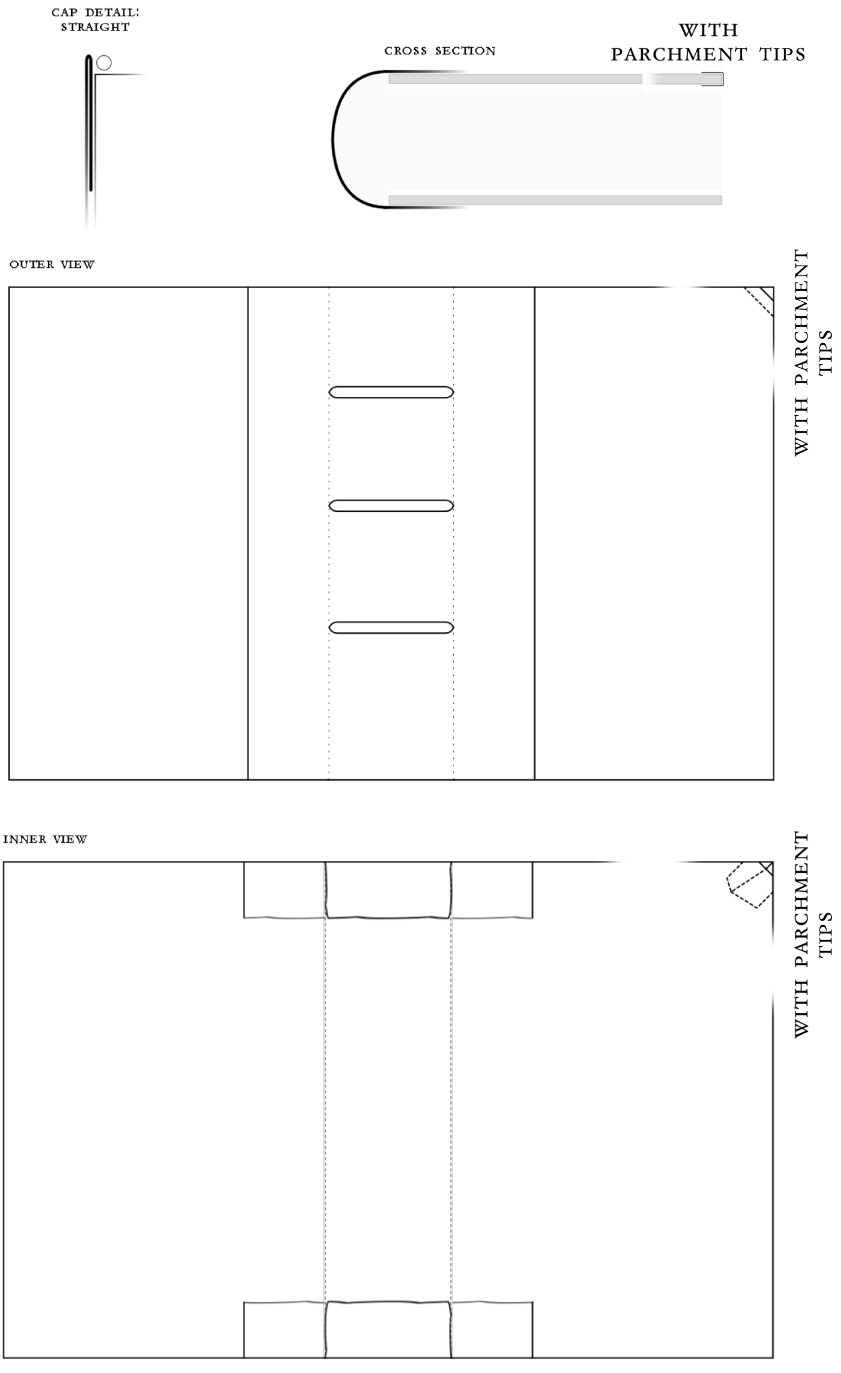


INNER VIEW



COVERING - 5

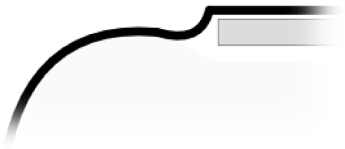
OVER IN BOARD - HALF



COVERING - 6

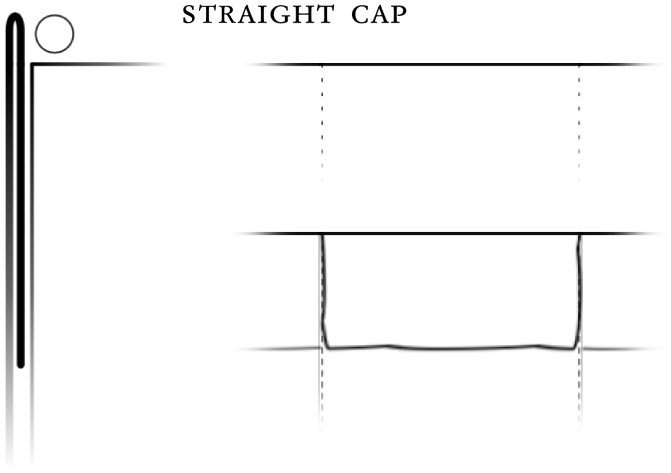
GROOVED JOINT

STEPPED JOINT

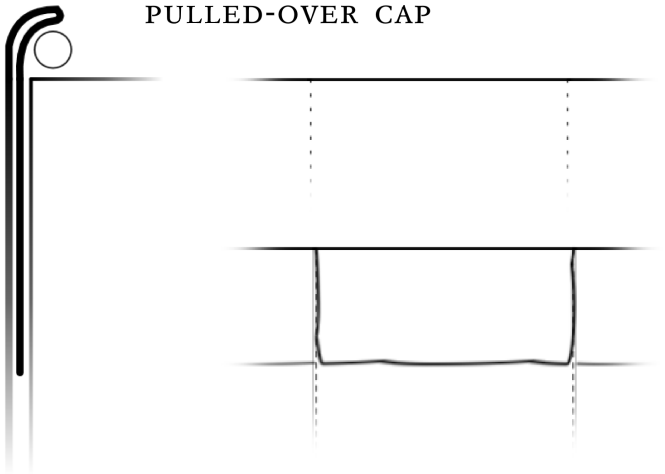


TIGHT JOINT

STRAIGHT CAP

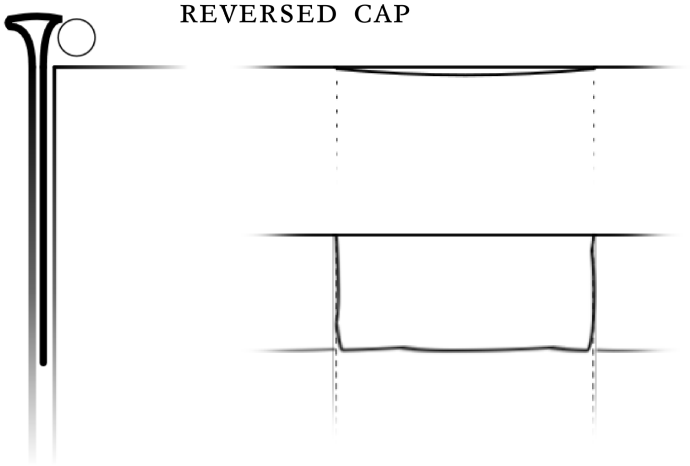


PULLED-OVER CAP

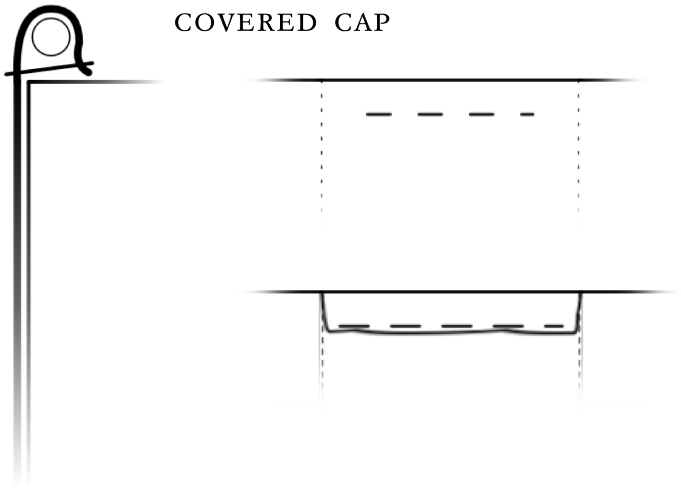


COVERING - 7

REVERSED CAP

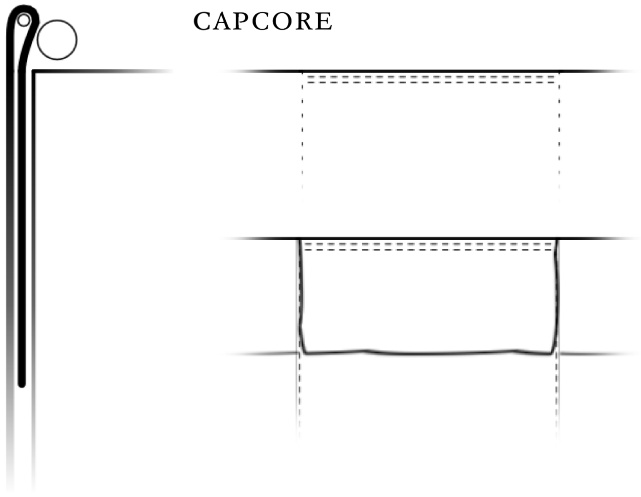


COVERED CAP

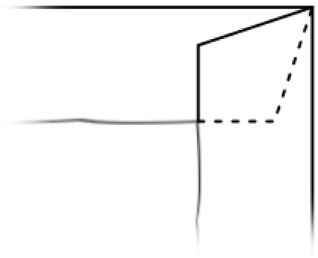


COVERING - 8

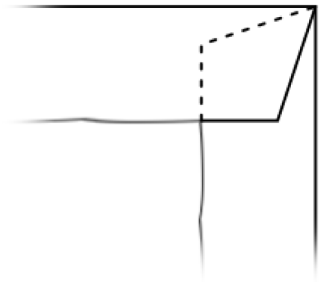
CAPCORE



LAPPED FORE-EDGE OVER

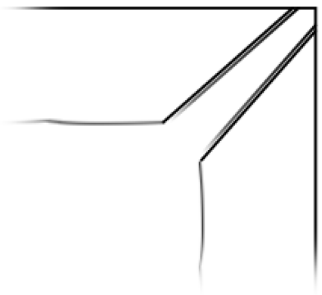


LAPPED HEAD & TAIL OVER



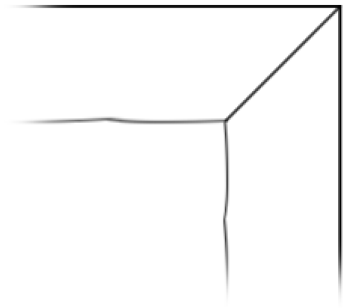
LAPPED MIXED

TONGUED MITRE

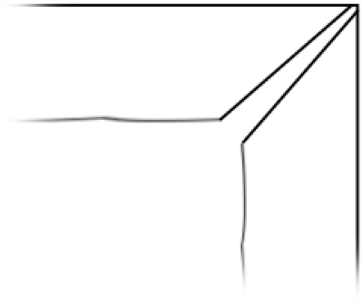


COVERING - 9

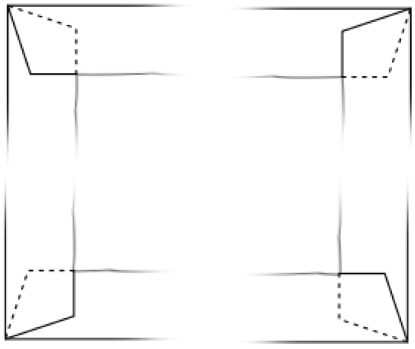
BUTT MITRE



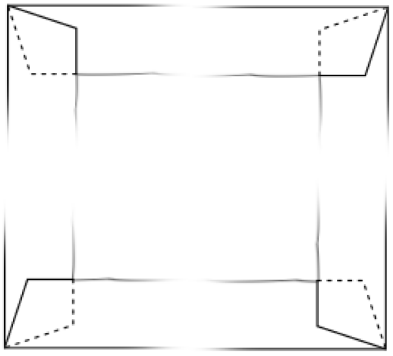
OPEN MITRE

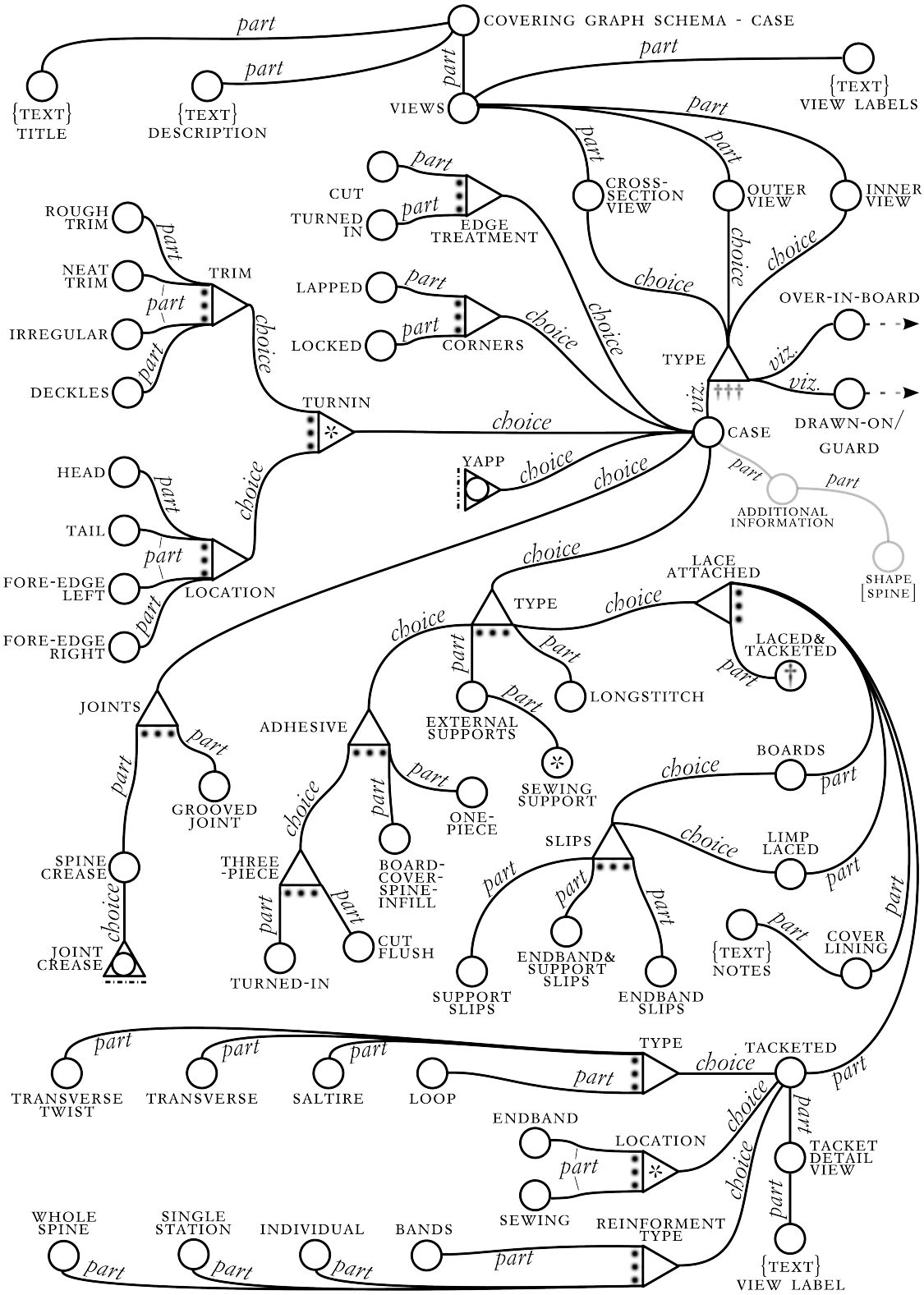


CLOCKWISE



ANTI-CLOCKWISE



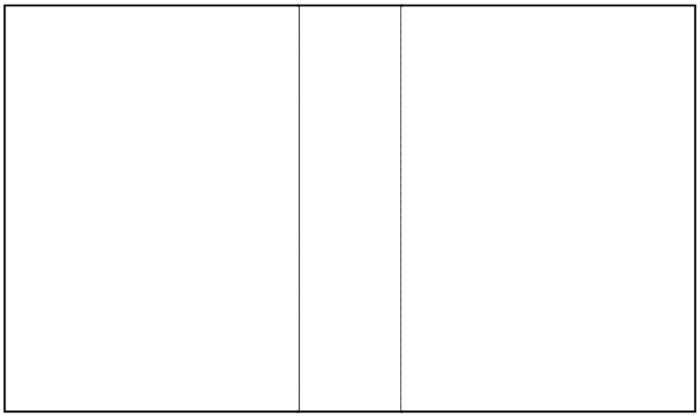

COVERING GRAPH SCHEMA - CASE
{TEXT} TITLE
{TEXT} DESCRIPTION
VIEWS
{TEXT} VIEW LABELS
CROSS-SECTION VIEW
OUTER VIEW
INNER VIEW
CUT
TURNED IN
EDGE TREATMENT
ROUGH TRIM
NEAT TRIM
IRREGULAR
DECKLES
TRIM
LAPPED
LOCKED
CORNERS
TURNIN
OVER-IN-BOARD
TYPE
DRAWN-ON/ GUARD
CASE
YAPP
ADDITIONAL INFORMATION
SHAPE [SPINE]
HEAD
TAIL
FORE-EDGE LEFT
FORE-EDGE RIGHT
LOCATION
LACE ATTACHED
LACED& TACKETED
LONGSTITCH
EXTERNAL SUPPORTS
SEWING SUPPORT
JOINTS
ADHESIVE
GROOVED JOINT
ONE-PIECE
BOARDS
SPINE CREASE
JOINT CREASE
THREE -PIECE
BOARD-COVER-SPINE-INFILL
TURNED-IN
CUT FLUSH
SLIPS
LIMP LACED
SUPPORT SLIPS
ENDBAND& SUPPORT SLIPS
ENDBAND SLIPS
{TEXT} NOTES
COVER LINING
TRANSVERSE TWIST
TRANSVERSE
SALTIRE
LOOP
TACKETED
ENDBAND
LOCATION
SEWING
REINFORMENT TYPE
TACKET DETAIL VIEW
WHOLE SPINE
SINGLE STATION
INDIVIDUAL
BANDS
{TEXT} VIEW LABEL

COVERING - 10

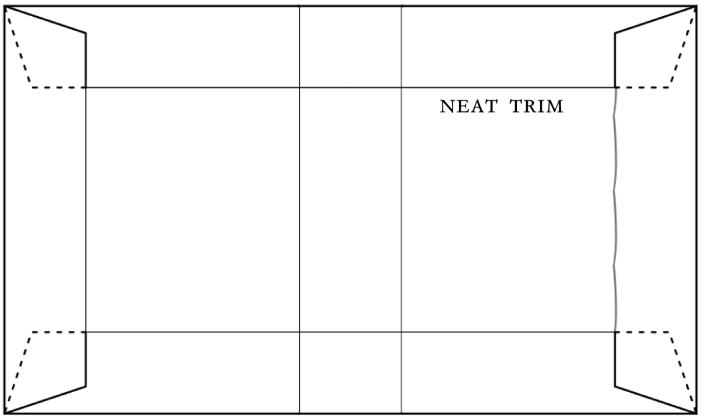
CASE - ADHESIVE - ONE PIECE

CROSS SECTION

OUTER VIEW



INNER VIEW

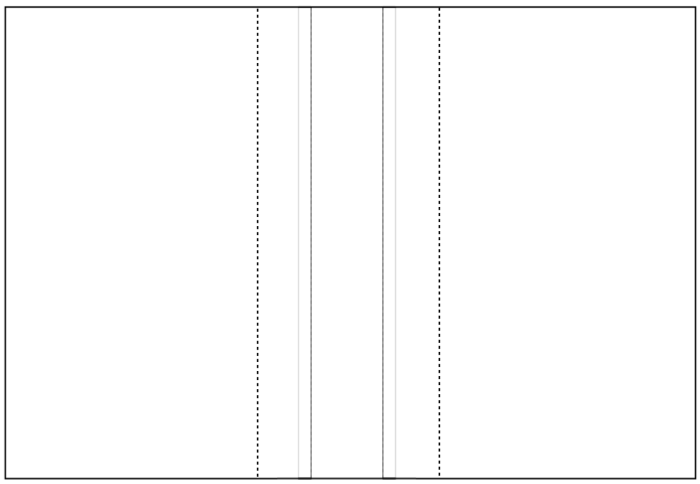


COVERING - II

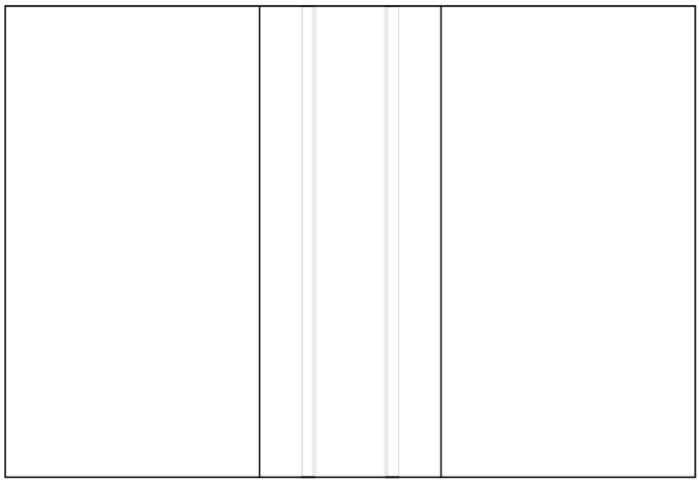
CASE - ADHESIVE - THREE PIECE (CUT FLUSH)

CROSS SECTION

OUTER VIEW



INNER VIEW

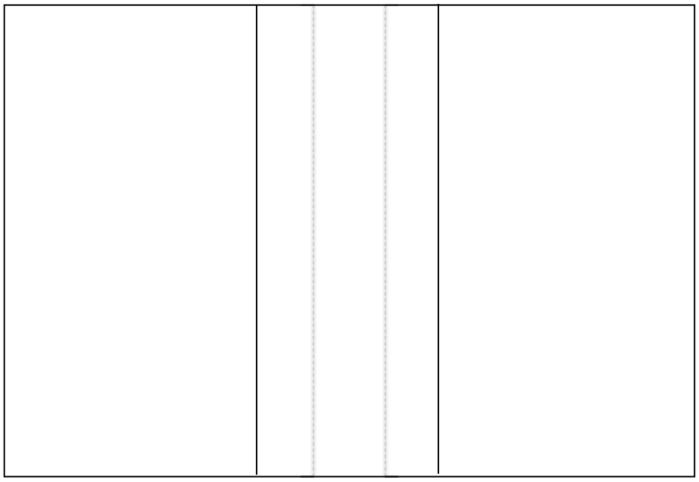


COVERING - 12

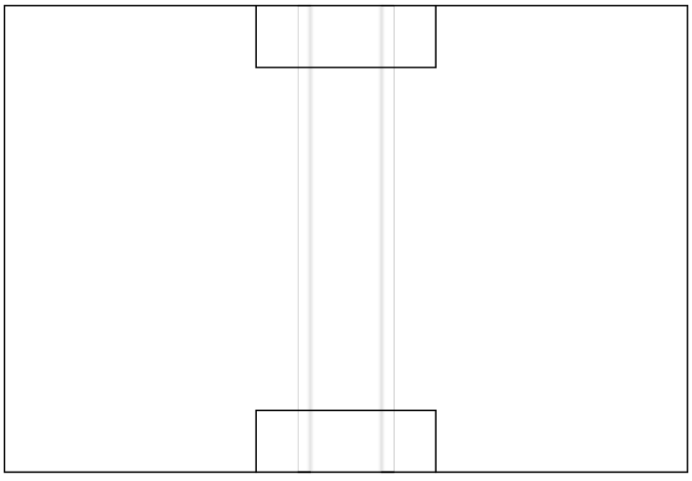
CASE - ADHESIVE - THREE PIECE (TURNED IN)

CROSS SECTION

OUTER VIEW



INNER VIEW

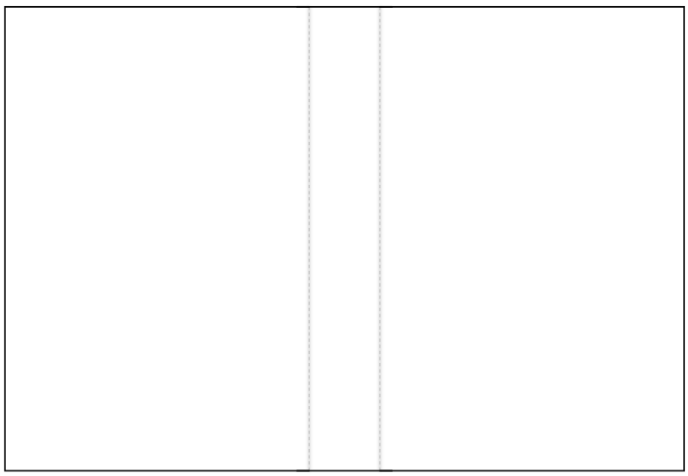


COVERING - 13

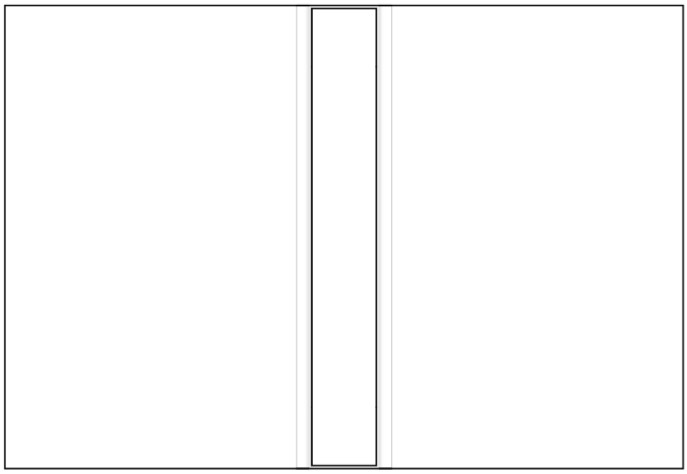
CASE - ADHESIVE - BOARD COVER SPINE INFILL

CROSS SECTION

OUTER VIEW



INNER VIEW

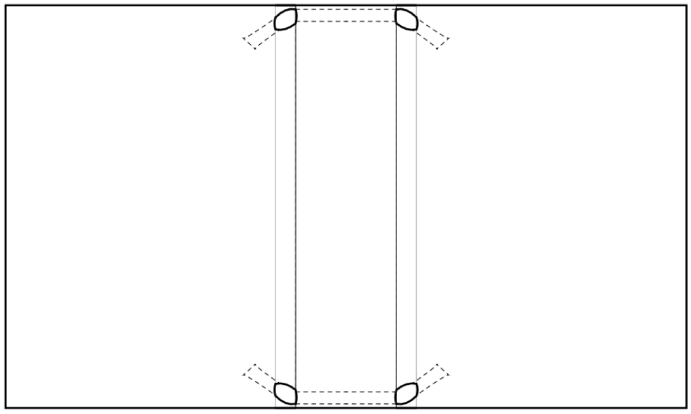


COVERING - 14

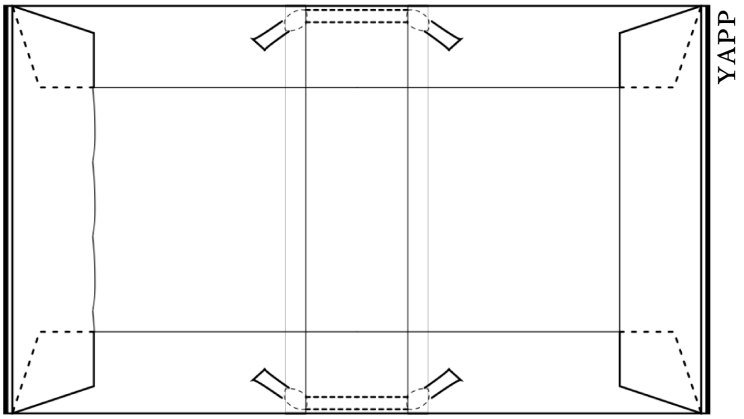
CASE - LACE ATTACHED - LIMP LACED (ENDBAND SLIPS)

CROSS SECTION

OUTER VIEW



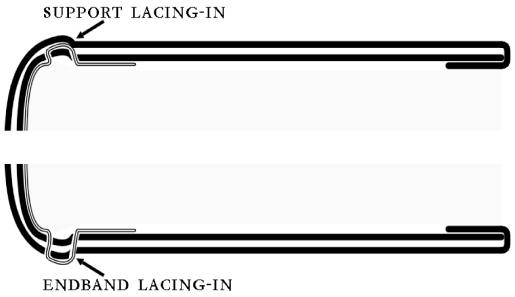
INNER VIEW



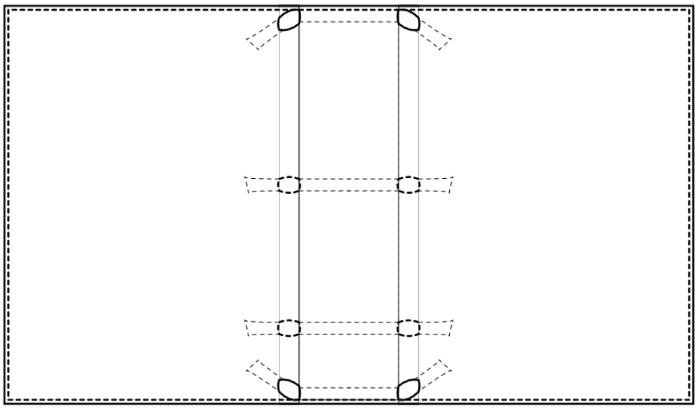
COVERING - 15

CASE - LACE ATTACHED - COVER LINING

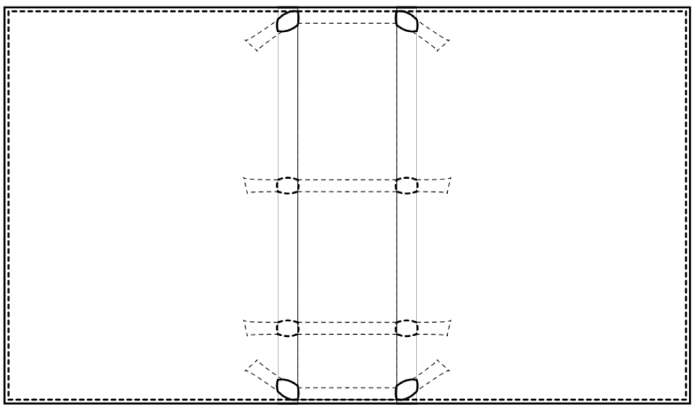
CROSS SECTION



OUTER VIEW



OUTER VIEW

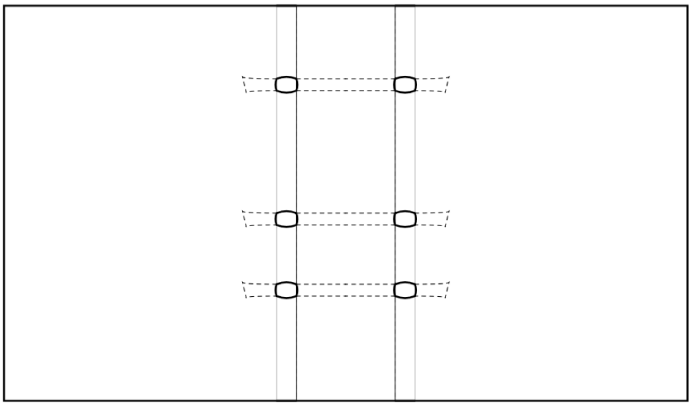


COVERING - 16

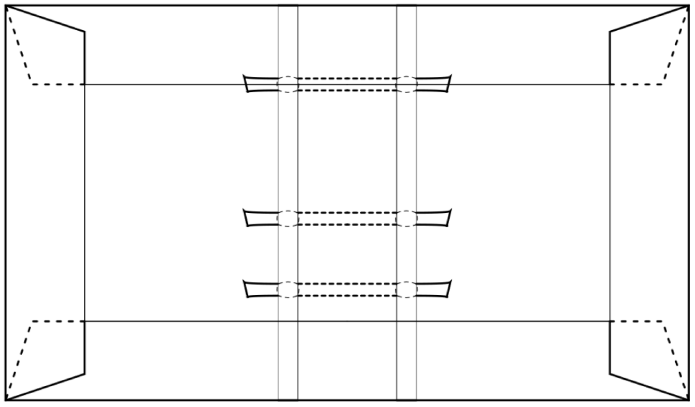
CASE - LACE ATTACHED - BOARDS (SUPPORT SLIPS)

CROSS SECTION

OUTER VIEW



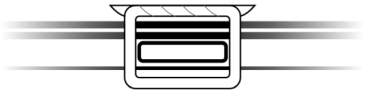
INNER VIEW



COVERING - 17

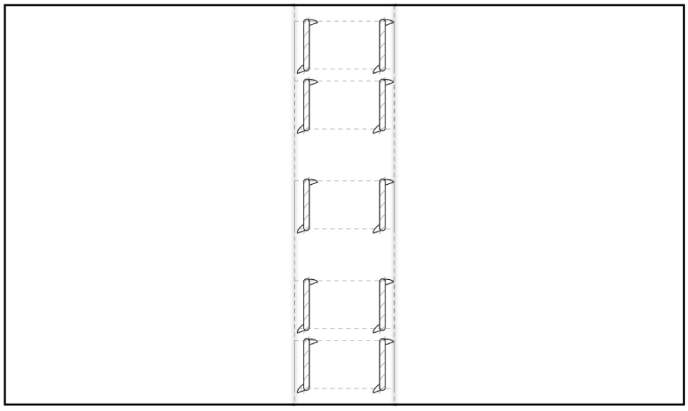
CASE - LACE ATTACHED - TACKETED (LOOP)

TACKET DETAIL

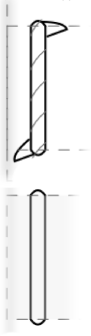


CROSS SECTION

OUTER VIEW

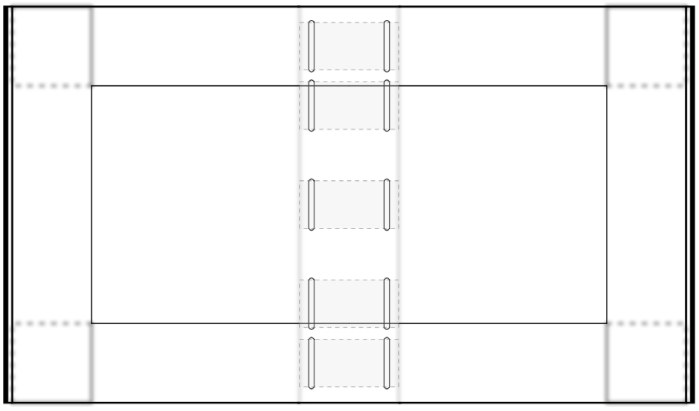


LOC. SEWING

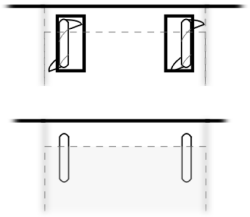


LOC. SEWING

INNER VIEW



LOC. ENDBAND

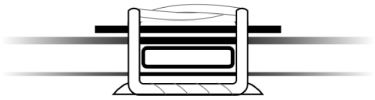


INDIVIDUAL REINF.

COVERING - 18

CASE - LACE ATTACHED - TACKETED (SALTIRE)

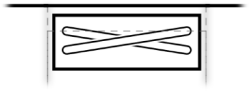
TACKET DETAIL



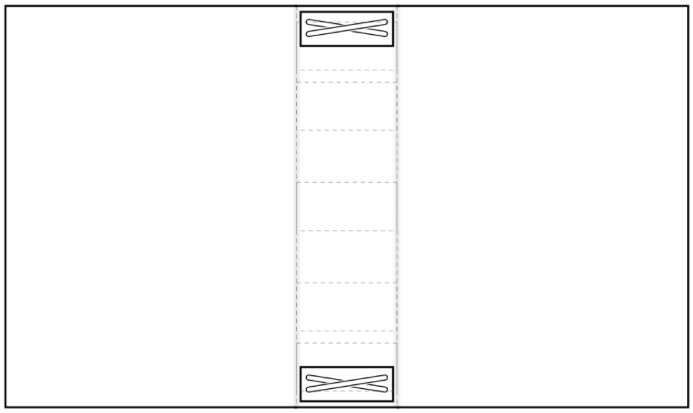
CROSS SECTION

OUTER VIEW

LOC. ENDBAND



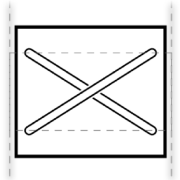
SINGLE STATION REINF.

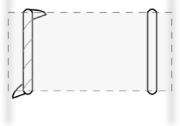


LOC. ENDBAND

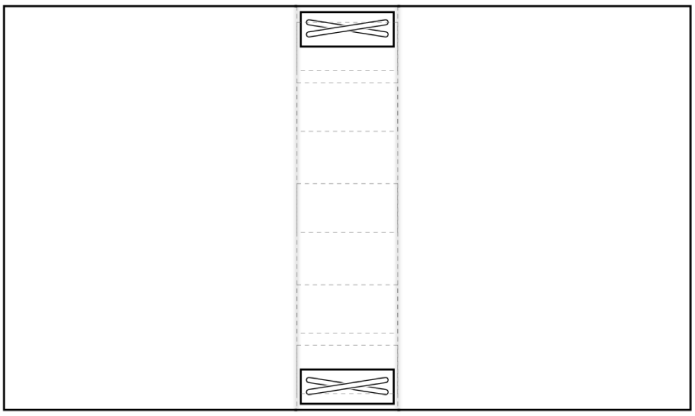
OUTER VIEW

LOC. SEWING





SINGLE STATION REINF.



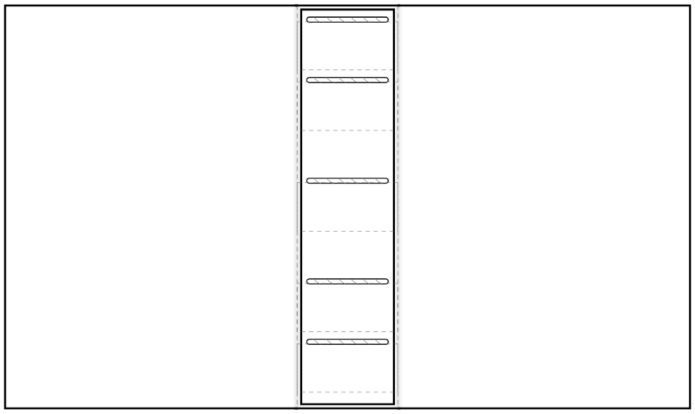
COVERING - 19

CASE - LACE ATTACHED - TACKETED (TRANSVERSE TWISTED)

TACKET DETAIL

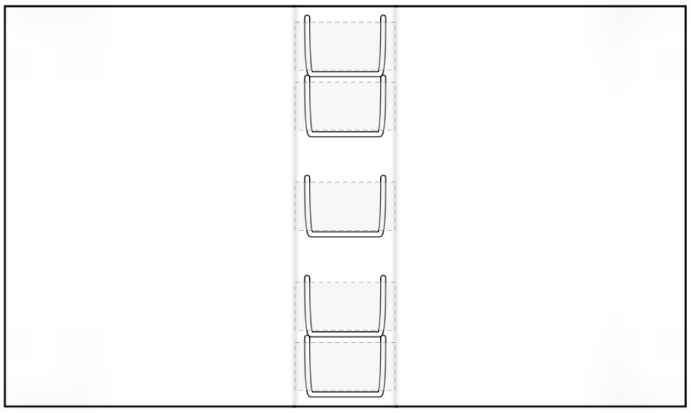
CROSS SECTION

OUTER VIEW



LOC. SEWING -
WHOLESPINE REINF.

INNER VIEW



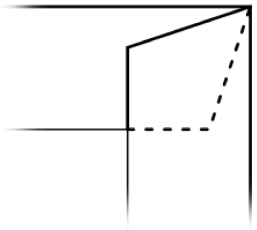
COVERING - 20

CASE - LACE ATTACHED - TACKETED (TRANSVERSE)

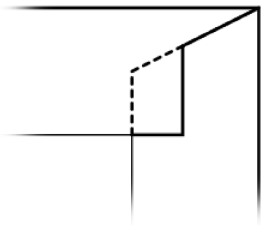
TACKET DETAIL

CROSS SECTION

LAPPED CORNER



LOCKED CORNER



EDGE TREATMENT:
CUT

EDGE TREATMENT:
TURNED IN

GROOVED JOINT

SPINE CREASE

WITHOUT JOINT CREASE

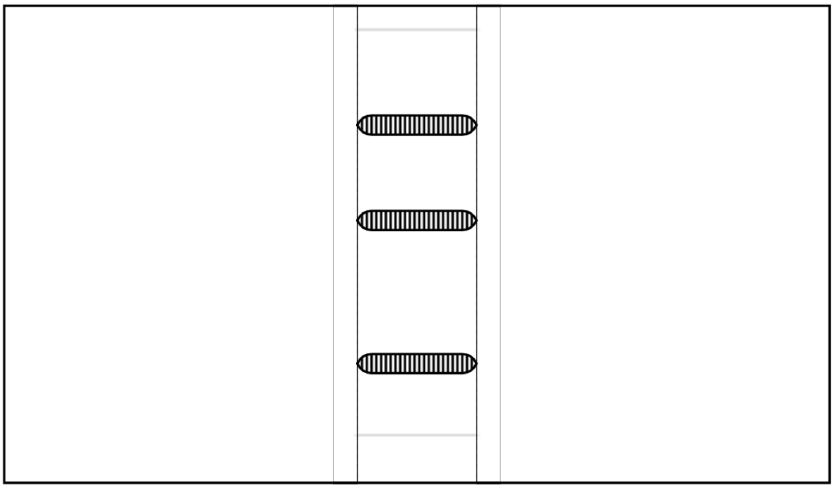
SPINE CREASE

WITH JOINT CREASE

COVERING - 21

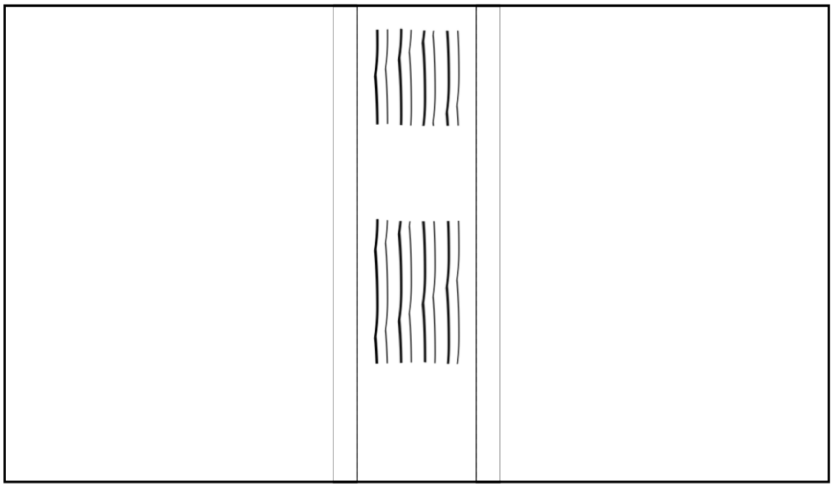
CASE - EXTERNAL SUPPORTS

OUTER VIEW



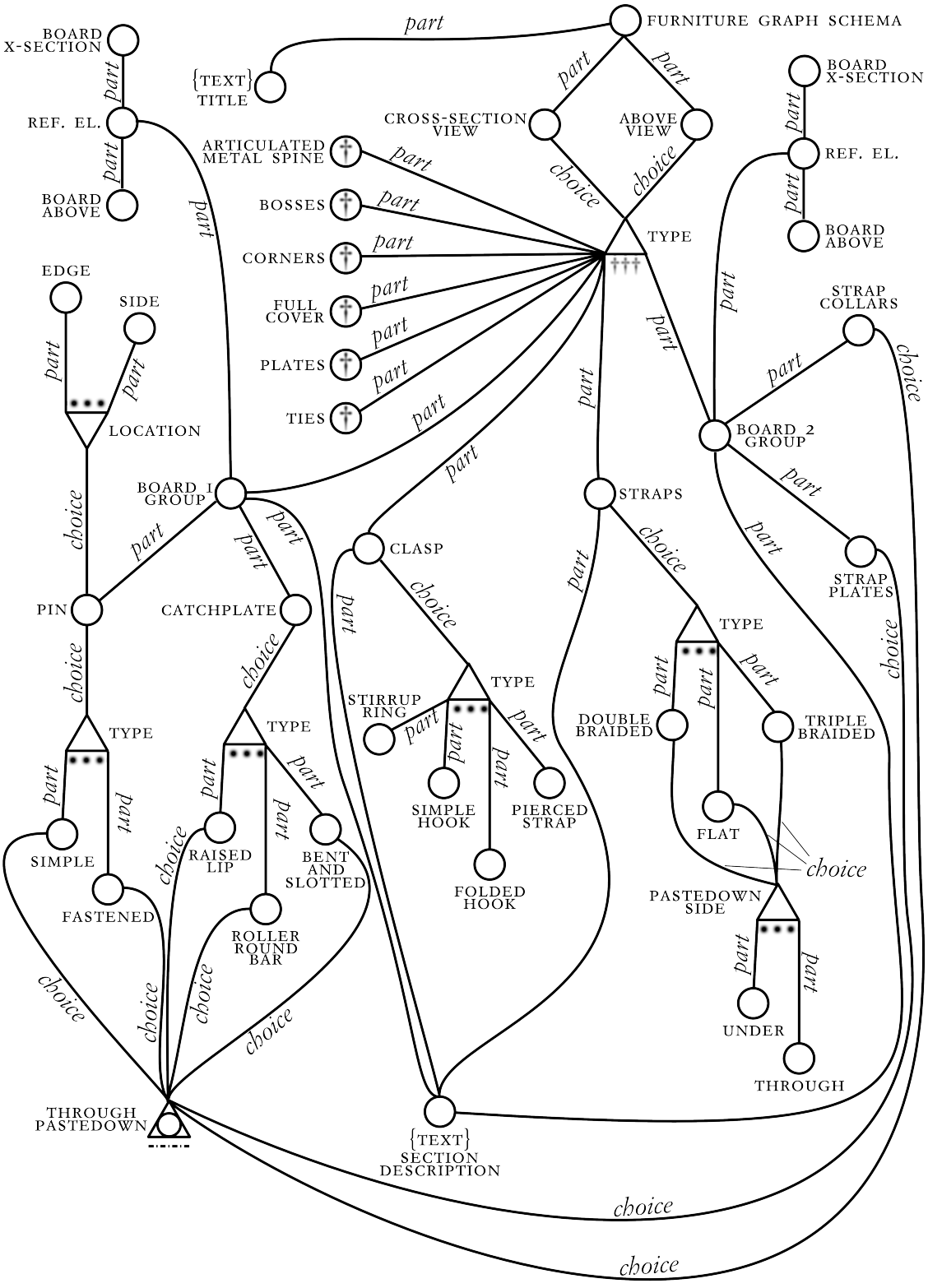
CASE - LONGSTITCH

OUTER VIEW

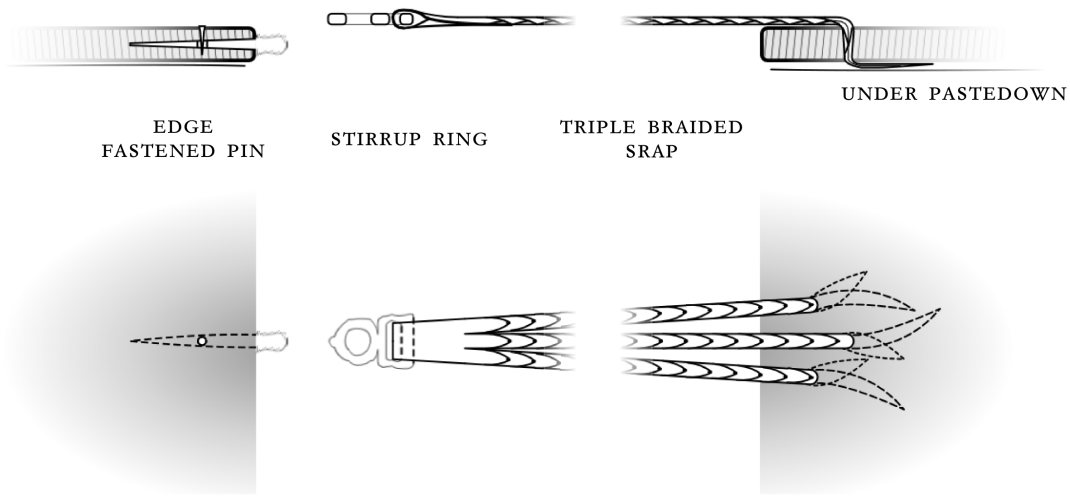


Furniture - fastenings

Furniture is the set of hardware that can be attached to a binding, usually with a protective, but also often decorative, function; this includes fastenings used to hold a book shut when not in use. Furniture include: (i) articulated metal spine, (ii) bosses, (iii) corners, (iv) full covers, (v) plates, (vi) ties, (vii) catchplates, (viii) clasps, (ix) pins, (x) straps, (xi) strap collars, (xii) and strap plates. Items *vii* to *xii* can be grouped as fastening parts. These are described in sufficient detail in the schema and their descriptions can be turned into useful visualizations. Items *i* to *vi*, instead, are only named by their generic name — e.g. boss or tie — leaving too much indeterminateness for the generation of meaningful visualizations. See §7.2.7.


FURNITURE GRAPH SCHEMA
{TEXT} TITLE
CROSS-SECTION VIEW
ABOVE VIEW
ARTICULATED METAL SPINE
BOSSES
CORNERS
FULL COVER
PLATES
TIES
TYPE
BOARD X-SECTION
REF. EL.
BOARD ABOVE
EDGE
SIDE
LOCATION
BOARD 1 GROUP
PIN
CATCHPLATE
SIMPLE
FASTENED
RAISED LIP
ROLLER ROUND BAR
BENT AND SLOTTED
THROUGH PASTEDOWN
CLASP
STIRRUP RING
SIMPLE HOOK
FOLDED HOOK
PIERCED STRAP
STRAPS
DOUBLE BRAIDED
TRIPLE BRAIDED
FLAT
PASTEDOWN SIDE
UNDER
THROUGH
STRAP COLLARS
STRAP PLATES
BOARD 2 GROUP
{TEXT} SECTION DESCRIPTION
part
choice

FURNITURE (FASTENINGS) - I

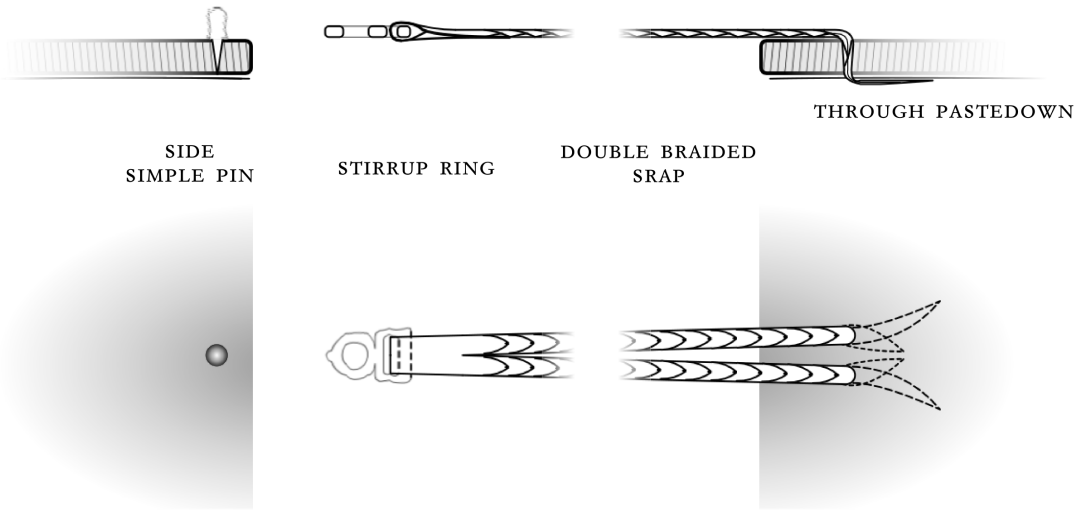


EDGE
FASTENED PIN

STIRRUP RING

TRIPLE BRAIDED
SRAP

UNDER PASTEDOWN



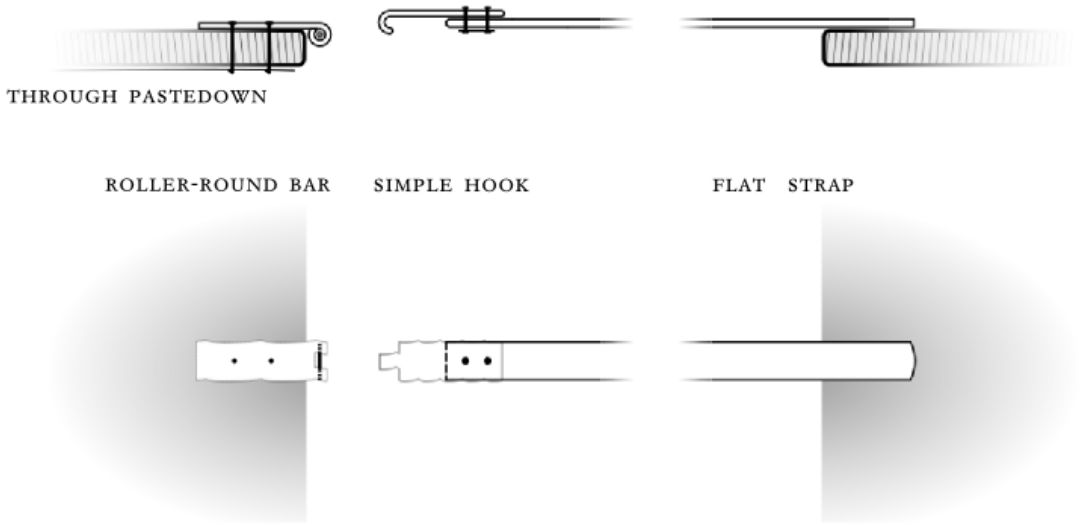
SIDE
SIMPLE PIN

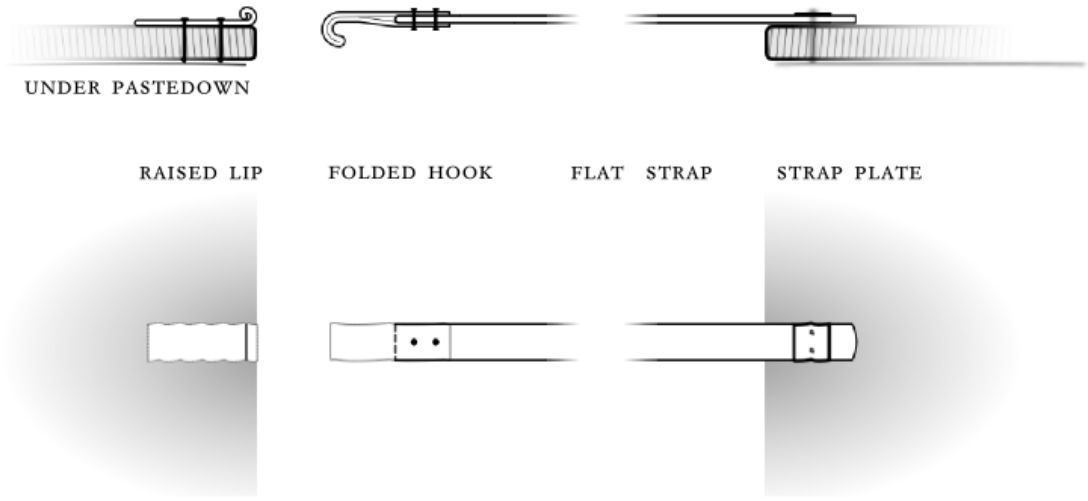
STIRRUP RING

DOUBLE BRAIDED
SRAP

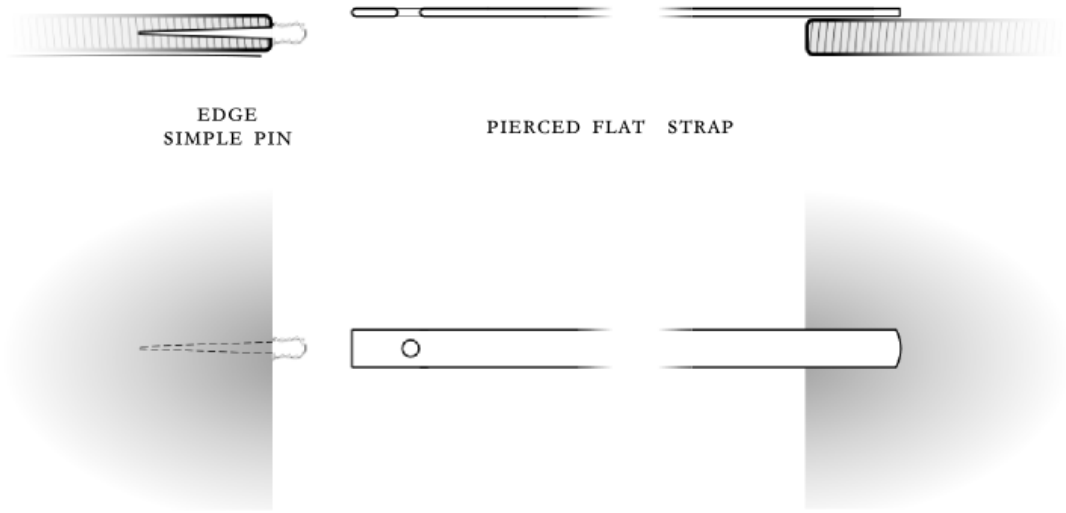
THROUGH PASTEDOWN

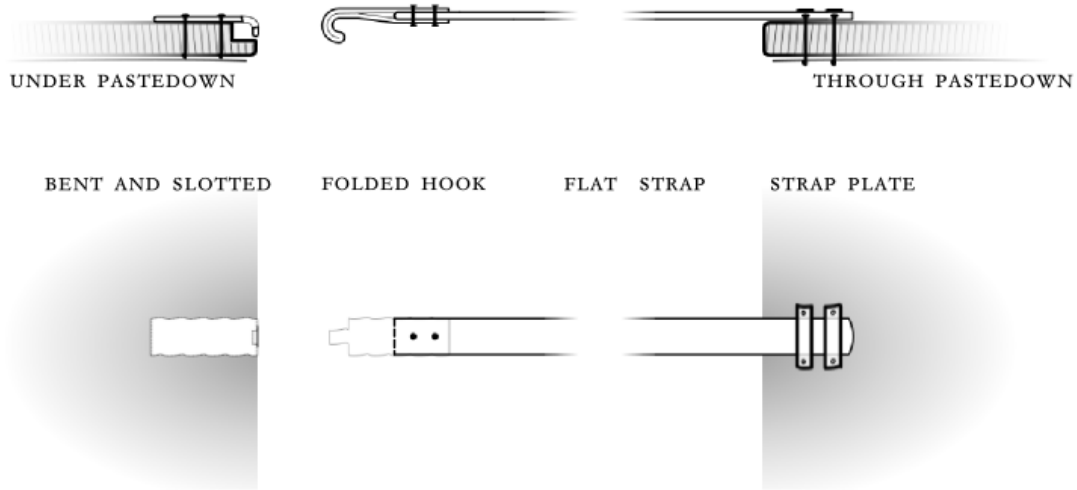
FURNITURE (FASTENINGS) - 2





FURNITURE (FASTENINGS) - 3





Appendix C. Coding example: endleaves

This appendix contains, as an example from the various transformation scenarios written for this project, the coding for endleaf structure visualizations.

For each transformation, there is an XSLT stylesheet that takes the information relative to a binding structure and generates an SVG visualization. The few elements of the visualizations that could be predefined and utilized where needed are defined in a master SVG file, from which they are called by the XSLT stylesheet; the SVG master also contains the code for the various filters applicable when needed to the diagram elements, e.g. to visualize uncertainty or imprecision. The appearance of lines and other graphical elements is ruled by classes defined within a Cascading Style Sheets (CSS)[506] file.

This appendix is subdivided into three sections: the XSLT stylesheet, the SVG master, and the CSS stylesheet for endleaf transformations. See §7.1.

506. CSS is a style sheet language used for describing the look and formatting of documents written in a markup language.Etemad 2011.

Endleaf XSLT

This section contains the XSLT coding written for this project to generate endleaf diagrams.

```
<?xml version="1.0" encoding="UTF-8"?>

<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
   xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:svg="http://www.w3.org/2000/svg"

   xmlns:lig="http://www.ligatus.org.uk/stcatherines/sites/ligatus.org.uk.stcather-
ines/files/basic-1.8_0.xsd"
    xmlns:xlink="http://www.w3.org/1999/xlink" xmlns:dict="www.mydict.my"
    exclude-result-prefixes="xs svg xlink lig dict" version="2.0">

   <xsl:output method="xml" indent="yes" encoding="utf-8" doctype-public="-//W3C//DTD
 SVG 1.1//EN"
          doctype-system="http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd" stan-
dalone="no"
         xpath-default-namespace="http://www.w3.org/2000/svg" exclude-result-pre-
fixes="xlink"
        include-content-type="no"/>


    <xsl:variable name="shelfmark" select="//bibliographical/shelfmark"/>
     <xsl:variable name="fileref" select="tokenize(replace($shelfmark, '/', '.'),
'\.')"/>
    <xsl:variable name="filenameLeft_structure"
        select="concat('../../Transformations/Endleaves/SVGoutput/',  $fileref[1],
 '/', $fileref[1], '_', 'leftEndleaves_structure', '.svg')"/>
    <xsl:variable name="filenameLeft_use"
        select="concat('../../Transformations/Endleaves/SVGoutput/',  $fileref[1],
 '/', $fileref[1], '_', 'leftEndleaves_use', '.svg')"/>
    <xsl:variable name="filenameRight_structure"
        select="concat('../../Transformations/Endleaves/SVGoutput/',  $fileref[1],
 '/', $fileref[1], '_', 'rightEndleaves_structure', '.svg')"/>
    <xsl:variable name="filenameRight_use"
        select="concat('../../Transformations/Endleaves/SVGoutput/',  $fileref[1],
 '/', $fileref[1], '_', 'rightEndleaves_use', '.svg')"/>


    <!-- X and Y reference values - i.e. the registration for the whole diagram,
changing these values, the whole diagram can be moved -->
    <xsl:param name="Ox" select="20"/>

    <xsl:param name="Oy"
             select="$Ox  +  $delta  *  (count(/book/endleaves/left/yes/type/separ-
ate/units/unit[1]/following-sibling::unit/components/component))"/>

    <!-- X and Y values to place the outermost gathering for both left and right
endleaves -->
    <xsl:variable name="Ax" select="$Ox + 155"/>
    <xsl:variable name="Ay" select="$Oy + 120"/>

   <!-- Value to determine the Y value of distance between the different components
 of the endleaves-->
    <xsl:variable name="delta" select="6"/>

    <xsl:template name="main" match="/">
        <xsl:choose>
            <xsl:when test="book/endleaves[left | right]">
                <xsl:for-each select="book/endleaves/left">
                  <!-- draw USE diagram (i.e. no differentiation between pastedowns
 or not -->
                     <xsl:result-document href="{$filenameLeft_use}" method="xml"
indent="yes"
                        encoding="utf-8" doctype-public="-//W3C//DTD SVG 1.1//EN"
                                        doctype-system="http://www.w3.org/Graph-
ics/SVG/1.1/DTD/svg11.dtd">
```

```
                        <xsl:processing-instruction name="xml-stylesheet">
                                <xsl:text>href="../../../../GitHub/XSLTransforma-
tions/Endleaves/CSS/style.css"&#32;</xsl:text>
                            <xsl:text>type="text/css"</xsl:text>
                    </xsl:processing-instruction>
                                <xsl:text>&#10;</xsl:text>
                                <xsl:comment>
                            <xsl:text>SVG file generated on: </xsl:text>
                             <xsl:value-of select="format-dateTime(current-dateTime(), '[D]
 [MNn] [Y] at [H]:[m]:[s]')"/>
                            <xsl:text> using </xsl:text>
                            <xsl:value-of select="system-property('xsl:product-name')"/>
                            <xsl:text> version </xsl:text>
                           <xsl:value-of select="system-property('xsl:product-version')"/>

                    </xsl:comment>
                                <xsl:text>&#10;</xsl:text>
                                  <svg xmlns="http://www.w3.org/2000/svg" xmlns:xlink="ht-
tp://www.w3.org/1999/xlink"
                                   version="1.1" x="0" y="0" width="420mm" height="297mm"
 viewBox="0 0 420 297"
                                  preserveAspectRatio="xMidYMid meet">
                                       <title>Left endleaves of book: <xsl:value-of se-
lect="$shelfmark"/></title>
                                  <xsl:copy-of
                                         select="document('../SVGmaster/EndleavesSVGmas-
ter.svg')/svg:svg/svg:defs"
                                    xpath-default-namespace="http://www.w3.org/2000/svg"
 copy-namespaces="no"/>
                                         <desc xmlns="http://www.w3.org/2000/svg">Left
endleaves</desc>
                                  <xsl:call-template name="title">
                                      <xsl:with-param name="side" select="'left'"/>
                                      <xsl:with-param name="use" select="'use'"/>
                                  </xsl:call-template>
                                  <xsl:variable name="unknown">
                                      <xsl:choose>
                                          <xsl:when test="./yes/type[NC | NK]">
                                              <xsl:text>ncNK</xsl:text>
                                          </xsl:when>
                                          <xsl:when test="./yes/type[other]">
                                              <xsl:text>other</xsl:text>
                                          </xsl:when>
                                      </xsl:choose>
                                  </xsl:variable>
                                  <xsl:call-template name="description">
                                      <xsl:with-param name="baseline"
                                             select="if (type/separate/units/unit/compon-
ents/component/type/hook/type/textHook)  then  $Ay  +  $delta  +  100  +  $delta  *
count(type/separate/units/unit/components/component[type/hook/type/textHook])  else
 $Ay + $delta + 100"/>
                                      <xsl:with-param name="unknown" select="$unknown"/>
                                  </xsl:call-template>
                                  <svg>
                                      <xsl:attribute name="x">
                                          <xsl:value-of select="$Ox"/>
                                      </xsl:attribute>
                                      <xsl:attribute name="y">
                                          <xsl:value-of select="$Oy"/>
                                      </xsl:attribute>
                                      <xsl:call-template name="leftEndleaves">
                                          <xsl:with-param name="use" select="false()"/>
                                      </xsl:call-template>
                                  </svg>
                              </svg>
```

```
                    </xsl:result-document>
                    <!-- draw STRUCTURE diagram (i.e. with differentiation between
 pastedowns or not -->
                  <xsl:result-document href="{$filenameLeft_structure}" method="xml"
 indent="yes"
                        encoding="utf-8" doctype-public="-//W3C//DTD SVG 1.1//EN"
                                        doctype-system="http://www.w3.org/Graph-
ics/SVG/1.1/DTD/svg11.dtd">
                        <xsl:processing-instruction name="xml-stylesheet">
                                 <xsl:text>href="../../../../GitHub/XSLTransforma-
tions/Endleaves/CSS/style.css"&#32;</xsl:text>
                    <xsl:text>type="text/css"</xsl:text>
                </xsl:processing-instruction>
                        <xsl:text>&#10;</xsl:text>
                        <xsl:comment>
                    <xsl:text>SVG file generated on: </xsl:text>
                     <xsl:value-of select="format-dateTime(current-dateTime(), '[D]
 [MNn] [Y] at [H]:[m]:[s]')"/>
                    <xsl:text> using </xsl:text>
                    <xsl:value-of select="system-property('xsl:product-name')"/>
                    <xsl:text> version </xsl:text>
                   <xsl:value-of select="system-property('xsl:product-version')"/>

                </xsl:comment>
                        <xsl:text>&#10;</xsl:text>
                          <svg xmlns="http://www.w3.org/2000/svg" xmlns:xlink="ht-
tp://www.w3.org/1999/xlink"
                           version="1.1" x="0" y="0" width="420mm" height="297mm"
 viewBox="0 0 420 297"
                           preserveAspectRatio="xMidYMid meet">
                                <title>Left endleaves of book: <xsl:value-of se-
lect="$shelfmark"/></title>
                           <xsl:copy-of
                                  select="document('../SVGmaster/EndleavesSVGmas-
ter.svg')/svg:svg/svg:defs"
                             xpath-default-namespace="http://www.w3.org/2000/svg"
 copy-namespaces="no"/>
                                  <desc xmlns="http://www.w3.org/2000/svg">Left
endleaves</desc>
                           <xsl:call-template name="title">
                               <xsl:with-param name="side" select="'left'"/>
                               <xsl:with-param name="use" select="'structure'"/>
                           </xsl:call-template>
                           <xsl:variable name="unknown">
                               <xsl:choose>
                                   <xsl:when test="./yes/type[NC | NK]">
                                       <xsl:text>ncNK</xsl:text>
                                   </xsl:when>
                                   <xsl:when test="./yes/type[other]">
                                       <xsl:text>other</xsl:text>
                                   </xsl:when>
                               </xsl:choose>
                           </xsl:variable>
                           <xsl:call-template name="description">
                               <xsl:with-param name="baseline"
                                      select="if (type/separate/units/unit/compon-
ents/component/type/hook/type/textHook)  then  $Ay  +  $delta  +  100  +  $delta  *
count(type/separate/units/unit/components/component[type/hook/type/textHook])  else
 $Ay + $delta + 100"/>
                               <xsl:with-param name="unknown" select="$unknown"/>
                           </xsl:call-template>
                           <svg>
                               <xsl:attribute name="x">
                                   <xsl:value-of select="$Ox"/>
                               </xsl:attribute>
```

```
                                <xsl:attribute name="y">
                                    <xsl:value-of select="$Oy"/>
                                </xsl:attribute>
                                <xsl:call-template name="leftEndleaves">
                                    <xsl:with-param name="use" select="true()"/>
                                </xsl:call-template>
                            </svg>
                        </svg>
                    </xsl:result-document>
                </xsl:for-each>
                <xsl:for-each select="book/endleaves/right">
                  <!-- draw USE diagram (i.e. no differentiation between pastedowns
 or not -->
                    <xsl:result-document href="{$filenameRight_use}" method="xml"
indent="yes"
                        encoding="utf-8" doctype-public="-//W3C//DTD SVG 1.1//EN"
                                    doctype-system="http://www.w3.org/Graph-
ics/SVG/1.1/DTD/svg11.dtd">
                        <xsl:processing-instruction name="xml-stylesheet">
                                 <xsl:text>href="../../../../GitHub/XSLTransforma-
tions/Endleaves/CSS/style.css"&#32;</xsl:text>
                    <xsl:text>type="text/css"</xsl:text>
                </xsl:processing-instruction>
                          <svg xmlns="http://www.w3.org/2000/svg" xmlns:xlink="ht-
tp://www.w3.org/1999/xlink"
                           version="1.1" x="0" y="0" width="420mm" height="297mm"
 viewBox="0 0 420 297"
                           preserveAspectRatio="xMidYMid meet">
                               <title>Right endleaves of book: <xsl:value-of se-
lect="$shelfmark"/></title>
                           <xsl:copy-of
                                  select="document('../SVGmaster/EndleavesSVGmas-
ter.svg')/svg:svg/svg:defs"
                             xpath-default-namespace="http://www.w3.org/2000/svg"
 copy-namespaces="no"/>
                           <xsl:call-template name="title">
                               <xsl:with-param name="side" select="'right'"/>
                               <xsl:with-param name="use" select="'use'"/>
                           </xsl:call-template>
                           <xsl:variable name="unknown">
                               <xsl:choose>
                                   <xsl:when test="./yes/type[NC | NK]">
                                       <xsl:text>ncNK</xsl:text>
                                   </xsl:when>
                                   <xsl:when test="./yes/type[other]">
                                       <xsl:text>other</xsl:text>
                                   </xsl:when>
                               </xsl:choose>
                           </xsl:variable>
                           <xsl:call-template name="description">
                               <xsl:with-param name="baseline"
                                     select="if (type/separate/units/unit/compon-
ents/component/type/hook/type/textHook) then $Ay + $delta + 100 + $delta *
count(type/separate/units/unit/components/component[type/hook/type/textHook]) else
 $Ay + $delta + 100"/>
                               <xsl:with-param name="unknown" select="$unknown"/>
                           </xsl:call-template>
                           <g transform="scale(-1 1)">
                               <desc>Right endleaves</desc>
                               <svg>
                                   <xsl:attribute name="x">
                                       <xsl:value-of select="$Ox - 365"/>
                                   </xsl:attribute>
                                   <xsl:attribute name="y">
                                       <xsl:value-of select="$Oy"/>
```

```
                                    </xsl:attribute>
                                    <xsl:call-template name="leftEndleaves">
                                      <xsl:with-param name="use" select="false()"/>

                                    </xsl:call-template>
                                </svg>
                            </g>
                        </svg>
                    </xsl:result-document>
                    <!-- draw STRUCTURE diagram (i.e. with differentiation between
 pastedowns or not -->
                      <xsl:result-document href="{$filenameRight_structure}" meth-
od="xml" indent="yes"
                        encoding="utf-8" doctype-public="-//W3C//DTD SVG 1.1//EN"
                                        doctype-system="http://www.w3.org/Graph-
ics/SVG/1.1/DTD/svg11.dtd">
                        <xsl:processing-instruction name="xml-stylesheet">
                                 <xsl:text>href="../../../../GitHub/XSLTransforma-
tions/Endleaves/CSS/style.css"&#32;</xsl:text>
                    <xsl:text>type="text/css"</xsl:text>
                </xsl:processing-instruction>
                          <svg xmlns="http://www.w3.org/2000/svg" xmlns:xlink="ht-
tp://www.w3.org/1999/xlink"
                           version="1.1" x="0" y="0" width="420mm" height="297mm"
 viewBox="0 0 420 297"
                           preserveAspectRatio="xMidYMid meet">
                               <title>Right endleaves of book: <xsl:value-of se-
lect="$shelfmark"/></title>
                           <xsl:copy-of
                                  select="document('../SVGmaster/EndleavesSVGmas-
ter.svg')/svg:svg/svg:defs"
                             xpath-default-namespace="http://www.w3.org/2000/svg"
 copy-namespaces="no"/>
                           <xsl:call-template name="title">
                               <xsl:with-param name="side" select="'right'"/>
                               <xsl:with-param name="use" select="'structure'"/>
                           </xsl:call-template>
                           <xsl:variable name="unknown">
                               <xsl:choose>
                                   <xsl:when test="./yes/type[NC | NK]">
                                       <xsl:text>ncNK</xsl:text>
                                   </xsl:when>
                                   <xsl:when test="./yes/type[other]">
                                       <xsl:text>other</xsl:text>
                                   </xsl:when>
                               </xsl:choose>
                           </xsl:variable>
                           <xsl:call-template name="description">
                               <xsl:with-param name="baseline"
                                      select="if (type/separate/units/unit/compon-
ents/component/type/hook/type/textHook) then $Ay + $delta + 100 + $delta *
count(type/separate/units/unit/components/component[type/hook/type/textHook]) else
 $Ay + $delta + 100"/>
                               <xsl:with-param name="unknown" select="$unknown"/>
                           </xsl:call-template>
                           <g transform="scale(-1 1)">
                               <desc>Right endleaves</desc>
                               <svg>
                                   <xsl:attribute name="x">
                                       <xsl:value-of select="$Ox - 365"/>
                                   </xsl:attribute>
                                   <xsl:attribute name="y">
                                       <xsl:value-of select="$Oy"/>
                                   </xsl:attribute>
                                   <xsl:call-template name="leftEndleaves">
```

```
                                        <xsl:with-param name="use" select="true()"/>

                                    </xsl:call-template>
                                </svg>
                            </g>
                        </svg>
                    </xsl:result-document>
                </xsl:for-each>
            </xsl:when>
            <xsl:otherwise>
                <svg xmlns="http://www.w3.org/2000/svg">
                    <xsl:call-template name="title">
                        <xsl:with-param name="detected" select="0"/>
                    </xsl:call-template>
                </svg>
            </xsl:otherwise>
        </xsl:choose>
    </xsl:template>

    <xsl:template name="leftEndleaves">
        <xsl:param name="use" select="false()"/>
        <xsl:choose>
            <!-- If endleaves are present, then the right sequence of templates is
 called to construct the diagram, otherwise only the outer gathering is drawn -->
            <xsl:when test="self::node()[yes | no]">
                <g xmlns="http://www.w3.org/2000/svg">
                    <xsl:choose>
                     <xsl:when test="./yes/type/separate/units/unit//hook/type[text-
Hook]">
                            <!-- shorter outermost gathering -->
                            <path xmlns="http://www.w3.org/2000/svg">
                                <xsl:choose>
                                    <xsl:when test="./yes/type[integral]">
                                        <xsl:attribute name="class">
                                            <xsl:text>line</xsl:text>
                                        </xsl:attribute>
                                    </xsl:when>
                                    <xsl:otherwise>
                                        <xsl:attribute name="class">
                                            <xsl:text>line_ref</xsl:text>
                                        </xsl:attribute>
                                    </xsl:otherwise>
                                </xsl:choose>
                                <xsl:attribute name="d">
                                    <xsl:text>M</xsl:text>
                                    <xsl:value-of select="$Ax + 140"/>
                                    <xsl:text>,</xsl:text>
                                    <xsl:value-of select="$Ay"/>
                                    <xsl:text>&#32;L</xsl:text>
                                    <xsl:value-of select="$Ax + $delta + 10"/>
                                    <xsl:text>,</xsl:text>
                                    <xsl:value-of select="$Ay"/>
                                    <xsl:text>&#32;A</xsl:text>
                                    <xsl:value-of select="10"/>
                                    <xsl:text>,</xsl:text>
                                    <xsl:value-of select="10"/>
                                    <xsl:text>&#32;</xsl:text>
                                    <xsl:text>0</xsl:text>
                                    <xsl:text>&#32;</xsl:text>
                                    <xsl:text>0</xsl:text>
                                    <xsl:text>,</xsl:text>
                                    <xsl:text>0</xsl:text>
                                    <xsl:value-of select="$Ax + $delta + 10"/>
                                    <xsl:text>,</xsl:text>
                                    <xsl:value-of select="$Ay + 20"/>
```

```
                                    <xsl:text>&#32;L</xsl:text>
                                    <xsl:value-of select="$Ax + 140"/>
                                    <xsl:text>,</xsl:text>
                                    <xsl:value-of select="$Ay + 20"/>
                                    <xsl:text>&#32;z</xsl:text>
                                </xsl:attribute>
                            </path>
                        </xsl:when>
                        <xsl:otherwise>
                            <use xlink:href="#outermostGL">
                                <xsl:choose>
                                    <xsl:when test="./yes/type[integral]">
                                        <xsl:attribute name="class">
                                            <xsl:text>line</xsl:text>
                                        </xsl:attribute>
                                    </xsl:when>
                                    <xsl:otherwise>
                                        <xsl:attribute name="class">
                                            <xsl:text>line_ref</xsl:text>
                                        </xsl:attribute>
                                    </xsl:otherwise>
                                </xsl:choose>
                                <xsl:attribute name="x">
                                    <xsl:value-of select="$Ax"/>
                                </xsl:attribute>
                                <xsl:attribute name="y">
                                    <xsl:value-of select="$Ay"/>
                                </xsl:attribute>
                            </use>
                            <use xlink:href="#outermostGL">
                                <xsl:attribute name="class">
                                    <xsl:text>line_ref</xsl:text>
                                </xsl:attribute>
                                <xsl:attribute name="x">
                                    <xsl:value-of select="$Ax"/>
                                </xsl:attribute>
                                <xsl:attribute name="y">
                                    <xsl:value-of select="$Ay + $delta + 20"/>
                                </xsl:attribute>
                            </use>
                        </xsl:otherwise>
                    </xsl:choose>
                </g>
                <xsl:choose>
                    <xsl:when test="./yes/type[integral]">
                        <xsl:call-template name="leftEndleavesIntegral">
                            <xsl:with-param name="use" select="$use"/>
                        </xsl:call-template>
                    </xsl:when>
                </xsl:choose>
                <xsl:choose>
                    <xsl:when test="./yes/type[separate]">
                        <xsl:call-template name="leftEndleavesSeparate">
                            <xsl:with-param name="use" select="$use"/>
                        </xsl:call-template>
                    </xsl:when>
                    <xsl:otherwise>
                        <desc xmlns="http://www.w3.org/2000/svg">Type of endleaves
not checked, not
                            known, or other</desc>
                    </xsl:otherwise>
                </xsl:choose>
            </xsl:when>
        </xsl:choose>
    </xsl:template>
```

```
    <xsl:template name="leftEndleavesIntegral">
        <xsl:param name="use" select="false()" as="xs:boolean"/>
          <xsl:param name="totLeaves" select="./yes/type/integral/numberOfLeaves"
as="xs:integer"/>
        <xsl:param name="currentLeaf" select="1"/>
        <xsl:variable name="baseline_int" select="$Ay"/>
        <xsl:variable name="B1x" select="$Ax - 145"/>
        <xsl:variable name="B1y"
            select="$baseline_int - ($delta * $totLeaves) - ($delta * ($totLeaves
- $currentLeaf)) - 5 -
                      (if  (./yes/type/separate//pastedown/yes)  then  ($delta  *
(count(./yes/type/separate//pastedown/yes))) else 0)"/>
        <desc xmlns="http://www.w3.org/2000/svg">Integral endleaves</desc>
        <desc xmlns="http://www.w3.org/2000/svg">Leaf N.<xsl:value-of
                select="$totLeaves - $currentLeaf + 1"/></desc>
        <xsl:choose>
           <xsl:when test="./yes/type/integral/pastedown[yes] and $use eq false()">

                <xsl:call-template name="leftEndleavesIntegral-Pastedown">
                   <xsl:with-param name="totLeaves" select="$totLeaves" as="xs:in-
teger"/>
                    <xsl:with-param name="currentLeaf" select="$currentLeaf"/>
                    <xsl:with-param name="baseline_int" select="$baseline_int"/>
                    <xsl:with-param name="B1x" select="$B1x"/>
                    <xsl:with-param name="B1y" select="$B1y"/>
                </xsl:call-template>
            </xsl:when>
           <xsl:when test="./yes/type/integral/pastedown[yes] and $use eq true()">

                <xsl:call-template name="leftEndleavesIntegral-Flyleaves">
                   <xsl:with-param name="totLeaves" select="$totLeaves" as="xs:in-
teger"/>
                    <xsl:with-param name="currentLeaf" select="$currentLeaf"/>
                    <xsl:with-param name="baseline_int" select="$baseline_int"/>
                    <xsl:with-param name="B1x" select="$B1x"/>
                    <xsl:with-param name="B1y" select="$B1y"/>
                    <xsl:with-param name="use" select="$use"/>
                </xsl:call-template>
            </xsl:when>
            <xsl:when test="./yes/type/integral/pastedown[no]">
                <xsl:call-template name="leftEndleavesIntegral-Flyleaves">
                   <xsl:with-param name="totLeaves" select="$totLeaves" as="xs:in-
teger"/>
                    <xsl:with-param name="currentLeaf" select="$currentLeaf"/>
                    <xsl:with-param name="baseline_int" select="$baseline_int"/>
                    <xsl:with-param name="B1x" select="$B1x"/>
                    <xsl:with-param name="B1y" select="$B1y"/>
                    <xsl:with-param name="use" select="$use"/>
                </xsl:call-template>
            </xsl:when>
            <xsl:otherwise>
                  <desc xmlns="http://www.w3.org/2000/svg">Integral flyleaves not
checked, not known,
                    or other.</desc>
                <!-- Draw as if no, but with uncertainty -->
                <xsl:call-template name="leftEndleavesIntegral-Flyleaves">
                   <xsl:with-param name="totLeaves" select="$totLeaves" as="xs:in-
teger"/>
                    <xsl:with-param name="currentLeaf" select="$currentLeaf"/>
                    <xsl:with-param name="baseline_int" select="$baseline_int"/>
                    <xsl:with-param name="B1x" select="$B1x"/>
                    <xsl:with-param name="B1y" select="$B1y"/>
                     <xsl:with-param name="certainty" select="if ($use eq true())
then 100 else 50"/>
```

```
                    <xsl:with-param name="use" select="$use"/>
                </xsl:call-template>
            </xsl:otherwise>
        </xsl:choose>
    </xsl:template>

    <xsl:template name="leftEndleavesIntegral-Pastedown">
        <xsl:param name="totLeaves" as="xs:integer"/>
        <xsl:param name="currentLeaf"/>
        <xsl:param name="baseline_int"/>
        <xsl:param name="B1x"/>
        <xsl:param name="B1y"/>
        <xsl:choose>
            <xsl:when test="$currentLeaf = $totLeaves">
                <desc xmlns="http://www.w3.org/2000/svg">Pastedown</desc>
                <g xmlns="http://www.w3.org/2000/svg">
                    <use xlink:href="#pastedown">
                        <xsl:attribute name="x">
                            <xsl:value-of select="$B1x"/>
                        </xsl:attribute>
                        <xsl:attribute name="y">
                            <xsl:value-of select="$B1y"/>
                        </xsl:attribute>
                    </use>
                    <path>
                        <xsl:attribute name="class">
                            <xsl:text>line</xsl:text>
                        </xsl:attribute>
                        <xsl:attribute name="d">
                            <xsl:text>M</xsl:text>
                            <xsl:value-of select="$Ax"/>
                            <xsl:text>,</xsl:text>
                            <xsl:value-of select="$baseline_int + 10"/>
                            <xsl:text>&#32;Q</xsl:text>
                            <xsl:value-of select="$Ax"/>
                            <xsl:text>,</xsl:text>
                            <xsl:value-of select="$B1y + $delta"/>
                            <xsl:text>&#32;</xsl:text>
                            <xsl:value-of select="$B1x + 130"/>
                            <xsl:text>,</xsl:text>
                            <xsl:value-of select="$B1y"/>
                        </xsl:attribute>
                    </path>
                </g>
            </xsl:when>
            <xsl:when test="$currentLeaf lt $totLeaves">
                <desc xmlns="http://www.w3.org/2000/svg">Flyleaf</desc>
                <xsl:call-template name="leftEndleavesIntegral-Flyleaf">
                   <xsl:with-param name="totLeaves" select="$totLeaves" as="xs:in-
teger"/>
                    <xsl:with-param name="currentLeaf" select="$currentLeaf"/>
                    <xsl:with-param name="B1x" select="$B1x"/>
                    <xsl:with-param name="B1y" select="$B1y"/>
                    <xsl:with-param name="baseline_int" select="$baseline_int"/>
                </xsl:call-template>
                <xsl:call-template name="leftEndleavesIntegral">
                   <xsl:with-param name="totLeaves" select="$totLeaves" as="xs:in-
teger"/>
                    <xsl:with-param name="currentLeaf" select="$currentLeaf + 1"/>
                </xsl:call-template>
            </xsl:when>
        </xsl:choose>
    </xsl:template>

    <xsl:template name="leftEndleavesIntegral-Flyleaves">
```

```
        <xsl:param name="use" select="false()"/>
        <xsl:param name="totLeaves" as="xs:integer"/>
        <xsl:param name="currentLeaf"/>
        <xsl:param name="baseline_int"/>
        <xsl:param name="B1x"/>
        <xsl:param name="B1y"/>
        <xsl:param name="certainty" select="100"/>
        <xsl:choose>
            <xsl:when test="$currentLeaf = $totLeaves">
                <desc xmlns="http://www.w3.org/2000/svg">Flyleaf</desc>
                <xsl:call-template name="leftEndleavesIntegral-Flyleaf">
                   <xsl:with-param name="totLeaves" select="$totLeaves" as="xs:in-
teger"/>
                    <xsl:with-param name="currentLeaf" select="$currentLeaf"/>
                    <xsl:with-param name="B1x" select="$B1x"/>
                    <xsl:with-param name="B1y" select="$B1y"/>
                    <xsl:with-param name="baseline_int" select="$baseline_int"/>
                    <xsl:with-param name="certainty" select="$certainty"/>
                </xsl:call-template>
            </xsl:when>
            <xsl:when test="$currentLeaf lt $totLeaves">
                <desc xmlns="http://www.w3.org/2000/svg">Flyleaf</desc>
                <xsl:call-template name="leftEndleavesIntegral-Flyleaf">
                   <xsl:with-param name="totLeaves" select="$totLeaves" as="xs:in-
teger"/>
                    <xsl:with-param name="currentLeaf" select="$currentLeaf"/>
                    <xsl:with-param name="B1x" select="$B1x"/>
                    <xsl:with-param name="B1y" select="$B1y"/>
                    <xsl:with-param name="baseline_int" select="$baseline_int"/>
                    <xsl:with-param name="certainty" select="$certainty"/>
                </xsl:call-template>
                <xsl:call-template name="leftEndleavesIntegral">
                   <xsl:with-param name="totLeaves" select="$totLeaves" as="xs:in-
teger"/>
                    <xsl:with-param name="currentLeaf" select="$currentLeaf + 1"/>
                    <xsl:with-param name="use" select="$use"/>
                </xsl:call-template>
            </xsl:when>
        </xsl:choose>
    </xsl:template>

    <xsl:template name="leftEndleavesIntegral-Flyleaf">
        <xsl:param name="totLeaves"/>
        <xsl:param name="currentLeaf"/>
        <xsl:param name="baseline_int"/>
        <xsl:param name="B1x"/>
        <xsl:param name="B1y"/>
        <xsl:param name="certainty" select="100" as="xs:integer"/>
        <xsl:variable name="Ax"
                      select="if  (./yes/type/separate/units/unit/components/compon-
ent/type/hook/type/textHook) then $Ax + 6 else $Ax"/>
        <xsl:variable name="Ax2"
                      select="if  (./yes/type/separate/units/unit/components/compon-
ent/type/hook/type/textHook) then $Ax + 134 else $Ax + 140"/>
        <g xmlns="http://www.w3.org/2000/svg">
            <path>
                <xsl:attribute name="class">
                    <xsl:text>line</xsl:text>
                </xsl:attribute>
                <xsl:call-template name="certainty">
                    <xsl:with-param name="certainty" select="$certainty"/>
                    <xsl:with-param name="type" select="'2'"/>
                </xsl:call-template>
                <xsl:attribute name="d">
                    <xsl:text>M</xsl:text>
```

```
                    <xsl:value-of select="$Ax"/>
                    <xsl:text>,</xsl:text>
                    <xsl:value-of select="$baseline_int + 10"/>
                    <xsl:text>&#32;Q</xsl:text>
                    <xsl:value-of select="$Ax"/>
                    <xsl:text>,</xsl:text>
                 <xsl:value-of select="($baseline_int - ($delta * $currentLeaf))"/>

                    <xsl:text>&#32;</xsl:text>
                    <xsl:value-of select="$Ax + 10"/>
                    <xsl:text>,</xsl:text>
                 <xsl:value-of select="($baseline_int - ($delta * $currentLeaf))"/>

                    <xsl:text>&#32;L</xsl:text>
                    <xsl:value-of select="$Ax2"/>
                    <xsl:text>,</xsl:text>
                 <xsl:value-of select="($baseline_int - ($delta * $currentLeaf))"/>

                </xsl:attribute>
            </path>
        </g>
    </xsl:template>

    <xsl:template name="leftEndleavesSeparate">
        <xsl:param name="use" select="false()" as="xs:boolean"/>
        <desc xmlns="http://www.w3.org/2000/svg">Separate endleaves</desc>
        <xsl:for-each select="./yes/type/separate/units/unit">
            <!-- Variable to count the number of Units -->
            <xsl:variable name="countUnits" select="last()"/>
            <!-- Counter variable for the current unit -->
            <xsl:variable name="currentUnit" select="position()"/>
               <desc xmlns="http://www.w3.org/2000/svg">Unit N. <xsl:value-of se-
lect="$currentUnit"
                /></desc>
            <xsl:call-template name="leftEndleavesSeparate_components">
                <xsl:with-param name="use" select="$use"/>
                <xsl:with-param name="countUnits" select="$countUnits"/>
                <xsl:with-param name="currentUnit" select="$currentUnit"/>
                <xsl:with-param name="baseline"
                         select="if (following-sibling::unit[1]/components/compon-
ent//type[textHook]) then $Ay - $delta * count(following-sibling::unit[1]/compon-
ents/component)
                   - (if (ancestor::yes/type/integral) then $delta * count(ancest-
or::yes/type/integral/xs:integer(numberOfLeaves)) + $delta * 1.5 else 0)
                     else $Ay - ((if (ancestor::yes/type[integral]) then $delta *
ancestor::yes/type/integral/xs:integer(numberOfLeaves) else 0)
                 + (if (preceding-sibling::unit[1]/components/component//type[text-
Hook]) then $delta * count(preceding-sibling::unit[1]/components/component) else 0)

                    + (if (components/component//type[textHook]) then (if (ancest-
or::yes/type/integral) then - $delta * ancestor::yes/type/integral/numberOfLeaves
else 0)
                  else $delta * 2 * count(following-sibling::unit/components/com-
ponent)
                     + (($delta - 2) * count(following-sibling::unit[not(compon-
ents//type[textHook])]))
                           - (if (following-sibling::unit[1]/components/compon-
ent/type[singleLeaf | NC | NK | other]) then $delta * 1.5 else 0)
                  ))"
                />
            </xsl:call-template>
        </xsl:for-each>
    </xsl:template>

    <xsl:template name="leftEndleavesSeparate_components">
```

```
        <xsl:param name="use" select="false()" as="xs:boolean"/>
        <xsl:param name="countUnits"/>
        <xsl:param name="currentUnit"/>
        <xsl:param name="baseline"/>
        <xsl:for-each select="./components/component">
            <!-- Variable to count the number of Components -->
            <xsl:variable name="countComponents" select="last()"/>
            <!-- Counter variable for the current component -->
            <xsl:variable name="currentComponent" select="position()"/>
            <g xmlns="http://www.w3.org/2000/svg">
                <xsl:attribute name="class">
                    <xsl:choose>
                        <xsl:when test="material/parchment">
                            <xsl:text>line2</xsl:text>
                        </xsl:when>
                        <xsl:otherwise>
                            <xsl:text>line</xsl:text>
                        </xsl:otherwise>
                    </xsl:choose>
                </xsl:attribute>
               <desc xmlns="http://www.w3.org/2000/svg">Component N. <xsl:value-of

                        select="$currentComponent"/></desc>
                <xsl:choose>
                    <xsl:when test="./type/hook/type[textHook]">
                       <use xmlns="http://www.w3.org/2000/svg" xlink:href="#outer-
mostGL">
                            <xsl:attribute name="class">
                                <xsl:text>line_ref</xsl:text>
                            </xsl:attribute>
                            <xsl:attribute name="x">
                                <xsl:value-of select="$Ax"/>
                            </xsl:attribute>
                            <xsl:attribute name="y">
                                 <xsl:value-of select="$Ay + $delta + 20 + $delta *
 $countComponents"
                                 />
                            </xsl:attribute>
                        </use>
                    </xsl:when>
                </xsl:choose>
                <xsl:choose>
                    <xsl:when test="./pastedown[yes] and $use eq false()">
                        <desc xmlns="http://www.w3.org/2000/svg">Pastedown</desc>
                        <xsl:call-template name="leftEndleavesSeparatePastedown">
                            <xsl:with-param name="use" select="$use"/>
                             <xsl:with-param name="countComponents" select="$count-
Components"/>
                          <xsl:with-param name="currentComponent" select="$current-
Component"/>
                            <xsl:with-param name="baseline" select="$baseline"/>
                        </xsl:call-template>
                    </xsl:when>
                    <xsl:when test="./pastedown[yes] and $use eq true()">
                        <desc xmlns="http://www.w3.org/2000/svg">Flyleaves</desc>
                        <xsl:call-template name="leftEndleavesSeparateFlyleaves">
                            <xsl:with-param name="use" select="$use"/>
                            <xsl:with-param name="baseline" select="$baseline"/>
                             <xsl:with-param name="countComponents" select="$count-
Components"/>
                          <xsl:with-param name="currentComponent" select="$current-
Component"/>
                        </xsl:call-template>
                    </xsl:when>
                    <xsl:when test="./pastedown[no]">
```

```
                        <desc xmlns="http://www.w3.org/2000/svg">Flyleaves</desc>
                        <xsl:call-template name="leftEndleavesSeparateFlyleaves">
                            <xsl:with-param name="use" select="$use"/>
                            <xsl:with-param name="baseline" select="$baseline"/>
                             <xsl:with-param name="countComponents" select="$count-
Components"/>
                             <xsl:with-param name="currentComponent" select="$current-
Component"/>
                        </xsl:call-template>
                    </xsl:when>
                    <xsl:otherwise>
                         <desc xmlns="http://www.w3.org/2000/svg">Type of endleaf
component not
                            checked, not known, or other</desc>
                        <xsl:call-template name="leftEndleavesSeparateFlyleaves">
                            <xsl:with-param name="use" select="$use"/>
                            <xsl:with-param name="baseline" select="$baseline"/>
                             <xsl:with-param name="countComponents" select="$count-
Components"/>
                             <xsl:with-param name="currentComponent" select="$current-
Component"/>
                            <xsl:with-param name="certainty"
                                select="if ($use eq true()) then 100 else 50"/>
                        </xsl:call-template>
                    </xsl:otherwise>
                </xsl:choose>
            </g>
        </xsl:for-each>
    </xsl:template>


    <xsl:template name="leftEndleavesSeparateFlyleaves">
        <xsl:param name="use" select="false()" as="xs:boolean"/>
        <xsl:param name="baseline"/>
        <xsl:param name="countComponents"/>
        <xsl:param name="currentComponent"/>
        <xsl:param name="certainty" select="100" as="xs:integer"/>
        <xsl:choose>
            <xsl:when test="./type[fold]">
                <desc xmlns="http://www.w3.org/2000/svg">Type fold</desc>
                <xsl:call-template name="leftEndleavesSeparateFlyleaves-Fold">
                    <xsl:with-param name="use" select="$use"/>
                    <xsl:with-param name="baseline" select="$baseline"/>
                 <xsl:with-param name="countComponents" select="$countComponents"/>

                   <xsl:with-param name="currentComponent" select="$currentCompon-
ent"/>
                    <xsl:with-param name="certainty" select="$certainty"/>
                </xsl:call-template>
            </xsl:when>
            <xsl:when test="./type[guard]">
                <desc xmlns="http://www.w3.org/2000/svg">Type guard</desc>
                <xsl:call-template name="leftEndleavesSeparateFlyleaves-Guard">
                    <xsl:with-param name="use" select="$use"/>
                    <xsl:with-param name="baseline" select="$baseline"/>
                 <xsl:with-param name="countComponents" select="$countComponents"/>

                   <xsl:with-param name="currentComponent" select="$currentCompon-
ent"/>
                </xsl:call-template>
            </xsl:when>
            <xsl:when test="./type[hook]">
                <desc xmlns="http://www.w3.org/2000/svg">Type hook</desc>
                <xsl:choose>
                    <xsl:when test="./type/hook/type[endleafHook]">
```

```
                            <desc xmlns="http://www.w3.org/2000/svg">Type endleaf-
hook</desc>
                           <xsl:call-template name="leftEndleavesSeparateFlyleaves-
EndleafHook">
                            <xsl:with-param name="use" select="$use"/>
                            <xsl:with-param name="baseline" select="$baseline"/>
                             <xsl:with-param name="countComponents" select="$count-
Components"/>
                           <xsl:with-param name="currentComponent" select="$current-
Component"/>
                         </xsl:call-template>
                     </xsl:when>
                     <xsl:when test="./type/hook/type[textHook]">
                      <desc xmlns="http://www.w3.org/2000/svg">Type text-hook</desc>

                           <xsl:call-template name="leftEndleavesSeparateFlyleaves-
TextHook">
                            <xsl:with-param name="use" select="$use"/>
                            <xsl:with-param name="baseline" select="$baseline"/>
                             <xsl:with-param name="countComponents" select="$count-
Components"/>
                           <xsl:with-param name="currentComponent" select="$current-
Component"/>
                            <xsl:with-param name="certainty" select="$certainty"/>
                         </xsl:call-template>
                     </xsl:when>
                     <xsl:otherwise>
                         <desc xmlns="http://www.w3.org/2000/svg">Type not checked,
 not known, or
                             other type</desc>
                     </xsl:otherwise>
                 </xsl:choose>
             </xsl:when>
             <xsl:when test="./type[outsideHook]">
                 <desc xmlns="http://www.w3.org/2000/svg">Type outside hook</desc>
               <xsl:call-template name="leftEndleavesSeparateFlyleaves-OutsideHook">

                     <xsl:with-param name="use" select="$use"/>
                     <xsl:with-param name="baseline" select="$baseline"/>
                  <xsl:with-param name="countComponents" select="$countComponents"/>

                    <xsl:with-param name="currentComponent" select="$currentCompon-
ent"/>
                     <xsl:with-param name="certainty" select="$certainty"/>
                 </xsl:call-template>
             </xsl:when>
             <xsl:when test="./type[singleLeaf]">
                 <desc xmlns="http://www.w3.org/2000/svg">Type single leaf</desc>
                 <xsl:call-template name="leftEndleavesSeparateFlyleaf-SingleLeaf">
                     <xsl:with-param name="use" select="$use"/>
                     <xsl:with-param name="baseline" select="$baseline"/>
                  <xsl:with-param name="countComponents" select="$countComponents"/>

                    <xsl:with-param name="currentComponent" select="$currentCompon-
ent"/>
                    <xsl:with-param name="certainty" select="$certainty" as="xs:in-
teger"/>
                 </xsl:call-template>
                 <xsl:call-template name="componentAttachment">
                     <xsl:with-param name="use" select="$use"/>
                     <xsl:with-param name="baseline"
                          select="if (attachment/glued) then $baseline - ($delta *
$currentComponent) + ($delta) else $baseline"/>
                  <xsl:with-param name="countComponents" select="$countComponents"/>
```

```
                </xsl:call-template>
            </xsl:when>
            <xsl:otherwise>
              <desc xmlns="http://www.w3.org/2000/svg">Type not checked, not known,
 or other
                  type</desc>
                <!-- draw single leaf that fades away at the fold-->
                <xsl:call-template name="leftEndleavesSeparateFlyleaf-SingleLeaf">
                    <xsl:with-param name="use" select="$use"/>
                    <xsl:with-param name="baseline" select="$baseline"/>
                 <xsl:with-param name="countComponents" select="$countComponents"/>

                   <xsl:with-param name="currentComponent" select="$currentCompon-
ent"/>
                   <xsl:with-param name="certainty" select="$certainty" as="xs:in-
teger"/>
                    <xsl:with-param name="unknown" select="'yes'"/>
                </xsl:call-template>
                <xsl:call-template name="componentAttachment">
                    <xsl:with-param name="use" select="$use"/>
                    <xsl:with-param name="certainty" select="50" as="xs:integer"/>
                    <xsl:with-param name="baseline"
                         select="if (attachment/glued) then $baseline - ($delta *
$currentComponent) + ($delta) else $baseline"/>
                 <xsl:with-param name="countComponents" select="$countComponents"/>

                </xsl:call-template>
            </xsl:otherwise>
        </xsl:choose>
    </xsl:template>

    <xsl:template name="leftEndleavesSeparateFlyleaves-Fold">
        <xsl:param name="use" select="false()" as="xs:boolean"/>
        <xsl:param name="baseline"/>
        <xsl:param name="countComponents"/>
        <xsl:param name="currentComponent"/>
        <xsl:param name="certainty" select="100" as="xs:integer"/>
        <desc xmlns="http://www.w3.org/2000/svg">Folded flyleaves</desc>
        <g xmlns="http://www.w3.org/2000/svg">
            <path>
                <xsl:attribute name="d">
                    <xsl:text>M</xsl:text>
                    <xsl:value-of select="$Ax + 140"/>
                    <xsl:text>,</xsl:text>
                  <xsl:value-of select="$baseline - ($delta * $currentComponent)"/>

                    <xsl:text>&#32;L</xsl:text>
                    <xsl:value-of select="$Ax + ($delta * $countComponents) - 2"/>
                    <xsl:text>,</xsl:text>
                  <xsl:value-of select="$baseline - ($delta * $currentComponent)"/>

                    <xsl:text>&#32;A</xsl:text>
                    <xsl:value-of
                        select="$delta - 1 + (count(following-sibling::component)
* $delta)"/>
                    <xsl:text>,</xsl:text>
                    <xsl:value-of
                        select="$delta - 1 + (count(following-sibling::component)
* $delta)"/>
                    <xsl:text>&#32;</xsl:text>
                    <xsl:value-of select="0"/>
                    <xsl:text>&#32;</xsl:text>
                    <xsl:value-of select="0"/>
                    <xsl:text>,</xsl:text>
                    <xsl:value-of select="1"/>
```

```
                    <xsl:text>&#32;</xsl:text>
                    <xsl:value-of
                        select="$Ax + ($delta * $countComponents) - 2 - ($delta -
1 + (count(following-sibling::component) * $delta))"/>
                    <xsl:text>,</xsl:text>
                     <xsl:value-of select="$baseline - ($delta * $countComponents)
- 5"/>
                </xsl:attribute>
            </path>
            <path>
                <xsl:choose>
                    <xsl:when test="$certainty lt 100">
                        <xsl:call-template name="certainty">
                            <xsl:with-param name="certainty" select="$certainty"/>
                            <xsl:with-param name="type" select="'2'"/>
                        </xsl:call-template>
                    </xsl:when>
                </xsl:choose>
                <xsl:attribute name="d">
                    <xsl:text>M</xsl:text>
                    <xsl:value-of select="$Ax + 140"/>
                    <xsl:text>,</xsl:text>
                    <xsl:value-of
                        select="($baseline - ($delta * $currentComponent)) - (2 *
($delta - 1 + (count(following-sibling::component) * $delta)))"/>
                    <xsl:text>&#32;L</xsl:text>
                    <xsl:value-of select="$Ax + ($delta * $countComponents) - 2"/>
                    <xsl:text>,</xsl:text>
                    <xsl:value-of
                        select="($baseline - ($delta * $currentComponent)) - (2 *
($delta - 1 + (count(following-sibling::component) * $delta)))"/>
                    <xsl:text>&#32;A</xsl:text>
                    <xsl:value-of
                        select="$delta - 1 + (count(following-sibling::component)
* $delta)"/>
                    <xsl:text>,</xsl:text>
                    <xsl:value-of
                        select="$delta - 1 + (count(following-sibling::component)
* $delta)"/>
                    <xsl:text>&#32;</xsl:text>
                    <xsl:value-of select="0"/>
                    <xsl:text>&#32;</xsl:text>
                    <xsl:value-of select="0"/>
                    <xsl:text>,</xsl:text>
                    <xsl:value-of select="0"/>
                    <xsl:text>&#32;</xsl:text>
                    <xsl:value-of
                        select="$Ax + ($delta * $countComponents) - 2 - ($delta -
1 + (count(following-sibling::component) * $delta))"/>
                    <xsl:text>,</xsl:text>
                     <xsl:value-of select="$baseline - ($delta * $countComponents)
- 5"/>
                </xsl:attribute>
            </path>
        </g>
        <xsl:call-template name="componentAttachment">
            <xsl:with-param name="use" select="$use"/>
            <xsl:with-param name="countComponents" select="$countComponents"/>
            <xsl:with-param name="baseline"
                select="if (attachment/glued) then $baseline - ($delta * $current-
Component) + ($delta) else $baseline"
            />
        </xsl:call-template>
    </xsl:template>
```

```
    <xsl:template name="leftEndleavesSeparatePastedown">
        <xsl:param name="use" select="false()" as="xs:boolean"/>
        <xsl:param name="countComponents"/>
        <xsl:param name="currentComponent"/>
        <xsl:param name="baseline"/>
        <xsl:variable name="B1x"
          select="$Ax - ($delta * $countComponents) + $delta - 1 + (count(preceding-
sibling::component) * $delta) - 145"/>
        <xsl:variable name="B1y"
            select="$baseline -
            (2 * $delta * $countComponents) -
          (3 * $delta * count(ancestor::unit/preceding-sibling::unit/components/com-
ponent[pastedown/yes])) -
            ($delta * count(preceding-sibling::component)) +
           (if (ancestor::yes/type/integral/pastedown/yes) then ($delta * (ancest-
or::yes/type/integral/numberOfLeaves) + $delta) else 0)
            - 10 +
            (if (./type/hook/type[textHook]) then $delta * $countComponents -
               (if (ancestor::yes/type/integral) then $delta +  $delta * ancest-
or::yes/type/integral/numberOfLeaves else 0) - 5 else 0)"/>
        <xsl:call-template name="leftEndleavesSeparatePastedown-pastedown">
            <xsl:with-param name="use" select="$use"/>
            <xsl:with-param name="B1x" select="$B1x"/>
            <xsl:with-param name="B1y" select="$B1y"/>
            <xsl:with-param name="countComponents" select="$countComponents"/>
            <xsl:with-param name="baseline" select="$baseline"/>
            <xsl:with-param name="unknown"
                select="if (./type[NC | NK | other]) then 'yes' else 'no'"/>
        </xsl:call-template>
        <xsl:choose>
            <xsl:when test="./type[fold]">
                <desc xmlns="http://www.w3.org/2000/svg">Type fold</desc>
                <xsl:call-template name="leftEndleavesSeparatePastedown-Fold">
                    <xsl:with-param name="use" select="$use"/>
                    <xsl:with-param name="B1x" select="$B1x"/>
                    <xsl:with-param name="B1y" select="$B1y"/>
                 <xsl:with-param name="countComponents" select="$countComponents"/>

                   <xsl:with-param name="currentComponent" select="$currentCompon-
ent"/>
                    <xsl:with-param name="baseline" select="$baseline"/>
                </xsl:call-template>
            </xsl:when>
            <xsl:when test="./type[guard]">
                <desc xmlns="http://www.w3.org/2000/svg">Type guard</desc>
                <xsl:call-template name="leftEndleavesSeparatePastedown-Guard">
                    <xsl:with-param name="use" select="$use"/>
                    <xsl:with-param name="B1x" select="$B1x"/>
                    <xsl:with-param name="B1y" select="$B1y"/>
                 <xsl:with-param name="countComponents" select="$countComponents"/>

                   <xsl:with-param name="currentComponent" select="$currentCompon-
ent"/>
                    <xsl:with-param name="baseline" select="$baseline"/>
                </xsl:call-template>
            </xsl:when>
            <xsl:when test="./type[hook]">
                <desc xmlns="http://www.w3.org/2000/svg">Type hook</desc>
                <xsl:choose>
                    <xsl:when test="./type/hook/type[endleafHook]">
                          <desc xmlns="http://www.w3.org/2000/svg">Type pastedown-
endleaf-hook</desc>
                          <xsl:call-template name="leftEndleavesSeparatePastedown-
EndleafHook">
                            <xsl:with-param name="use" select="$use"/>
```

```
                            <xsl:with-param name="B1x" select="$B1x"/>
                            <xsl:with-param name="B1yParam" select="$B1y"/>
                            <xsl:with-param name="baseline" select="$baseline"/>
                             <xsl:with-param name="countComponents" select="$count-
Components"/>
                           <xsl:with-param name="currentComponent" select="$current-
Component"/>
                        </xsl:call-template>
                    </xsl:when>
                    <xsl:when test="./type/hook/type[textHook]">
                          <desc xmlns="http://www.w3.org/2000/svg">Type pastedown-
text-hook</desc>
                          <xsl:call-template name="leftEndleavesSeparatePastedown-
TextHook">
                            <xsl:with-param name="use" select="$use"/>
                            <xsl:with-param name="B1x" select="$B1x"/>
                            <xsl:with-param name="B1y" select="$B1y"/>
                             <xsl:with-param name="countComponents" select="$count-
Components"/>
                           <xsl:with-param name="currentComponent" select="$current-
Component"/>
                            <xsl:with-param name="baseline" select="$baseline"/>
                        </xsl:call-template>
                    </xsl:when>
                    <xsl:otherwise>
                        <desc xmlns="http://www.w3.org/2000/svg">Type not checked,
 not known, or
                            other type</desc>
                    </xsl:otherwise>
                </xsl:choose>
            </xsl:when>
            <xsl:when test="./type[outsideHook]">
                <desc xmlns="http://www.w3.org/2000/svg">Type outside hook</desc>
              <xsl:call-template name="leftEndleavesSeparatePastedown-OutsideHook">

                    <xsl:with-param name="use" select="$use"/>
                    <xsl:with-param name="B1x" select="$B1x"/>
                    <xsl:with-param name="B1y" select="$B1y"/>
                 <xsl:with-param name="countComponents" select="$countComponents"/>

                   <xsl:with-param name="currentComponent" select="$currentCompon-
ent"/>
                    <xsl:with-param name="baseline" select="$baseline"/>
                </xsl:call-template>
            </xsl:when>
            <xsl:when test="./type[singleLeaf]">
                <desc xmlns="http://www.w3.org/2000/svg">Type single leaf</desc>
                <!-- Pastedown already drawn when detected its presence -->
            </xsl:when>
            <xsl:otherwise>
              <desc xmlns="http://www.w3.org/2000/svg">Type not checked, not known,
 or other
                    type</desc>
                <!-- Pastedown already drawn when detected its presence -->
            </xsl:otherwise>
        </xsl:choose>
    </xsl:template>


    <xsl:template name="leftEndleavesSeparateFlyleaf-SingleLeaf">
        <xsl:param name="use" select="false()" as="xs:boolean"/>
        <xsl:param name="countComponents"/>
        <xsl:param name="currentComponent"/>
        <xsl:param name="baseline"/>
        <xsl:param name="certainty" select="100" as="xs:integer"/>
```

```
        <xsl:param name="unknown" select="'no'"/>
        <xsl:variable name="Dx" select="$Ax + ($delta * $countComponents) - 2"/>
        <xsl:variable name="Dy" select="$baseline - ($delta * $currentComponent)"/>

        <g xmlns="http://www.w3.org/2000/svg">
            <desc>Parametric path for the flyleaf</desc>
            <path>
                <xsl:choose>
                    <xsl:when test="$unknown eq 'yes'">
                        <xsl:attribute name="stroke">
                            <xsl:text>url(#fading)</xsl:text>
                        </xsl:attribute>
                    </xsl:when>
                </xsl:choose>
                <xsl:call-template name="certainty">
                    <xsl:with-param name="certainty" select="$certainty"/>
                    <xsl:with-param name="type" select="'4'"/>
                </xsl:call-template>
                <xsl:attribute name="d">
                    <xsl:text>M</xsl:text>
                    <xsl:value-of select="$Ax + 140"/>
                    <xsl:text>,</xsl:text>
                    <xsl:value-of select="$Dy"/>
                    <xsl:text>&#32;L</xsl:text>
                    <xsl:value-of select="$Ax + 2"/>
                    <xsl:text>,</xsl:text>
                    <xsl:value-of select="$Dy + (if ($unknown eq 'yes') then 0.0001
else 0)"/>
                </xsl:attribute>
            </path>
        </g>
    </xsl:template>

    <xsl:template name="leftEndleavesSeparateFlyleaf-SingleLeaf_part">
        <xsl:param name="use" select="false()" as="xs:boolean"/>
        <xsl:param name="countComponents"/>
        <xsl:param name="currentComponent"/>
        <xsl:param name="baseline"/>
        <xsl:param name="certainty" select="100" as="xs:integer"/>
        <xsl:variable name="Dx" select="$Ax + ($delta * $countComponents) - 2"/>
        <xsl:variable name="Dy" select="$baseline - ($delta * $currentComponent)"/>

        <g xmlns="http://www.w3.org/2000/svg">
            <desc>Parametric path for the flyleaf</desc>
            <path>
                <xsl:call-template name="certainty">
                    <xsl:with-param name="certainty" select="$certainty"/>
                    <xsl:with-param name="type" select="'4'"/>
                </xsl:call-template>
                <xsl:attribute name="d">
                    <xsl:text>M</xsl:text>
                    <xsl:value-of select="$Ax + 140"/>
                    <xsl:text>,</xsl:text>
                    <xsl:value-of select="$Dy"/>
                    <xsl:text>&#32;L</xsl:text>
                    <xsl:value-of select="$Dx"/>
                    <xsl:text>,</xsl:text>
                    <xsl:value-of select="$Dy"/>
                </xsl:attribute>
            </path>
        </g>
    </xsl:template>

    <xsl:template name="leftEndleavesSeparatePastedown-pastedown">
        <xsl:param name="use" select="false()" as="xs:boolean"/>
```

```
        <xsl:param name="B1x"/>
        <xsl:param name="B1y"/>
        <xsl:param name="countComponents"/>
        <xsl:param name="baseline"/>
        <xsl:param name="certainty" select="100"/>
        <xsl:param name="unknown" select="'no'"/>
       <g xmlns="http://www.w3.org/2000/svg" id="{concat('pastedown', position())}">

            <desc>pastedown</desc>
            <path>
                <xsl:choose>
                    <xsl:when test="$unknown eq 'yes'">
                        <xsl:attribute name="stroke">
                            <xsl:text>url(#fading2)</xsl:text>
                        </xsl:attribute>
                    </xsl:when>
                </xsl:choose>
                <xsl:call-template name="certainty">
                    <xsl:with-param name="certainty" select="$certainty"/>
                    <xsl:with-param name="type" select="'4'"/>
                </xsl:call-template>
                <xsl:attribute name="d">
                    <xsl:text>M</xsl:text>
                    <xsl:choose>
                        <xsl:when test="./type[outsideHook]">
                            <xsl:value-of select="$B1x + 115"/>
                        </xsl:when>
                        <xsl:when test="./type[guard]">
                            <xsl:value-of select="$B1x + 95"/>
                        </xsl:when>
                        <xsl:otherwise>
                            <xsl:value-of select="$B1x"/>
                        </xsl:otherwise>
                    </xsl:choose>
                    <xsl:text>,</xsl:text>
                    <xsl:choose>
                        <xsl:when test="./type/hook/type[textHook]">
                            <xsl:value-of select="$B1y + $delta"/>
                        </xsl:when>
                        <xsl:when test="./type/hook/type[endleafHook]">
                            <xsl:value-of select="$B1y - 5"/>
                        </xsl:when>
                        <xsl:otherwise>
                            <xsl:value-of select="$B1y"/>
                        </xsl:otherwise>
                    </xsl:choose>
                    <xsl:text>&#32;L</xsl:text>
                    <xsl:value-of select="$B1x + 130"/>
                    <xsl:text>,</xsl:text>
                    <xsl:choose>
                        <xsl:when test="./type/hook/type[textHook]">
                            <xsl:value-of
                               select="$B1y + $delta + (if ($unknown eq 'yes') then
 0.001 else 0)"
                            />
                        </xsl:when>
                        <xsl:when test="./type/hook/type[endleafHook]">
                            <xsl:value-of
                                  select="$B1y - 5 + (if ($unknown eq 'yes') then
0.001 else 0)"/>
                        </xsl:when>
                        <xsl:otherwise>
                             <xsl:value-of select="$B1y + (if ($unknown eq 'yes')
then 0.001 else 0)"
                            />
```

```
                        </xsl:otherwise>
                    </xsl:choose>
                </xsl:attribute>
            </path>
            <g xmlns="http://www.w3.org/2000/svg">
                <use xlink:href="#pasted">
                    <xsl:attribute name="x">
                        <xsl:value-of select="$B1x"/>
                    </xsl:attribute>
                    <xsl:choose>
                        <xsl:when test="./type/hook/type[textHook]">
                            <xsl:attribute name="y">
                                <xsl:value-of select="$B1y + $delta"/>
                            </xsl:attribute>
                        </xsl:when>
                        <xsl:when test="./type/hook/type[endleafHook]">
                            <xsl:attribute name="y">
                                <xsl:value-of select="$B1y - 5"/>
                            </xsl:attribute>
                        </xsl:when>
                        <xsl:otherwise>
                            <xsl:attribute name="y">
                                <xsl:value-of select="$B1y"/>
                            </xsl:attribute>
                        </xsl:otherwise>
                    </xsl:choose>
                    <xsl:choose>
                        <xsl:when test="./type[guard]">
                            <xsl:attribute name="clip-path">
                                <xsl:text>url(#guardClip)</xsl:text>
                            </xsl:attribute>
                        </xsl:when>
                        <xsl:when test="./type[outsideHook]">
                            <xsl:attribute name="clip-path">
                                <xsl:text>url(#outsideHookClip)</xsl:text>
                            </xsl:attribute>
                        </xsl:when>
                    </xsl:choose>
                </use>
            </g>
        </g>
        <xsl:choose>
            <xsl:when test="./type[singleLeaf]">
                <!-- do not call the component attachment -->
            </xsl:when>
            <xsl:otherwise>
                <xsl:call-template name="componentAttachment">
                    <xsl:with-param name="use" select="$use"/>
                 <xsl:with-param name="countComponents" select="$countComponents"/>

                    <xsl:with-param name="baseline" select="$baseline"/>
                    <xsl:with-param name="certainty"
                        select="if ($unknown eq 'yes') then 50 else 100" as="xs:in-
teger"/>
                    <xsl:with-param name="onlySewn"
                        select="if (./type[singleLeaf]) then 'yes' else 'no'"/>
                </xsl:call-template>
            </xsl:otherwise>
        </xsl:choose>
    </xsl:template>

    <xsl:template name="leftEndleavesSeparatePastedown-Fold">
        <xsl:param name="use" select="false()" as="xs:boolean"/>
        <xsl:param name="B1x"/>
        <xsl:param name="B1y"/>
```

```
        <xsl:param name="countComponents"/>
        <xsl:param name="currentComponent"/>
        <xsl:param name="baseline"/>
        <xsl:variable name="Dx" select="$Ax + ($delta * $countComponents) - 2"/>
       <xsl:variable name="Dy" select="$baseline - ($delta * $currentComponent)"/>

        <xsl:call-template name="leftEndleavesSeparateFlyleaf-SingleLeaf_part">
            <xsl:with-param name="use" select="$use"/>
            <xsl:with-param name="baseline" select="$baseline"/>
            <xsl:with-param name="countComponents" select="$countComponents"/>
            <xsl:with-param name="currentComponent" select="$currentComponent"/>
        </xsl:call-template>
        <g xmlns="http://www.w3.org/2000/svg">
              <desc>Parametric path describing the fold from the flyleaf to the
pastedown</desc>
            <path>
                <xsl:attribute name="d">
                    <xsl:text>M</xsl:text>
                    <xsl:value-of select="$Dx"/>
                    <xsl:text>,</xsl:text>
                    <xsl:value-of select="$Dy"/>
                    <xsl:text>&#32;A</xsl:text>
                    <xsl:value-of
                        select="$delta - 1 + (count(following-sibling::component)
* $delta)"/>
                    <xsl:text>,</xsl:text>
                    <xsl:value-of
                        select="$delta - 1 + (count(following-sibling::component)
* $delta)"/>
                    <xsl:text>&#32;</xsl:text>
                    <xsl:value-of select="0"/>
                    <xsl:text>&#32;</xsl:text>
                    <xsl:value-of select="0"/>
                    <xsl:text>,</xsl:text>
                    <xsl:value-of select="1"/>
                    <xsl:text>&#32;</xsl:text>
                     <xsl:value-of select="$Ax - 2 + ($delta * count(preceding-sib-
ling::component))"/>
                    <xsl:text>,</xsl:text>
                     <xsl:value-of select="$baseline - ($delta * $countComponents)
- 5"/>
                    <xsl:text>&#32;Q</xsl:text>
                     <xsl:value-of select="$Ax - 2 + ($delta * count(preceding-sib-
ling::component))"/>
                    <xsl:text>,</xsl:text>
                    <xsl:value-of select="$B1y"/>
                    <xsl:text>&#32;</xsl:text>
                    <xsl:value-of select="$B1x + 130"/>
                    <xsl:text>,</xsl:text>
                    <xsl:value-of select="$B1y"/>
                </xsl:attribute>
            </path>
        </g>
        <xsl:call-template name="componentAttachment">
            <xsl:with-param name="use" select="$use"/>
            <xsl:with-param name="countComponents" select="$countComponents"/>
            <xsl:with-param name="baseline"
                 select="if (attachment/glued) then $baseline - ($delta * $current-
Component) + ($delta) else $baseline"
            />
        </xsl:call-template>
    </xsl:template>

    <xsl:template name="leftEndleavesSeparateFlyleaves-Guard">
        <xsl:param name="use" select="false()" as="xs:boolean"/>
```

```
        <xsl:param name="baseline"/>
        <xsl:param name="countComponents"/>
        <xsl:param name="currentComponent"/>
        <desc xmlns="http://www.w3.org/2000/svg">Flyleaf guard</desc>
        <g xmlns="http://www.w3.org/2000/svg">
            <path>
                <xsl:attribute name="d">
                    <xsl:text>M</xsl:text>
                      <xsl:value-of select="$Ax + $delta + ($delta * $countCompon-
ents)"/>
                    <xsl:text>,</xsl:text>
                   <xsl:value-of select="$baseline - ($delta * $currentComponent)"/>

                    <xsl:text>&#32;L</xsl:text>
                    <xsl:value-of select="$Ax + ($delta * $countComponents) - 2"/>
                    <xsl:text>,</xsl:text>
                   <xsl:value-of select="$baseline - ($delta * $currentComponent)"/>

                    <xsl:text>&#32;A</xsl:text>
                    <xsl:value-of
                        select="$delta - 1 + (count(following-sibling::component)
* $delta)"/>
                    <xsl:text>,</xsl:text>
                    <xsl:value-of
                        select="$delta - 1 + (count(following-sibling::component)
* $delta)"/>
                    <xsl:text>&#32;</xsl:text>
                    <xsl:value-of select="0"/>
                    <xsl:text>&#32;</xsl:text>
                    <xsl:value-of select="0"/>
                    <xsl:text>,</xsl:text>
                    <xsl:value-of select="1"/>
                    <xsl:text>&#32;</xsl:text>
                    <xsl:value-of
                        select="$Ax + ($delta * $countComponents) - 2 - ($delta -
1 + (count(following-sibling::component) * $delta))"/>
                    <xsl:text>,</xsl:text>
                     <xsl:value-of select="$baseline - ($delta * $countComponents)
- 5"/>
                </xsl:attribute>
            </path>
            <path>
                <xsl:attribute name="d">
                    <xsl:text>M</xsl:text>
                   <xsl:value-of select="$Ax + (2 * $delta) + ($delta * $countCom-
ponents)"/>
                    <xsl:text>,</xsl:text>
                    <xsl:value-of
                        select="($baseline - ($delta * $currentComponent)) - (2 *
($delta - 1 + (count(following-sibling::component) * $delta)))"/>
                    <xsl:text>&#32;L</xsl:text>
                    <xsl:value-of select="$Ax + ($delta * $countComponents) - 2"/>
                    <xsl:text>,</xsl:text>
                    <xsl:value-of
                        select="($baseline - ($delta * $currentComponent)) - (2 *
($delta - 1 + (count(following-sibling::component) * $delta)))"/>
                    <xsl:text>&#32;A</xsl:text>
                    <xsl:value-of
                        select="$delta - 1 + (count(following-sibling::component)
* $delta)"/>
                    <xsl:text>,</xsl:text>
                    <xsl:value-of
                        select="$delta - 1 + (count(following-sibling::component)
* $delta)"/>
                    <xsl:text>&#32;</xsl:text>
```

```
                    <xsl:value-of select="0"/>
                    <xsl:text>&#32;</xsl:text>
                    <xsl:value-of select="0"/>
                    <xsl:text>,</xsl:text>
                    <xsl:value-of select="0"/>
                    <xsl:text>&#32;</xsl:text>
                    <xsl:value-of
                        select="$Ax + ($delta * $countComponents) - 2 - ($delta -
1 + (count(following-sibling::component) * $delta))"/>
                    <xsl:text>,</xsl:text>
                     <xsl:value-of select="$baseline - ($delta * $countComponents)
- 5"/>
                </xsl:attribute>
            </path>
        </g>
        <xsl:call-template name="componentAttachment">
            <xsl:with-param name="use" select="$use"/>
            <xsl:with-param name="countComponents" select="$countComponents"/>
            <xsl:with-param name="baseline"
                select="if (attachment/glued) then $baseline - ($delta * $current-
Component) + ($delta) else $baseline"
            />
        </xsl:call-template>
    </xsl:template>

    <xsl:template name="leftEndleavesSeparatePastedown-Guard">
        <xsl:param name="use" select="false()" as="xs:boolean"/>
        <xsl:param name="B1x"/>
        <xsl:param name="B1y"/>
        <xsl:param name="countComponents"/>
        <xsl:param name="currentComponent"/>
        <xsl:param name="baseline"/>
        <xsl:variable name="Dx" select="$Ax + ($delta * $countComponents) - 2"/>
       <xsl:variable name="Dy" select="$baseline - ($delta * $currentComponent)"/>

        <g xmlns="http://www.w3.org/2000/svg">
              <desc>Parametric path describing the fold from the flyleaf to the
pastedown</desc>
            <path>
                <xsl:call-template name="certainty">
                    <xsl:with-param name="certainty"
                        select="if ($currentComponent eq $countComponents) then 50
 else 100"/>
                    <xsl:with-param name="type" select="'4'"/>
                </xsl:call-template>
                <xsl:attribute name="d">
                    <xsl:text>M</xsl:text>
                      <xsl:value-of select="$Ax + $delta + ($delta * $countCompon-
ents)"/>
                    <xsl:text>,</xsl:text>
                  <xsl:value-of select="$baseline - ($delta * $currentComponent)"/>

                    <xsl:text>&#32;L</xsl:text>
                    <xsl:value-of select="$Ax + ($delta * $countComponents) - 2"/>
                    <xsl:text>,</xsl:text>
                  <xsl:value-of select="$baseline - ($delta * $currentComponent)"/>

                    <xsl:text>&#32;A</xsl:text>
                    <xsl:value-of
                        select="$delta - 1 + (count(following-sibling::component)
* $delta)"/>
                    <xsl:text>,</xsl:text>
                    <xsl:value-of
                        select="$delta - 1 + (count(following-sibling::component)
* $delta)"/>
```

```
                    <xsl:text>&#32;</xsl:text>
                    <xsl:value-of select="0"/>
                    <xsl:text>&#32;</xsl:text>
                    <xsl:value-of select="0"/>
                    <xsl:text>,</xsl:text>
                    <xsl:value-of select="1"/>
                    <xsl:text>&#32;</xsl:text>
                     <xsl:value-of select="$Ax - 2 + ($delta * count(preceding-sib-
ling::component))"/>
                    <xsl:text>,</xsl:text>
                     <xsl:value-of select="$baseline - ($delta * $countComponents)
- 5"/>
                </xsl:attribute>
            </path>
        </g>
        <g xmlns="http://www.w3.org/2000/svg">
              <desc>Parametric path describing the fold from the flyleaf to the
pastedown</desc>
            <path>
                <xsl:attribute name="d">
                    <xsl:text>M</xsl:text>
                     <xsl:value-of select="$Ax - 2 + ($delta * count(preceding-sib-
ling::component))"/>
                    <xsl:text>,</xsl:text>
                     <xsl:value-of select="$baseline - ($delta * $countComponents)
- 5"/>
                    <xsl:text>&#32;Q</xsl:text>
                     <xsl:value-of select="$Ax - 2 + ($delta * count(preceding-sib-
ling::component))"/>
                    <xsl:text>,</xsl:text>
                    <xsl:value-of select="$B1y"/>
                    <xsl:text>&#32;</xsl:text>
                    <xsl:value-of select="$B1x + 130"/>
                    <xsl:text>,</xsl:text>
                    <xsl:value-of select="$B1y"/>
                </xsl:attribute>
            </path>
        </g>
        <xsl:call-template name="componentAttachment">
            <xsl:with-param name="use" select="$use"/>
            <xsl:with-param name="countComponents" select="$countComponents"/>
            <xsl:with-param name="baseline"
                select="if (attachment/glued) then $baseline - ($delta * $current-
Component) + ($delta) else $baseline"
            />
        </xsl:call-template>
    </xsl:template>


    <xsl:template name="leftEndleavesSeparateFlyleaves-EndleafHook">
        <xsl:param name="use" select="false()" as="xs:boolean"/>
        <xsl:param name="baseline"/>
        <xsl:param name="countComponents"/>
        <xsl:param name="currentComponent"/>
        <xsl:param name="certainty" select="100"/>
        <desc xmlns="http://www.w3.org/2000/svg">Endleaf-hook</desc>
        <g xmlns="http://www.w3.org/2000/svg">
            <xsl:choose>
                <xsl:when test="./type/hook/double/yes">
                    <g xmlns="http://www.w3.org/2000/svg">
                        <path xmlns="http://www.w3.org/2000/svg">
                            <xsl:attribute name="d">
                                <xsl:text>M</xsl:text>
                                   <xsl:value-of select="$Ax + $delta + ($delta *
$countComponents)"/>
```

```
                                <xsl:text>,</xsl:text>
                               <xsl:value-of select="$baseline - ($delta * $current-
Component) + 2"/>
                                <xsl:text>&#32;L</xsl:text>
                                <xsl:value-of select="$Ax + ($delta * $countCompon-
ents)"/>
                                <xsl:text>,</xsl:text>
                               <xsl:value-of select="$baseline - ($delta * $current-
Component) + 2"/>
                                <xsl:text>&#32;A</xsl:text>
                                <xsl:value-of
                                select="$delta - 1 + (count(following-sibling::com-
ponent) * $delta) + 2.5"/>
                                <xsl:text>,</xsl:text>
                                <xsl:value-of
                                select="$delta - 1 + (count(following-sibling::com-
ponent) * $delta)  + 2.5"/>
                                <xsl:text>&#32;</xsl:text>
                                <xsl:value-of select="0"/>
                                <xsl:text>&#32;</xsl:text>
                                <xsl:value-of select="0"/>
                                <xsl:text>,</xsl:text>
                                <xsl:value-of select="1"/>
                                <xsl:text>&#32;</xsl:text>
                                <xsl:value-of
                                    select="$Ax + ($delta * $countComponents) - 2
- ($delta - 1 + (count(following-sibling::component) * $delta)) - 2"/>
                                <xsl:text>,</xsl:text>
                               <xsl:value-of select="$baseline - ($delta * $count-
Components) - 5"/>
                            </xsl:attribute>
                        </path>
                        <path xmlns="http://www.w3.org/2000/svg">
                            <xsl:call-template name="certainty">
                             <xsl:with-param name="certainty" select="$certainty"/>

                                <xsl:with-param name="type" select="'4'"/>
                            </xsl:call-template>
                            <xsl:attribute name="d">
                                <xsl:text>M</xsl:text>
                                <xsl:value-of select="$Ax + 140"/>
                                <xsl:text>,</xsl:text>
                                <xsl:value-of
                                 select="($baseline - ($delta * $currentComponent))
 - (2 * ($delta - 1 + (count(following-sibling::component) * $delta))) - 2"/>
                                <xsl:text>&#32;L</xsl:text>
                               <xsl:value-of select="$Ax + ($delta * $countCompon-
ents)"/>
                                <xsl:text>,</xsl:text>
                                <xsl:value-of
                                 select="($baseline - ($delta * $currentComponent))
 - (2 * ($delta - 1 + (count(following-sibling::component) * $delta))) - 2"/>
                                <xsl:text>&#32;A</xsl:text>
                                <xsl:value-of
                                select="$delta - 1 + (count(following-sibling::com-
ponent) * $delta) + 2.5"/>
                                <xsl:text>,</xsl:text>
                                <xsl:value-of
                                select="$delta - 1 + (count(following-sibling::com-
ponent) * $delta)  + 2.5"/>
                                <xsl:text>&#32;</xsl:text>
                                <xsl:value-of select="0"/>
                                <xsl:text>&#32;</xsl:text>
                                <xsl:value-of select="0"/>
                                <xsl:text>,</xsl:text>
```

```
                                <xsl:value-of select="0"/>
                                <xsl:text>&#32;</xsl:text>
                                <xsl:value-of
                                    select="$Ax + ($delta * $countComponents) - 2
- ($delta - 1 + (count(following-sibling::component) * $delta)) - 2"/>
                                <xsl:text>,</xsl:text>
                               <xsl:value-of select="$baseline - ($delta * $count-
Components) - 5"/>
                            </xsl:attribute>
                        </path>
                    </g>
                    <g xmlns="http://www.w3.org/2000/svg">
                        <g xmlns="http://www.w3.org/2000/svg">
                            <path xmlns="http://www.w3.org/2000/svg">
                                <xsl:attribute name="d">
                                    <xsl:text>M</xsl:text>
                                    <xsl:value-of
                                       select="$Ax + $delta + ($delta * $countCom-
ponents)"/>
                                    <xsl:text>,</xsl:text>
                                    <xsl:value-of
                                        select="$baseline - ($delta * $currentCom-
ponent) -1"/>
                                    <xsl:text>&#32;L</xsl:text>
                                   <xsl:value-of select="$Ax + ($delta * $countCom-
ponents)"/>
                                    <xsl:text>,</xsl:text>
                                    <xsl:value-of
                                        select="$baseline - ($delta * $currentCom-
ponent) -1"/>
                                    <xsl:text>&#32;A</xsl:text>
                                    <xsl:value-of
                                        select="$delta - 1 + (count(following-sib-
ling::component) * $delta) -0.5"/>
                                    <xsl:text>,</xsl:text>
                                    <xsl:value-of
                                        select="$delta - 1 + (count(following-sib-
ling::component) * $delta) -0.5"/>
                                    <xsl:text>&#32;</xsl:text>
                                    <xsl:value-of select="0"/>
                                    <xsl:text>&#32;</xsl:text>
                                    <xsl:value-of select="0"/>
                                    <xsl:text>,</xsl:text>
                                    <xsl:value-of select="1"/>
                                    <xsl:text>&#32;</xsl:text>
                                    <xsl:value-of
                                        select="$Ax + ($delta * $countComponents)
- 2 - ($delta - 1 + (count(following-sibling::component) * $delta)) + 1"/>
                                    <xsl:text>,</xsl:text>
                                    <xsl:value-of
                                       select="$baseline - ($delta * $countCompon-
ents) - 5"/>
                                </xsl:attribute>
                            </path>
                            <path xmlns="http://www.w3.org/2000/svg">
                                <xsl:call-template name="certainty">
                                    <xsl:with-param name="certainty" select="$cer-
tainty"/>
                                    <xsl:with-param name="type" select="'4'"/>
                                </xsl:call-template>
                                <xsl:attribute name="d">
                                    <xsl:text>M</xsl:text>
                                    <xsl:value-of select="$Ax + 140"/>
                                    <xsl:text>,</xsl:text>
                                    <xsl:value-of
```

```
                                    select="($baseline - ($delta * $currentCom-
ponent)) - (2 * ($delta - 1 + (count(following-sibling::component) * $delta))) +
1"/>
                                  <xsl:text>&#32;L</xsl:text>
                                 <xsl:value-of select="$Ax + ($delta * $countCom-
ponents)"/>
                                  <xsl:text>,</xsl:text>
                                  <xsl:value-of
                                     select="($baseline - ($delta * $currentCom-
ponent)) - (2 * ($delta - 1 + (count(following-sibling::component) * $delta))) +
1"/>
                                  <xsl:text>&#32;A</xsl:text>
                                  <xsl:value-of
                                      select="$delta - 1 + (count(following-sib-
ling::component) * $delta) - 0.5"/>
                                  <xsl:text>,</xsl:text>
                                  <xsl:value-of
                                      select="$delta - 1 + (count(following-sib-
ling::component) * $delta) - 0.5"/>
                                  <xsl:text>&#32;</xsl:text>
                                  <xsl:value-of select="0"/>
                                  <xsl:text>&#32;</xsl:text>
                                  <xsl:value-of select="0"/>
                                  <xsl:text>,</xsl:text>
                                  <xsl:value-of select="0"/>
                                  <xsl:text>&#32;</xsl:text>
                                  <xsl:value-of
                                      select="$Ax + ($delta * $countComponents)
- 2 - ($delta - 1 + (count(following-sibling::component) * $delta)) + 1"/>
                                  <xsl:text>,</xsl:text>
                                  <xsl:value-of
                                     select="$baseline - ($delta * $countCompon-
ents) - 5"/>
                              </xsl:attribute>
                           </path>
                        </g>
                         <path xmlns="http://www.w3.org/2000/svg" stroke-line-
cap="round">
                          <!-- always uncertain as the schema does not tell if
the double is a single or two sheets -->
                          <xsl:call-template name="certainty">
                              <xsl:with-param name="certainty" select="50"/>
                              <xsl:with-param name="type" select="'4'"/>
                          </xsl:call-template>
                          <xsl:attribute name="d">
                              <xsl:text>M</xsl:text>
                                 <xsl:value-of select="$Ax + $delta + ($delta *
$countComponents)"/>
                              <xsl:text>,</xsl:text>
                            <xsl:value-of select="$baseline - ($delta * $current-
Component) - 1"/>
                              <xsl:text>&#32;Q</xsl:text>
                              <xsl:value-of
                               select="$Ax + $delta + ($delta * $countComponents)
 + 2"/>
                              <xsl:text>,</xsl:text>
                            <xsl:value-of select="$baseline - ($delta * $current-
Component) - 1"/>
                              <xsl:text>&#32;</xsl:text>
                              <xsl:value-of
                               select="$Ax + $delta + ($delta * $countComponents)
 + 2"/>
                              <xsl:text>,</xsl:text>
                            <xsl:value-of select="$baseline - ($delta * $current-
Component)"/>
```

```
                                <xsl:text>&#32;Q</xsl:text>
                                <xsl:value-of
                                 select="$Ax + $delta + ($delta * $countComponents)
 + 2"/>
                                <xsl:text>,</xsl:text>
                               <xsl:value-of select="$baseline - ($delta * $current-
Component) + 2"/>
                                <xsl:text>&#32;</xsl:text>
                                   <xsl:value-of select="$Ax + $delta + ($delta *
$countComponents)"/>
                                <xsl:text>,</xsl:text>
                               <xsl:value-of select="$baseline - ($delta * $current-
Component) + 2"
                                />
                            </xsl:attribute>
                        </path>
                    </g>
                </xsl:when>
                <xsl:otherwise>
                    <path xmlns="http://www.w3.org/2000/svg">
                        <xsl:attribute name="d">
                            <xsl:text>M</xsl:text>
                             <xsl:value-of select="$Ax + $delta + ($delta * $count-
Components)"/>
                            <xsl:text>,</xsl:text>
                           <xsl:value-of select="$baseline - ($delta * $currentCom-
ponent)"/>
                            <xsl:text>&#32;L</xsl:text>
                           <xsl:value-of select="$Ax + ($delta * $countComponents)
 - 2"/>
                            <xsl:text>,</xsl:text>
                          <xsl:value-of select="$baseline - ($delta * $currentCom-
ponent)"/>
                            <xsl:text>&#32;A</xsl:text>
                            <xsl:value-of
                               select="$delta - 1 + (count(following-sibling::com-
ponent) * $delta)"/>
                            <xsl:text>,</xsl:text>
                            <xsl:value-of
                               select="$delta - 1 + (count(following-sibling::com-
ponent) * $delta)"/>
                            <xsl:text>&#32;</xsl:text>
                            <xsl:value-of select="0"/>
                            <xsl:text>&#32;</xsl:text>
                            <xsl:value-of select="0"/>
                            <xsl:text>,</xsl:text>
                            <xsl:value-of select="1"/>
                            <xsl:text>&#32;</xsl:text>
                            <xsl:value-of
                                  select="$Ax + ($delta * $countComponents) - 2 -
($delta - 1 + (count(following-sibling::component) * $delta))"/>
                            <xsl:text>,</xsl:text>
                             <xsl:value-of select="$baseline - ($delta * $countCom-
ponents) - 5"/>
                        </xsl:attribute>
                    </path>
                    <path xmlns="http://www.w3.org/2000/svg">
                        <xsl:call-template name="certainty">
                            <xsl:with-param name="certainty" select="$certainty"/>
                            <xsl:with-param name="type" select="'4'"/>
                        </xsl:call-template>
                        <xsl:attribute name="d">
                            <xsl:text>M</xsl:text>
                            <xsl:value-of select="$Ax + 140"/>
                            <xsl:text>,</xsl:text>
```

```
                            <xsl:value-of
                                select="($baseline - ($delta * $currentComponent))
 - (2 * ($delta - 1 + (count(following-sibling::component) * $delta)))"/>
                            <xsl:text>&#32;L</xsl:text>
                            <xsl:value-of select="$Ax + ($delta * $countComponents)
 - 2"/>
                            <xsl:text>,</xsl:text>
                            <xsl:value-of
                                select="($baseline - ($delta * $currentComponent))
 - (2 * ($delta - 1 + (count(following-sibling::component) * $delta)))"/>
                            <xsl:text>&#32;A</xsl:text>
                            <xsl:value-of
                               select="$delta - 1 + (count(following-sibling::com-
ponent) * $delta)"/>
                            <xsl:text>,</xsl:text>
                            <xsl:value-of
                               select="$delta - 1 + (count(following-sibling::com-
ponent) * $delta)"/>
                            <xsl:text>&#32;</xsl:text>
                            <xsl:value-of select="0"/>
                            <xsl:text>&#32;</xsl:text>
                            <xsl:value-of select="0"/>
                            <xsl:text>,</xsl:text>
                            <xsl:value-of select="0"/>
                            <xsl:text>&#32;</xsl:text>
                            <xsl:value-of
                                 select="$Ax + ($delta * $countComponents) - 2 -
($delta - 1 + (count(following-sibling::component) * $delta))"/>
                            <xsl:text>,</xsl:text>
                             <xsl:value-of select="$baseline - ($delta * $countCom-
ponents) - 5"/>
                        </xsl:attribute>
                    </path>
                </xsl:otherwise>
            </xsl:choose>
        </g>
        <xsl:call-template name="componentAttachment">
            <xsl:with-param name="use" select="$use"/>
            <xsl:with-param name="countComponents" select="$countComponents"/>
            <xsl:with-param name="baseline"
                select="if (attachment/glued) then $baseline - ($delta * $current-
Component) + ($delta) else $baseline"
            />
        </xsl:call-template>
    </xsl:template>

    <xsl:template name="leftEndleavesSeparatePastedown-EndleafHook">
        <xsl:param name="use" select="false()" as="xs:boolean"/>
        <xsl:param name="B1x"/>
        <xsl:param name="B1yParam"/>
        <xsl:param name="countComponents"/>
        <xsl:param name="currentComponent"/>
        <xsl:param name="baseline"/>
        <xsl:param name="certainty" select="100"/>
        <xsl:variable name="Dx" select="$Ax + ($delta * $countComponents) - 2"/>
       <xsl:variable name="Dy" select="$baseline - ($delta * $currentComponent)"/>

        <xsl:variable name="B1y" select="$B1yParam - 5"/>
        <desc xmlns="http://www.w3.org/2000/svg">Pastedown endleaf-hook</desc>
        <g xmlns="http://www.w3.org/2000/svg">
            <xsl:choose>
                <xsl:when test="./type/hook/double/yes">
                    <path xmlns="http://www.w3.org/2000/svg">
                        <xsl:attribute name="d">
                            <xsl:text>M</xsl:text>
```

```
                            <xsl:value-of select="$Ax + $delta + ($delta * $count-
Components)"/>
                            <xsl:text>,</xsl:text>
                           <xsl:value-of select="$baseline - ($delta * $currentCom-
ponent) + 2"/>
                            <xsl:text>&#32;L</xsl:text>
                           <xsl:value-of select="$Ax + ($delta * $countComponents)
 - 2"/>
                            <xsl:text>,</xsl:text>
                           <xsl:value-of select="$baseline - ($delta * $currentCom-
ponent) + 2"/>
                            <xsl:text>&#32;Q</xsl:text>
                            <xsl:value-of
                                  select="$Ax - 2 + ($delta * count(preceding-sib-
ling::component)) - 2"/>
                            <xsl:text>,</xsl:text>
                           <xsl:value-of select="$baseline - ($delta * $currentCom-
ponent) + 2"/>
                            <xsl:text>&#32;</xsl:text>
                            <xsl:value-of
                                  select="$Ax - 2 + ($delta * count(preceding-sib-
ling::component)) - 2"/>
                            <xsl:text>,</xsl:text>
                            <xsl:value-of select="$baseline - ($delta * $countCom-
ponents) - 5"/>
                            <xsl:text>&#32;Q</xsl:text>
                            <xsl:value-of
                                  select="$Ax - 2 + ($delta * count(preceding-sib-
ling::component)) - 2"/>
                            <xsl:text>,</xsl:text>
                            <xsl:value-of select="$B1y"/>
                            <xsl:text>&#32;</xsl:text>
                            <xsl:value-of select="$B1x + 130"/>
                            <xsl:text>,</xsl:text>
                            <xsl:value-of select="$B1y"/>
                        </xsl:attribute>
                    </path>
                    <path xmlns="http://www.w3.org/2000/svg">
                        <xsl:call-template name="certainty">
                            <xsl:with-param name="certainty" select="$certainty"/>
                            <xsl:with-param name="type" select="'4'"/>
                        </xsl:call-template>
                        <xsl:attribute name="d">
                            <xsl:text>M</xsl:text>
                            <xsl:value-of select="$Ax + $delta + ($delta * $count-
Components)"/>
                            <xsl:text>,</xsl:text>
                           <xsl:value-of select="$baseline - ($delta * $currentCom-
ponent)"/>
                            <xsl:text>&#32;L</xsl:text>
                           <xsl:value-of select="$Ax + ($delta * $countComponents)
 - 2"/>
                            <xsl:text>,</xsl:text>
                           <xsl:value-of select="$baseline - ($delta * $currentCom-
ponent)"/>
                            <xsl:text>&#32;A</xsl:text>
                            <xsl:value-of
                               select="$delta - 1 + (count(following-sibling::com-
ponent) * $delta)"/>
                            <xsl:text>,</xsl:text>
                            <xsl:value-of
                               select="$delta - 1 + (count(following-sibling::com-
ponent) * $delta)"/>
                            <xsl:text>&#32;</xsl:text>
                            <xsl:value-of select="0"/>
```

```
                        <xsl:text>&#32;</xsl:text>
                        <xsl:value-of select="0"/>
                        <xsl:text>,</xsl:text>
                        <xsl:value-of select="1"/>
                        <xsl:text>&#32;</xsl:text>
                       <xsl:value-of select="$Ax + ($delta * $countComponents)
 - 2"/>
                        <xsl:text>,</xsl:text>
                        <xsl:value-of
                            select="($baseline - ($delta * $currentComponent))
 - (2 * ($delta - 1 + (count(following-sibling::component) * $delta)))"/>
                        <xsl:text>&#32;L</xsl:text>
                        <xsl:value-of select="$Ax + 140"/>
                        <xsl:text>,</xsl:text>
                        <xsl:value-of
                            select="($baseline - ($delta * $currentComponent))
 - (2 * ($delta - 1 + (count(following-sibling::component) * $delta)))"
                        />
                    </xsl:attribute>
                </path>
                <path xmlns="http://www.w3.org/2000/svg">

                    <xsl:call-template name="certainty">
                        <xsl:with-param name="certainty" select="50"/>
                        <xsl:with-param name="type" select="'4'"/>
                    </xsl:call-template>
                    <xsl:attribute name="d">
                        <xsl:text>M</xsl:text>
                        <xsl:value-of select="$Ax + $delta + ($delta * $count-
Components)"/>
                        <xsl:text>,</xsl:text>
                       <xsl:value-of select="$baseline - ($delta * $currentCom-
ponent)"/>
                        <xsl:text>&#32;Q</xsl:text>
                        <xsl:value-of select="$Ax + $delta + ($delta * $count-
Components) + 1"/>
                        <xsl:text>,</xsl:text>
                       <xsl:value-of select="$baseline - ($delta * $currentCom-
ponent)"/>
                        <xsl:text>&#32;</xsl:text>
                        <xsl:value-of select="$Ax + $delta + ($delta * $count-
Components) + 1"/>
                        <xsl:text>,</xsl:text>
                       <xsl:value-of select="$baseline - ($delta * $currentCom-
ponent) + 1"/>
                        <xsl:text>&#32;Q</xsl:text>
                        <xsl:value-of select="$Ax + $delta + ($delta * $count-
Components) + 1"/>
                        <xsl:text>,</xsl:text>
                       <xsl:value-of select="$baseline - ($delta * $currentCom-
ponent) + 2"/>
                        <xsl:text>&#32;</xsl:text>
                        <xsl:value-of select="$Ax + $delta + ($delta * $count-
Components)"/>
                        <xsl:text>,</xsl:text>
                       <xsl:value-of select="$baseline - ($delta * $currentCom-
ponent) + 2"/>
                    </xsl:attribute>
                </path>
            </xsl:when>
            <xsl:otherwise>
                <path xmlns="http://www.w3.org/2000/svg">
                    <xsl:call-template name="certainty">
                        <xsl:with-param name="certainty" select="$certainty"/>
                        <xsl:with-param name="type" select="'4'"/>
```

```
                        </xsl:call-template>
                        <xsl:attribute name="d">
                            <xsl:text>M</xsl:text>
                            <xsl:value-of select="$Ax + $delta + ($delta * $count-
Components)"/>
                            <xsl:text>,</xsl:text>
                           <xsl:value-of select="$baseline - ($delta * $currentCom-
ponent)"/>
                            <xsl:text>&#32;L</xsl:text>
                           <xsl:value-of select="$Ax + ($delta * $countComponents)
 - 2"/>
                            <xsl:text>,</xsl:text>
                           <xsl:value-of select="$baseline - ($delta * $currentCom-
ponent)"/>
                            <xsl:text>&#32;A</xsl:text>
                            <xsl:value-of
                               select="$delta - 1 + (count(following-sibling::com-
ponent) * $delta)"/>
                            <xsl:text>,</xsl:text>
                            <xsl:value-of
                               select="$delta - 1 + (count(following-sibling::com-
ponent) * $delta)"/>
                            <xsl:text>&#32;</xsl:text>
                            <xsl:value-of select="0"/>
                            <xsl:text>&#32;</xsl:text>
                            <xsl:value-of select="0"/>
                            <xsl:text>,</xsl:text>
                            <xsl:value-of select="1"/>
                            <xsl:text>&#32;</xsl:text>
                            <xsl:value-of
                                  select="$Ax - 2 + ($delta * count(preceding-sib-
ling::component))"/>
                            <xsl:text>,</xsl:text>
                            <xsl:value-of select="$baseline - ($delta * $countCom-
ponents) - 5"/>
                            <xsl:text>&#32;Q</xsl:text>
                            <xsl:value-of
                                  select="$Ax - 2 + ($delta * count(preceding-sib-
ling::component))"/>
                            <xsl:text>,</xsl:text>
                            <xsl:value-of select="$B1y"/>
                            <xsl:text>&#32;</xsl:text>
                            <xsl:value-of select="$B1x + 130"/>
                            <xsl:text>,</xsl:text>
                            <xsl:value-of select="$B1y"/>
                        </xsl:attribute>
                    </path>
                </xsl:otherwise>
            </xsl:choose>
        </g>
        <xsl:call-template name="componentAttachment">
            <xsl:with-param name="use" select="$use"/>
            <xsl:with-param name="countComponents" select="$countComponents"/>
            <xsl:with-param name="baseline"
                select="if (attachment/glued) then $baseline - ($delta * $current-
Component) + ($delta) else $baseline"
            />
        </xsl:call-template>
    </xsl:template>


    <xsl:template name="leftEndleavesSeparateFlyleaves-TextHook">
        <xsl:param name="use" select="false()" as="xs:boolean"/>
        <xsl:param name="baseline"/>
        <xsl:param name="countComponents"/>
```

```
        <xsl:param name="currentComponent"/>
        <xsl:param name="certainty" select="100"/>
        <desc xmlns="http://www.w3.org/2000/svg">Text-hook</desc>
        <g xmlns="http://www.w3.org/2000/svg">
            <xsl:choose>
                <xsl:when test="./type/hook/double/yes">
                    <g xmlns="http://www.w3.org/2000/svg">
                        <path xmlns="http://www.w3.org/2000/svg">
                            <xsl:attribute name="d">
                                <xsl:text>M</xsl:text>
                                <xsl:value-of
                                    select="$Ax + (2 * $delta) + ($delta * $count-
Components)"/>
                                <xsl:text>,</xsl:text>
                                <xsl:value-of
                                      select="$baseline + $delta + 20 + ($delta *
$countComponents) - ($delta * $currentComponent) - 1"/>
                                <xsl:text>&#32;L</xsl:text>
                                <xsl:value-of select="$Ax + $delta + 10"/>
                                <xsl:text>,</xsl:text>
                                <xsl:value-of
                                      select="$baseline + $delta + 20 + ($delta *
$countComponents) - ($delta * $currentComponent) - 1"/>
                                <xsl:text>&#32;A</xsl:text>
                                <xsl:value-of
                                       select="$delta + 10 + (count(following-sib-
ling::component) * $delta) - 1.5"/>
                                <xsl:text>,</xsl:text>
                                <xsl:value-of
                                       select="$delta + 10 + (count(following-sib-
ling::component) * $delta) - 1.5"/>
                                <xsl:text>&#32;</xsl:text>
                                <xsl:value-of select="0"/>
                                <xsl:text>&#32;</xsl:text>
                                <xsl:value-of select="0"/>
                                <xsl:text>,</xsl:text>
                                <xsl:value-of select="1"/>
                                <xsl:text>&#32;</xsl:text>
                                <xsl:value-of
                                      select="$Ax - ($delta * $countComponents) +
$delta - 1 + (count(preceding-sibling::component) * $delta) + 1"/>
                                <xsl:text>,</xsl:text>
                                <xsl:value-of select="$baseline + 10"/>
                            </xsl:attribute>
                        </path>
                        <xsl:choose>
                            <xsl:when test="ancestor::yes/type/integral">
                                <path>
                                    <xsl:call-template name="certainty">
                                     <xsl:with-param name="certainty" select="$cer-
tainty"/>
                                        <xsl:with-param name="type" select="'4'"/>
                                    </xsl:call-template>
                                    <xsl:attribute name="d">
                                        <xsl:text>M</xsl:text>
                                        <xsl:value-of
                                          select="$Ax - ($delta * $countComponents)
 + $delta - 1 + (count(preceding-sibling::component) * $delta) + 1"/>
                                        <xsl:text>,</xsl:text>
                                        <xsl:value-of select="$baseline + 10"/>
                                        <xsl:text>&#32;Q</xsl:text>
                                        <xsl:value-of
                                           select="$Ax - $delta * (count(following-
sibling::component))"/>
                                        <xsl:text>,</xsl:text>
```

```
                                    <xsl:value-of
                                        select="($baseline - ($delta * ancest-
or::yes/type/integral/numberOfLeaves) - $delta) - $delta * (count(following-sib-
ling::component))"/>
                                    <xsl:text>&#32;</xsl:text>
                                    <xsl:value-of select="$Ax + 16"/>
                                    <xsl:text>,</xsl:text>
                                    <xsl:value-of
                                        select="($baseline - ($delta * ancest-
or::yes/type/integral/numberOfLeaves) - $delta) - $delta * (count(following-sib-
ling::component))"/>
                                    <xsl:text>&#32;L</xsl:text>
                                    <xsl:value-of select="$Ax + 140"/>
                                    <xsl:text>,</xsl:text>
                                    <xsl:value-of
                                        select="($baseline - ($delta * ancest-
or::yes/type/integral/numberOfLeaves) - $delta) - $delta * (count(following-sib-
ling::component))"
                                    />
                                </xsl:attribute>
                            </path>
                        </xsl:when>
                        <xsl:otherwise>
                            <path xmlns="http://www.w3.org/2000/svg">
                                <xsl:call-template name="certainty">
                                 <xsl:with-param name="certainty" select="$cer-
tainty"/>
                                    <xsl:with-param name="type" select="'4'"/>
                                </xsl:call-template>
                                <xsl:attribute name="d">
                                    <xsl:text>M</xsl:text>
                                    <xsl:value-of select="$Ax + 140"/>
                                    <xsl:text>,</xsl:text>
                                    <xsl:value-of
                                        select="$baseline - ($delta + ($delta
* $countComponents) - ($delta * $currentComponent)) + 1"/>
                                    <xsl:text>&#32;L</xsl:text>
                                    <xsl:value-of select="$Ax + $delta + 10"/>
                                    <xsl:text>,</xsl:text>
                                    <xsl:value-of
                                        select="$baseline - ($delta + ($delta
* $countComponents) - ($delta * $currentComponent)) + 1"/>
                                    <xsl:text>&#32;A</xsl:text>
                                    <xsl:value-of
                                       select="$delta + 10 + (count(following-
sibling::component) * $delta) - 1.5"/>
                                    <xsl:text>,</xsl:text>
                                    <xsl:value-of
                                       select="$delta + 10 + (count(following-
sibling::component) * $delta) - 1.5"/>
                                    <xsl:text>&#32;</xsl:text>
                                    <xsl:value-of select="0"/>
                                    <xsl:text>&#32;</xsl:text>
                                    <xsl:value-of select="0"/>
                                    <xsl:text>,</xsl:text>
                                    <xsl:value-of select="0"/>
                                    <xsl:text>&#32;</xsl:text>
                                    <xsl:value-of
                                      select="$Ax - ($delta * $countComponents)
 + $delta - 1 + (count(preceding-sibling::component) * $delta) + 1"/>
                                    <xsl:text>,</xsl:text>
                                    <xsl:value-of select="$baseline + 10"/>
                                </xsl:attribute>
                            </path>
                        </xsl:otherwise>
```

```
                    </xsl:choose>
                </g>
                <g xmlns="http://www.w3.org/2000/svg">
                    <g xmlns="http://www.w3.org/2000/svg">
                        <path xmlns="http://www.w3.org/2000/svg">
                            <xsl:attribute name="d">
                                <xsl:text>M</xsl:text>
                                <xsl:value-of
                                       select="$Ax + (2 * $delta) + ($delta *
$countComponents)"/>
                                <xsl:text>,</xsl:text>
                                <xsl:value-of
                                    select="$baseline + $delta + 20 + ($delta
* $countComponents) - ($delta * $currentComponent) + 1"/>
                                <xsl:text>&#32;L</xsl:text>
                                <xsl:value-of select="$Ax + $delta + 10"/>
                                <xsl:text>,</xsl:text>
                                <xsl:value-of
                                    select="$baseline + $delta + 20 + ($delta
* $countComponents) - ($delta * $currentComponent) + 1"/>
                                <xsl:text>&#32;A</xsl:text>
                                <xsl:value-of
                                   select="$delta + 10 + (count(following-sib-
ling::component) * $delta) + 0.5"/>
                                <xsl:text>,</xsl:text>
                                <xsl:value-of
                                   select="$delta + 10 + (count(following-sib-
ling::component) * $delta) + 0.5"/>
                                <xsl:text>&#32;</xsl:text>
                                <xsl:value-of select="0"/>
                                <xsl:text>&#32;</xsl:text>
                                <xsl:value-of select="0"/>
                                <xsl:text>,</xsl:text>
                                <xsl:value-of select="1"/>
                                <xsl:text>&#32;</xsl:text>
                                <xsl:value-of
                                    select="$Ax - ($delta * $countComponents)
+ $delta - 1 + (count(preceding-sibling::component) * $delta) - 1"/>
                                <xsl:text>,</xsl:text>
                                <xsl:value-of select="$baseline + 10"/>
                            </xsl:attribute>
                        </path>
                        <xsl:choose>
                            <xsl:when test="ancestor::yes/type/integral">
                                <path>
                                    <xsl:call-template name="certainty">
                                          <xsl:with-param name="certainty" se-
lect="$certainty"/>
                                     <xsl:with-param name="type" select="'4'"/>

                                    </xsl:call-template>
                                    <xsl:attribute name="d">
                                        <xsl:text>M</xsl:text>
                                        <xsl:value-of
                                            select="$Ax - ($delta * $countCom-
ponents) + $delta - 1 + (count(preceding-sibling::component) * $delta) - 1"/>
                                        <xsl:text>,</xsl:text>
                                       <xsl:value-of select="$baseline + 10"/>

                                        <xsl:text>&#32;Q</xsl:text>
                                        <xsl:value-of
                                            select="$Ax - $delta * (count(fol-
lowing-sibling::component)) - 2"/>
                                        <xsl:text>,</xsl:text>
                                        <xsl:value-of
```

```
                                                select="($baseline - ($delta * an-
cestor::yes/type/integral/numberOfLeaves) - $delta) - 2 - $delta * (count(following-
sibling::component))"/>
                                            <xsl:text>&#32;</xsl:text>
                                            <xsl:value-of select="$Ax + 16"/>
                                            <xsl:text>,</xsl:text>
                                            <xsl:value-of
                                                select="($baseline - ($delta * an-
cestor::yes/type/integral/numberOfLeaves) - $delta) - 2 - $delta * (count(following-
sibling::component))"/>
                                            <xsl:text>&#32;L</xsl:text>
                                            <xsl:value-of select="$Ax + 140"/>
                                            <xsl:text>,</xsl:text>
                                            <xsl:value-of
                                                select="($baseline - ($delta * an-
cestor::yes/type/integral/numberOfLeaves) - $delta) - 2 - $delta * (count(following-
sibling::component))"
                                            />
                                        </xsl:attribute>
                                    </path>
                                </xsl:when>
                                <xsl:otherwise>
                                    <path xmlns="http://www.w3.org/2000/svg">
                                        <xsl:call-template name="certainty">
                                              <xsl:with-param name="certainty" se-
lect="$certainty"/>
                                         <xsl:with-param name="type" select="'4'"/>

                                        </xsl:call-template>
                                        <xsl:attribute name="d">
                                            <xsl:text>M</xsl:text>
                                            <xsl:value-of select="$Ax + 140"/>
                                            <xsl:text>,</xsl:text>
                                            <xsl:value-of
                                              select="$baseline - ($delta + ($delta
 * $countComponents) - ($delta * $currentComponent)) - 1"/>
                                            <xsl:text>&#32;L</xsl:text>
                                             <xsl:value-of select="$Ax + $delta +
10"/>
                                            <xsl:text>,</xsl:text>
                                            <xsl:value-of
                                              select="$baseline - ($delta + ($delta
 * $countComponents) - ($delta * $currentComponent)) - 1"/>
                                            <xsl:text>&#32;A</xsl:text>
                                            <xsl:value-of
                                               select="$delta + 10 + (count(follow-
ing-sibling::component) * $delta) + 0.5"/>
                                            <xsl:text>,</xsl:text>
                                            <xsl:value-of
                                               select="$delta + 10 + (count(follow-
ing-sibling::component) * $delta) + 0.5"/>
                                            <xsl:text>&#32;</xsl:text>
                                            <xsl:value-of select="0"/>
                                            <xsl:text>&#32;</xsl:text>
                                            <xsl:value-of select="0"/>
                                            <xsl:text>,</xsl:text>
                                            <xsl:value-of select="0"/>
                                            <xsl:text>&#32;</xsl:text>
                                            <xsl:value-of
                                                select="$Ax - ($delta * $countCom-
ponents) + $delta - 1 + (count(preceding-sibling::component) * $delta) - 1"/>
                                            <xsl:text>,</xsl:text>
                                           <xsl:value-of select="$baseline + 10"/>

                                        </xsl:attribute>
```

```
                                </path>
                            </xsl:otherwise>
                        </xsl:choose>
                    </g>
                    <path xmlns="http://www.w3.org/2000/svg">
                        <xsl:call-template name="certainty">
                            <xsl:with-param name="certainty" select="50"/>
                            <xsl:with-param name="type" select="'3'"/>
                        </xsl:call-template>
                        <xsl:attribute name="d">
                            <xsl:text>M</xsl:text>
                            <xsl:value-of
                                select="$Ax + (2 * $delta) + ($delta * $count-
Components)"/>
                            <xsl:text>,</xsl:text>
                            <xsl:value-of
                                  select="$baseline + $delta + 20 + ($delta *
$countComponents) - ($delta * $currentComponent) + 1"/>
                            <xsl:text>&#32;Q</xsl:text>
                            <xsl:value-of
                                select="$Ax + (2 * $delta) + ($delta * $count-
Components) + 2"/>
                            <xsl:text>,</xsl:text>
                            <xsl:value-of
                                  select="$baseline + $delta + 20 + ($delta *
$countComponents) - ($delta * $currentComponent) + 1"/>
                            <xsl:text>&#32;</xsl:text>
                            <xsl:value-of
                                select="$Ax + (2 * $delta) + ($delta * $count-
Components) + 2"/>
                            <xsl:text>&#32;</xsl:text>
                            <xsl:value-of
                                  select="$baseline + $delta + 20 + ($delta *
$countComponents) - ($delta * $currentComponent)"/>
                            <xsl:text>&#32;Q</xsl:text>
                            <xsl:value-of
                                select="$Ax + (2 * $delta) + ($delta * $count-
Components) + 2"/>
                            <xsl:text>,</xsl:text>
                            <xsl:value-of
                                  select="$baseline + $delta + 20 + ($delta *
$countComponents) - ($delta * $currentComponent) - 1"/>
                            <xsl:text>&#32;</xsl:text>
                            <xsl:value-of
                                select="$Ax + (2 * $delta) + ($delta * $count-
Components)"/>
                            <xsl:text>,</xsl:text>
                            <xsl:value-of
                                  select="$baseline + $delta + 20 + ($delta *
$countComponents) - ($delta * $currentComponent) - 1"
                                />
                        </xsl:attribute>
                    </path>
                </g>
            </xsl:when>
            <xsl:otherwise>
                <g xmlns="http://www.w3.org/2000/svg">
                    <path xmlns="http://www.w3.org/2000/svg">
                        <xsl:attribute name="d">
                            <xsl:text>M</xsl:text>
                            <xsl:value-of
                                select="$Ax + (2 * $delta) + ($delta * $count-
Components)"/>
                            <xsl:text>,</xsl:text>
                            <xsl:value-of
```

```
                                    select="$baseline + $delta + 20 + ($delta *
$countComponents) - ($delta * $currentComponent) - 1"/>
                                <xsl:text>&#32;L</xsl:text>
                                <xsl:value-of select="$Ax + $delta + 10"/>
                                <xsl:text>,</xsl:text>
                                <xsl:value-of
                                      select="$baseline + $delta + 20 + ($delta *
$countComponents) - ($delta * $currentComponent) - 1"/>
                                <xsl:text>&#32;A</xsl:text>
                                <xsl:value-of
                                       select="$delta + 10 + (count(following-sib-
ling::component) * $delta) - 1.5"/>
                                <xsl:text>,</xsl:text>
                                <xsl:value-of
                                       select="$delta + 10 + (count(following-sib-
ling::component) * $delta) - 1.5"/>
                                <xsl:text>&#32;</xsl:text>
                                <xsl:value-of select="0"/>
                                <xsl:text>&#32;</xsl:text>
                                <xsl:value-of select="0"/>
                                <xsl:text>,</xsl:text>
                                <xsl:value-of select="1"/>
                                <xsl:text>&#32;</xsl:text>
                                <xsl:value-of
                                      select="$Ax - ($delta * $countComponents) +
$delta - 1 + (count(preceding-sibling::component) * $delta) + 1"/>
                                <xsl:text>,</xsl:text>
                                <xsl:value-of select="$baseline + 10"/>
                            </xsl:attribute>
                        </path>
                        <xsl:choose>
                            <xsl:when test="ancestor::yes/type/integral">
                                <path>
                                    <xsl:call-template name="certainty">
                                     <xsl:with-param name="certainty" select="$cer-
tainty"/>
                                        <xsl:with-param name="type" select="'4'"/>
                                    </xsl:call-template>
                                    <xsl:attribute name="d">
                                        <xsl:text>M</xsl:text>
                                        <xsl:value-of
                                          select="$Ax - ($delta * $countComponents)
 + $delta - 1 + (count(preceding-sibling::component) * $delta) + 1"/>
                                        <xsl:text>,</xsl:text>
                                        <xsl:value-of select="$baseline + 10"/>
                                        <xsl:text>&#32;Q</xsl:text>
                                        <xsl:value-of
                                           select="$Ax - $delta * (count(following-
sibling::component))"/>
                                        <xsl:text>,</xsl:text>
                                        <xsl:value-of
                                            select="($baseline - ($delta * ancest-
or::yes/type/integral/numberOfLeaves) - $delta) - $delta * (count(following-sib-
ling::component))"/>
                                        <xsl:text>&#32;</xsl:text>
                                        <xsl:value-of select="$Ax + 16"/>
                                        <xsl:text>,</xsl:text>
                                        <xsl:value-of
                                            select="($baseline - ($delta * ancest-
or::yes/type/integral/numberOfLeaves) - $delta) - $delta * (count(following-sib-
ling::component))"/>
                                        <xsl:text>&#32;L</xsl:text>
                                        <xsl:value-of select="$Ax + 140"/>
                                        <xsl:text>,</xsl:text>
                                        <xsl:value-of
```

```
                                        select="($baseline - ($delta * ancest-
or::yes/type/integral/numberOfLeaves) - $delta) - $delta * (count(following-sib-
ling::component))"
                                        />
                                    </xsl:attribute>
                                </path>
                            </xsl:when>
                            <xsl:otherwise>
                                <path xmlns="http://www.w3.org/2000/svg">
                                    <xsl:call-template name="certainty">
                                     <xsl:with-param name="certainty" select="$cer-
tainty"/>
                                        <xsl:with-param name="type" select="'4'"/>
                                    </xsl:call-template>
                                    <xsl:attribute name="d">
                                        <xsl:text>M</xsl:text>
                                        <xsl:value-of select="$Ax + 140"/>
                                        <xsl:text>,</xsl:text>
                                        <xsl:value-of
                                            select="$baseline - ($delta + ($delta
* $countComponents) - ($delta * $currentComponent)) + 1"/>
                                        <xsl:text>&#32;L</xsl:text>
                                        <xsl:value-of select="$Ax + $delta + 10"/>
                                        <xsl:text>,</xsl:text>
                                        <xsl:value-of
                                            select="$baseline - ($delta + ($delta
* $countComponents) - ($delta * $currentComponent)) + 1"/>
                                        <xsl:text>&#32;A</xsl:text>
                                        <xsl:value-of
                                           select="$delta + 10 + (count(following-
sibling::component) * $delta) - 1.5"/>
                                        <xsl:text>,</xsl:text>
                                        <xsl:value-of
                                           select="$delta + 10 + (count(following-
sibling::component) * $delta) - 1.5"/>
                                        <xsl:text>&#32;</xsl:text>
                                        <xsl:value-of select="0"/>
                                        <xsl:text>&#32;</xsl:text>
                                        <xsl:value-of select="0"/>
                                        <xsl:text>,</xsl:text>
                                        <xsl:value-of select="0"/>
                                        <xsl:text>&#32;</xsl:text>
                                        <xsl:value-of
                                          select="$Ax - ($delta * $countComponents)
 + $delta - 1 + (count(preceding-sibling::component) * $delta) + 1"/>
                                        <xsl:text>,</xsl:text>
                                        <xsl:value-of select="$baseline + 10"/>
                                    </xsl:attribute>
                                </path>
                            </xsl:otherwise>
                        </xsl:choose>
                    </g>
                </xsl:otherwise>
            </xsl:choose>
        </g>
        <xsl:call-template name="componentAttachment">
            <xsl:with-param name="use" select="$use"/>
            <xsl:with-param name="countComponents" select="$countComponents"/>
            <!-- GOOD -->
            <xsl:with-param name="baseline"
               select="if (attachment/glued) then $baseline + ($delta * $countCom-
ponents) - ($delta * $currentComponent) else $baseline"
            />
        </xsl:call-template>
    </xsl:template>
```

```
    <xsl:template name="leftEndleavesSeparatePastedown-TextHook">
        <xsl:param name="use" select="false()" as="xs:boolean"/>
        <xsl:param name="baseline"/>
        <xsl:param name="countComponents"/>
        <xsl:param name="currentComponent"/>
        <xsl:param name="B1x"/>
        <xsl:param name="B1y"/>
        <xsl:param name="certainty" select="100"/>
        <desc xmlns="http://www.w3.org/2000/svg">Text-hook</desc>
        <g xmlns="http://www.w3.org/2000/svg">
            <xsl:choose>
                <xsl:when test="./type/hook/double/yes">
                    <path xmlns="http://www.w3.org/2000/svg">
                        <xsl:call-template name="certainty">
                            <xsl:with-param name="certainty" select="$certainty"/>
                            <xsl:with-param name="type" select="'4'"/>
                        </xsl:call-template>
                        <xsl:attribute name="d">
                            <xsl:text>M</xsl:text>
                             <xsl:value-of select="$Ax + (2 * $delta) + ($delta *
$countComponents)"/>
                            <xsl:text>,</xsl:text>
                            <xsl:value-of
                               select="$baseline + $delta + 20 + ($delta * $count-
Components) - ($delta * $currentComponent) + 1"/>
                            <xsl:text>&#32;L</xsl:text>
                            <xsl:value-of select="$Ax + $delta + 10"/>
                            <xsl:text>,</xsl:text>
                            <xsl:value-of
                               select="$baseline + $delta + 20 + ($delta * $count-
Components) - ($delta * $currentComponent) + 1"/>
                            <xsl:text>&#32;A</xsl:text>
                            <xsl:value-of
                              select="$delta + 10 + (count(following-sibling::com-
ponent) * $delta) + 0.5"/>
                            <xsl:text>,</xsl:text>
                            <xsl:value-of
                              select="$delta + 10 + (count(following-sibling::com-
ponent) * $delta) + 0.5"/>
                            <xsl:text>&#32;</xsl:text>
                            <xsl:value-of select="0"/>
                            <xsl:text>&#32;</xsl:text>
                            <xsl:value-of select="0"/>
                            <xsl:text>,</xsl:text>
                            <xsl:value-of select="1"/>
                            <xsl:text>&#32;</xsl:text>
                            <xsl:value-of
                                select="$Ax - ($delta * $countComponents) + $delta
 - 1 + (count(preceding-sibling::component) * $delta) - 2"/>
                            <xsl:text>,</xsl:text>
                            <xsl:value-of select="$baseline + 10"/>
                        </xsl:attribute>
                    </path>
                    <xsl:choose>
                        <xsl:when test="ancestor::yes/type/integral">
                            <path>
                                <xsl:call-template name="certainty">
                                    <xsl:with-param name="certainty" select="$cer-
tainty"/>
                                    <xsl:with-param name="type" select="'4'"/>
                                </xsl:call-template>
                                <xsl:attribute name="d">
                                    <xsl:text>M</xsl:text>
                                    <xsl:value-of
```

```
                                    select="$Ax - ($delta * $countComponents)
+ $delta - 1 + (count(preceding-sibling::component) * $delta) - 2"/>
                                <xsl:text>,</xsl:text>
                                <xsl:value-of select="$baseline + 10"/>
                                <xsl:text>&#32;Q</xsl:text>
                                <xsl:value-of
                                    select="$Ax - ($delta * $countComponents)
+ $delta - 1 + (count(preceding-sibling::component) * $delta) - 2"/>
                                <xsl:text>,</xsl:text>
                                <xsl:value-of select="$B1y + $delta"/>
                                <xsl:text>&#32;</xsl:text>
                                <xsl:value-of select="$B1x + 130"/>
                                <xsl:text>,</xsl:text>
                                <xsl:value-of select="$B1y + $delta"/>
                            </xsl:attribute>
                        </path>
                    </xsl:when>
                    <xsl:otherwise>
                        <path xmlns="http://www.w3.org/2000/svg">
                            <xsl:call-template name="certainty">
                                <xsl:with-param name="certainty" select="$cer-
tainty"/>
                                <xsl:with-param name="type" select="'4'"/>
                            </xsl:call-template>
                            <xsl:attribute name="d">
                                <xsl:text>M</xsl:text>
                                <xsl:value-of
                                    select="$Ax - ($delta * $countComponents)
+ $delta - 1 + (count(preceding-sibling::component) * $delta) - 2"/>
                                <xsl:text>,</xsl:text>
                                <xsl:value-of select="$baseline + 10"/>
                                <xsl:text>&#32;Q</xsl:text>
                                <xsl:value-of
                                    select="$Ax - ($delta * $countComponents)
+ $delta - 1 + (count(preceding-sibling::component) * $delta) - 2"/>
                                <xsl:text>,</xsl:text>
                                <xsl:value-of select="$B1y + $delta"/>
                                <xsl:text>&#32;</xsl:text>
                                <xsl:value-of select="$B1x + 130"/>
                                <xsl:text>,</xsl:text>
                                <xsl:value-of select="$B1y + $delta"/>
                            </xsl:attribute>
                        </path>
                    </xsl:otherwise>
                </xsl:choose>
                <path xmlns="http://www.w3.org/2000/svg">
                    <xsl:call-template name="certainty">
                        <xsl:with-param name="certainty" select="$certainty"/>
                        <xsl:with-param name="type" select="'4'"/>
                    </xsl:call-template>
                    <xsl:attribute name="d">
                        <xsl:text>M</xsl:text>
                         <xsl:value-of select="$Ax + (2 * $delta) + ($delta *
$countComponents)"/>
                        <xsl:text>,</xsl:text>
                        <xsl:value-of
                           select="$baseline + $delta + 20 + ($delta * $count-
Components) - ($delta * $currentComponent) - 1"/>
                        <xsl:text>&#32;L</xsl:text>
                        <xsl:value-of select="$Ax + $delta + 10"/>
                        <xsl:text>,</xsl:text>
                        <xsl:value-of
                           select="$baseline + $delta + 20 + ($delta * $count-
Components) - ($delta * $currentComponent) - 1"/>
                        <xsl:text>&#32;A</xsl:text>
```

```
                            <xsl:value-of
                              select="$delta + 10 + (count(following-sibling::com-
ponent) * $delta) - 1.5"/>
                            <xsl:text>,</xsl:text>
                            <xsl:value-of
                              select="$delta + 10 + (count(following-sibling::com-
ponent) * $delta) - 1.5"/>
                            <xsl:text>&#32;</xsl:text>
                            <xsl:value-of select="0"/>
                            <xsl:text>&#32;</xsl:text>
                            <xsl:value-of select="0"/>
                            <xsl:text>,</xsl:text>
                            <xsl:value-of select="1"/>
                            <xsl:text>&#32;</xsl:text>
                            <xsl:value-of
                                select="$Ax - ($delta * $countComponents) + $delta
 - 1 + (count(preceding-sibling::component) * $delta) + 1"/>
                            <xsl:text>,</xsl:text>
                            <xsl:value-of select="$baseline + 10"/>
                        </xsl:attribute>
                    </path>
                    <xsl:choose>
                        <xsl:when test="ancestor::yes/type/integral">
                            <path>
                                <xsl:call-template name="certainty">
                                    <xsl:with-param name="certainty" select="$cer-
tainty"/>
                                    <xsl:with-param name="type" select="'4'"/>
                                </xsl:call-template>
                                <xsl:attribute name="d">
                                    <xsl:text>M</xsl:text>
                                    <xsl:value-of
                                        select="$Ax - ($delta * $countComponents)
+ $delta - 1 + (count(preceding-sibling::component) * $delta) + 1"/>
                                    <xsl:text>,</xsl:text>
                                    <xsl:value-of select="$baseline + 10"/>
                                    <xsl:text>&#32;Q</xsl:text>
                                    <xsl:value-of
                                      select="$Ax - $delta * (count(following-
sibling::component))"/>
                                    <xsl:text>,</xsl:text>
                                    <xsl:value-of
                                        select="($baseline - ($delta * ancest-
or::yes/type/integral/numberOfLeaves) - $delta) - $delta * (count(following-sib-
ling::component))"/>
                                    <xsl:text>&#32;</xsl:text>
                                    <xsl:value-of select="$Ax + 16"/>
                                    <xsl:text>,</xsl:text>
                                    <xsl:value-of
                                        select="($baseline - ($delta * ancest-
or::yes/type/integral/numberOfLeaves) - $delta) - $delta * (count(following-sib-
ling::component))"/>
                                    <xsl:text>&#32;L</xsl:text>
                                    <xsl:value-of select="$Ax + 140"/>
                                    <xsl:text>,</xsl:text>
                                    <xsl:value-of
                                        select="($baseline - ($delta * ancest-
or::yes/type/integral/numberOfLeaves) - $delta) - $delta * (count(following-sib-
ling::component))"
                                    />
                                </xsl:attribute>
                            </path>
                        </xsl:when>
                        <xsl:otherwise>
                            <path xmlns="http://www.w3.org/2000/svg">
```

```
                                <xsl:call-template name="certainty">
                                    <xsl:with-param name="certainty" select="$cer-
tainty"/>
                                    <xsl:with-param name="type" select="'4'"/>
                                </xsl:call-template>
                                <xsl:attribute name="d">
                                    <xsl:text>M</xsl:text>
                                    <xsl:value-of select="$Ax + 140"/>
                                    <xsl:text>,</xsl:text>
                                    <xsl:value-of
                                          select="$baseline - ($delta + ($delta *
$countComponents) - ($delta * $currentComponent)) + 1"/>
                                    <xsl:text>&#32;L</xsl:text>
                                    <xsl:value-of select="$Ax + $delta + 10"/>
                                    <xsl:text>,</xsl:text>
                                    <xsl:value-of
                                          select="$baseline - ($delta + ($delta *
$countComponents) - ($delta * $currentComponent)) + 1"/>
                                    <xsl:text>&#32;A</xsl:text>
                                    <xsl:value-of
                                       select="$delta + 10 + (count(following-sib-
ling::component) * $delta) - 1.5"/>
                                    <xsl:text>,</xsl:text>
                                    <xsl:value-of
                                       select="$delta + 10 + (count(following-sib-
ling::component) * $delta) - 1.5"/>
                                    <xsl:text>&#32;</xsl:text>
                                    <xsl:value-of select="0"/>
                                    <xsl:text>&#32;</xsl:text>
                                    <xsl:value-of select="0"/>
                                    <xsl:text>,</xsl:text>
                                    <xsl:value-of select="0"/>
                                    <xsl:text>&#32;</xsl:text>
                                    <xsl:value-of
                                        select="$Ax - ($delta * $countComponents)
+ $delta - 1 + (count(preceding-sibling::component) * $delta) + 1"/>
                                    <xsl:text>,</xsl:text>
                                    <xsl:value-of select="$baseline + 10"/>
                                </xsl:attribute>
                            </path>
                        </xsl:otherwise>
                    </xsl:choose>
                    <path xmlns="http://www.w3.org/2000/svg">
                        <xsl:call-template name="certainty">
                            <xsl:with-param name="certainty" select="50"/>
                            <xsl:with-param name="type" select="'4'"/>
                        </xsl:call-template>
                        <xsl:attribute name="d">
                            <xsl:text>M</xsl:text>
                             <xsl:value-of select="$Ax + (2 * $delta) + ($delta *
$countComponents)"/>
                            <xsl:text>,</xsl:text>
                            <xsl:value-of
                               select="$baseline + $delta + 20 + ($delta * $count-
Components) - ($delta * $currentComponent) + 1"/>
                            <xsl:text>&#32;Q</xsl:text>
                            <xsl:value-of
                              select="$Ax + (2 * $delta) + ($delta * $countCompon-
ents) + 2"/>
                            <xsl:text>,</xsl:text>
                            <xsl:value-of
                               select="$baseline + $delta + 20 + ($delta * $count-
Components) - ($delta * $currentComponent) + 1"/>
                            <xsl:text>&#32;</xsl:text>
                            <xsl:value-of
```

```
                                select="$Ax + (2 * $delta) + ($delta * $countCompon-
ents) + 2"/>
                            <xsl:text>&#32;</xsl:text>
                            <xsl:value-of
                                select="$baseline + $delta + 20 + ($delta * $count-
Components) - ($delta * $currentComponent)"/>
                            <xsl:text>&#32;Q</xsl:text>
                            <xsl:value-of
                                select="$Ax + (2 * $delta) + ($delta * $countCompon-
ents) + 2"/>
                            <xsl:text>,</xsl:text>
                            <xsl:value-of
                                select="$baseline + $delta + 20 + ($delta * $count-
Components) - ($delta * $currentComponent) - 1"/>
                            <xsl:text>&#32;</xsl:text>
                             <xsl:value-of select="$Ax + (2 * $delta) + ($delta *
$countComponents)"/>
                            <xsl:text>,</xsl:text>
                            <xsl:value-of
                                select="$baseline + $delta + 20 + ($delta * $count-
Components) - ($delta * $currentComponent) - 1"
                                />
                        </xsl:attribute>
                    </path>
                </xsl:when>
                <xsl:otherwise>
                    <path xmlns="http://www.w3.org/2000/svg">
                        <xsl:call-template name="certainty">
                            <xsl:with-param name="certainty" select="$certainty"/>
                            <xsl:with-param name="type" select="'4'"/>
                        </xsl:call-template>
                        <xsl:attribute name="d">
                            <xsl:text>M</xsl:text>
                             <xsl:value-of select="$Ax + (2 * $delta) + ($delta *
$countComponents)"/>
                            <xsl:text>,</xsl:text>
                            <xsl:value-of
                                select="$baseline + $delta + 20 + ($delta * $count-
Components) - ($delta * $currentComponent)"/>
                            <xsl:text>&#32;L</xsl:text>
                            <xsl:value-of select="$Ax + $delta + 10"/>
                            <xsl:text>,</xsl:text>
                            <xsl:value-of
                                select="$baseline + $delta + 20 + ($delta * $count-
Components) - ($delta * $currentComponent)"/>
                            <xsl:text>&#32;A</xsl:text>
                            <xsl:value-of
                                select="$delta + 10 + (count(following-sibling::com-
ponent) * $delta)"/>
                            <xsl:text>,</xsl:text>
                            <xsl:value-of
                                select="$delta + 10 + (count(following-sibling::com-
ponent) * $delta)"/>
                            <xsl:text>&#32;</xsl:text>
                            <xsl:value-of select="0"/>
                            <xsl:text>&#32;</xsl:text>
                            <xsl:value-of select="0"/>
                            <xsl:text>,</xsl:text>
                            <xsl:value-of select="1"/>
                            <xsl:text>&#32;</xsl:text>
                            <xsl:value-of
                                select="$Ax - ($delta * $countComponents) + $delta
 - 1 + (count(preceding-sibling::component) * $delta)"/>
                            <xsl:text>,</xsl:text>
                            <xsl:value-of select="$baseline + 10"/>
```

```
                            <xsl:text>&#32;Q</xsl:text>
                            <xsl:value-of
                                select="$Ax - ($delta * $countComponents) + $delta
 - 1 + (count(preceding-sibling::component) * $delta)"/>
                            <xsl:text>,</xsl:text>
                            <xsl:value-of select="$B1y + $delta"/>
                            <xsl:text>&#32;</xsl:text>
                            <xsl:value-of select="$B1x + 130"/>
                            <xsl:text>,</xsl:text>
                            <xsl:value-of select="$B1y + $delta"/>
                        </xsl:attribute>
                    </path>
                </xsl:otherwise>
            </xsl:choose>
        </g>
        <xsl:call-template name="componentAttachment">
            <xsl:with-param name="use" select="$use"/>
            <xsl:with-param name="countComponents" select="$countComponents"/>
            <xsl:with-param name="baseline"
                select="if (attachment/glued) then $baseline + ($delta * $countCom-
ponents) - ($delta * $currentComponent) else $baseline"
            />
        </xsl:call-template>
    </xsl:template>


    <xsl:template name="leftEndleavesSeparateFlyleaves-OutsideHook">
        <xsl:param name="use" select="false()" as="xs:boolean"/>
        <xsl:param name="baseline"/>
        <xsl:param name="countComponents"/>
        <xsl:param name="currentComponent"/>
        <xsl:param name="certainty"/>
        <desc xmlns="http://www.w3.org/2000/svg">Outside hook</desc>
        <g xmlns="http://www.w3.org/2000/svg">
            <path>
                <xsl:attribute name="d">
                    <xsl:text>M</xsl:text>
                    <xsl:value-of select="$Ax + 140"/>
                    <xsl:text>,</xsl:text>
                  <xsl:value-of select="$baseline - ($delta * $currentComponent)"/>

                    <xsl:text>&#32;L</xsl:text>
                    <xsl:value-of select="$Ax + ($delta * $countComponents) - 2"/>
                    <xsl:text>,</xsl:text>
                  <xsl:value-of select="$baseline - ($delta * $currentComponent)"/>

                    <xsl:text>&#32;A</xsl:text>
                    <xsl:value-of
                        select="$delta - 1 + (count(following-sibling::component)
* $delta)"/>
                    <xsl:text>,</xsl:text>
                    <xsl:value-of
                        select="$delta - 1 + (count(following-sibling::component)
* $delta)"/>
                    <xsl:text>&#32;</xsl:text>
                    <xsl:value-of select="0"/>
                    <xsl:text>&#32;</xsl:text>
                    <xsl:value-of select="0"/>
                    <xsl:text>,</xsl:text>
                    <xsl:value-of select="1"/>
                    <xsl:text>&#32;</xsl:text>
                    <xsl:value-of
                        select="$Ax + ($delta * $countComponents) - 2 - ($delta -
1 + (count(following-sibling::component) * $delta))"/>
                    <xsl:text>,</xsl:text>
```

```
                    <xsl:value-of select="$baseline - ($delta * $countComponents)
- 5"/>
                </xsl:attribute>
            </path>
            <path>
                <xsl:call-template name="certainty">
                    <xsl:with-param name="certainty" select="$certainty"/>
                    <xsl:with-param name="type" select="'4'"/>
                </xsl:call-template>
                <xsl:attribute name="d">
                    <xsl:text>M</xsl:text>
                   <xsl:value-of select="$Ax + (2 * $delta) + ($delta * $countCom-
ponents)"/>
                    <xsl:text>,</xsl:text>
                    <xsl:value-of
                        select="($baseline - ($delta * $currentComponent)) - (2 *
($delta - 1 + (count(following-sibling::component) * $delta)))"/>
                    <xsl:text>&#32;L</xsl:text>
                    <xsl:value-of select="$Ax + ($delta * $countComponents) - 2"/>
                    <xsl:text>,</xsl:text>
                    <xsl:value-of
                        select="($baseline - ($delta * $currentComponent)) - (2 *
($delta - 1 + (count(following-sibling::component) * $delta)))"/>
                    <xsl:text>&#32;A</xsl:text>
                    <xsl:value-of
                        select="$delta - 1 + (count(following-sibling::component)
* $delta)"/>
                    <xsl:text>,</xsl:text>
                    <xsl:value-of
                        select="$delta - 1 + (count(following-sibling::component)
* $delta)"/>
                    <xsl:text>&#32;</xsl:text>
                    <xsl:value-of select="0"/>
                    <xsl:text>&#32;</xsl:text>
                    <xsl:value-of select="0"/>
                    <xsl:text>,</xsl:text>
                    <xsl:value-of select="0"/>
                    <xsl:text>&#32;</xsl:text>
                    <xsl:value-of
                        select="$Ax + ($delta * $countComponents) - 2 - ($delta -
1 + (count(following-sibling::component) * $delta))"/>
                    <xsl:text>,</xsl:text>
                     <xsl:value-of select="$baseline - ($delta * $countComponents)
- 5"/>
                </xsl:attribute>
            </path>
        </g>
        <xsl:call-template name="componentAttachment">
            <xsl:with-param name="use" select="$use"/>
            <xsl:with-param name="countComponents" select="$countComponents"/>
            <xsl:with-param name="baseline"
                select="if (attachment/glued) then $baseline - ($delta * $current-
Component) + ($delta) else $baseline"
            />
        </xsl:call-template>
    </xsl:template>

    <xsl:template name="leftEndleavesSeparatePastedown-OutsideHook">
        <xsl:param name="use" select="false()" as="xs:boolean"/>
        <xsl:param name="B1x"/>
        <xsl:param name="B1y"/>
        <xsl:param name="countComponents"/>
        <xsl:param name="currentComponent"/>
        <xsl:param name="baseline"/>
        <xsl:variable name="Dx" select="$Ax + ($delta * $countComponents) - 2"/>
```

```
        <xsl:variable name="Dy" select="$baseline - ($delta * $currentComponent)"/>

        <g xmlns="http://www.w3.org/2000/svg">
            <desc>Parametric path for the flyleaf</desc>
            <path>
                <!-- Because the schema does not go down to the element level and
does not allow for each part of the endleaf
                     to be pasted as pastedown, this flyleaf id drawn as uncertain
to accommodate for the possibility that it
                     was actually a pastedown as it is common in Dutch bindings -->
                <xsl:call-template name="certainty">
                    <xsl:with-param name="certainty" select="50"/>
                    <xsl:with-param name="type" select="'4'"/>
                </xsl:call-template>
                <xsl:attribute name="d">
                    <xsl:text>M</xsl:text>
                    <xsl:value-of select="$Ax + 140"/>
                    <xsl:text>,</xsl:text>
                    <xsl:value-of select="$Dy"/>
                    <xsl:text>&#32;L</xsl:text>
                    <xsl:value-of select="$Dx"/>
                    <xsl:text>,</xsl:text>
                    <xsl:value-of select="$Dy"/>
                    <xsl:text>&#32;A</xsl:text>
                    <xsl:value-of
                        select="$delta - 1 + (count(following-sibling::component)
* $delta)"/>
                    <xsl:text>,</xsl:text>
                    <xsl:value-of
                        select="$delta - 1 + (count(following-sibling::component)
* $delta)"/>
                    <xsl:text>&#32;</xsl:text>
                    <xsl:value-of select="0"/>
                    <xsl:text>&#32;</xsl:text>
                    <xsl:value-of select="0"/>
                    <xsl:text>,</xsl:text>
                    <xsl:value-of select="1"/>
                    <xsl:text>&#32;</xsl:text>
                    <xsl:value-of select="$Ax - 2 + ($delta * count(preceding-sib-
ling::component))"/>
                    <xsl:text>,</xsl:text>
                    <xsl:value-of select="$baseline - ($delta * $countComponents)
- 5"/>
                </xsl:attribute>
            </path>
        </g>
        <g xmlns="http://www.w3.org/2000/svg">
              <desc>Parametric path describing the fold from the flyleaf to the
pastedown</desc>
            <path>
                <xsl:attribute name="d">
                    <xsl:text>M</xsl:text>
                    <xsl:value-of select="$Ax - 2 + ($delta * count(preceding-sib-
ling::component))"/>
                    <xsl:text>,</xsl:text>
                    <xsl:value-of select="$baseline - ($delta * $countComponents)
- 5"/>
                    <xsl:text>&#32;Q</xsl:text>
                    <xsl:value-of select="$Ax - 2 + ($delta * count(preceding-sib-
ling::component))"/>
                    <xsl:text>,</xsl:text>
                    <xsl:value-of select="$B1y"/>
                    <xsl:text>&#32;</xsl:text>
                    <xsl:value-of select="$B1x + 130"/>
                    <xsl:text>,</xsl:text>
```

```
                    <xsl:value-of select="$B1y"/>
                </xsl:attribute>
            </path>
        </g>
        <xsl:call-template name="componentAttachment">
            <xsl:with-param name="use" select="$use"/>
            <xsl:with-param name="countComponents" select="$countComponents"/>
            <xsl:with-param name="baseline"
                select="if (attachment/glued) then $baseline - ($delta * $current-
Component) + ($delta) else $baseline"
            />
        </xsl:call-template>
    </xsl:template>


    <xsl:template name="componentAttachment">
        <xsl:param name="use" select="false()" as="xs:boolean"/>
        <xsl:param name="countComponents"/>
        <xsl:param name="baseline"/>
        <xsl:param name="certainty" select="100"/>
        <xsl:param name="onlySewn" select="'no'"/>
       <!-- Only draw the sewn attachment and not the glued option for single leaves
 and all units but textHook if there is a integral pastedown -->
        <xsl:variable name="onlySewn"
            select="if (($onlySewn eq 'yes') or
                                (ancestor::yes/type/integral/pastedown/yes   and
boolean(not(type/hook/type[textHook]))
            )) then 'yes' else 'no'"/>
        <xsl:variable name="rotation"
            select="if (boolean(
            (pastedown/yes and $use eq false() and $countComponents le 3)
           or (following-sibling::component/pastedown/yes  and $use eq false() and
 $countComponents le 3)
           or (preceding-sibling::component/pastedown/yes  and $use eq false() and
 $countComponents le 3)
            )) then 45 else 0"/>
        <xsl:choose>
            <xsl:when test="./attachment[sewn]">
                <desc xmlns="http://www.w3.org/2000/svg">Sewn component</desc>
                <xsl:choose>
                    <xsl:when test="./type/hook/type[textHook]">
                        <xsl:call-template name="sewnComponent-textHook">
                            <xsl:with-param name="countComponents" select="$count-
Components"/>
                            <xsl:with-param name="baseline" select="$baseline"/>
                        </xsl:call-template>
                    </xsl:when>
                    <xsl:when test="./type/hook/type[endleafHook]">
                        <xsl:call-template name="sewnComponent-endleafHook">
                            <xsl:with-param name="countComponents" select="$count-
Components"/>
                            <xsl:with-param name="baseline" select="$baseline"/>
                            <xsl:with-param name="rotation" select="$rotation"/>
                        </xsl:call-template>
                    </xsl:when>
                    <xsl:otherwise>
                        <xsl:call-template name="sewnComponent">
                            <xsl:with-param name="countComponents" select="$count-
Components"/>
                            <xsl:with-param name="baseline" select="$baseline"/>
                            <xsl:with-param name="rotation" select="$rotation"/>
                        </xsl:call-template>
                    </xsl:otherwise>
                </xsl:choose>
            </xsl:when>
```

```
            <xsl:when test="./attachment[glued]">
                <desc xmlns="http://www.w3.org/2000/svg">Glued component</desc>
                <xsl:choose>
                    <xsl:when test="./type/hook/type[textHook]">
                        <xsl:call-template name="gluedComponent-textHook">
                            <xsl:with-param name="baseline" select="$baseline"/>
                        </xsl:call-template>
                    </xsl:when>
                    <xsl:otherwise>
                        <xsl:choose>
                            <xsl:when test="$onlySewn eq 'no'">
                               <!-- Only draw the sewn attachment and not the glued
 option for single leaves and all units but textHook if there is a integral pastedown
 -->
                                <xsl:call-template name="gluedComponent">
                                            <xsl:with-param name="baseline" se-
lect="$baseline"/>
                                    <xsl:with-param name="certainty" select="$cer-
tainty"
                                        as="xs:integer"/>
                                </xsl:call-template>
                            </xsl:when>
                        </xsl:choose>
                    </xsl:otherwise>
                </xsl:choose>
            </xsl:when>
            <xsl:otherwise>
                  <desc xmlns="http://www.w3.org/2000/svg">Attachment method not
checked, not known,
                    or other</desc>
            </xsl:otherwise>
        </xsl:choose>
    </xsl:template>

    <xsl:template name="sewnComponent">
        <xsl:param name="countComponents"/>
        <xsl:param name="baseline"/>
        <xsl:param name="rotation" select="0"/>
        <g xmlns="http://www.w3.org/2000/svg">
            <path>
                <xsl:attribute name="class">
                    <xsl:text>thread</xsl:text>
                </xsl:attribute>
                <xsl:attribute name="transform">
                    <xsl:text>rotate(</xsl:text>
                    <xsl:value-of select="-$rotation"/>
                    <xsl:text>,</xsl:text>
                    <xsl:value-of select="$Ax + ($delta * $countComponents) - 2"/>
                    <xsl:text>,</xsl:text>
                     <xsl:value-of select="$baseline - ($delta * $countComponents)
- 5"/>
                    <xsl:text>)</xsl:text>
                </xsl:attribute>
                <xsl:attribute name="d">
                    <xsl:text>M</xsl:text>
                    <xsl:value-of select="$Ax + ($delta * $countComponents) - 2"/>
                    <xsl:text>,</xsl:text>
                     <xsl:value-of select="$baseline - ($delta * $countComponents)
- 5"/>
                    <xsl:text>&#32;L</xsl:text>
                    <xsl:value-of select="$Ax - ($delta * 1.5)"/>
                    <xsl:text>,</xsl:text>
                     <xsl:value-of select="$baseline - ($delta * $countComponents)
- 5"/>
                </xsl:attribute>
```

```
            </path>
        </g>
    </xsl:template>

    <xsl:template name="sewnComponent-endleafHook">
        <xsl:param name="countComponents"/>
        <xsl:param name="baseline"/>
        <xsl:param name="rotation" select="0"/>
        <g xmlns="http://www.w3.org/2000/svg">
            <path>
                <xsl:attribute name="class">
                    <xsl:text>thread</xsl:text>
                </xsl:attribute>
                <xsl:attribute name="transform">
                    <xsl:text>rotate(</xsl:text>
                    <xsl:value-of select="-$rotation"/>
                    <xsl:text>,</xsl:text>
                    <xsl:value-of select="$Ax + ($delta * $countComponents) - 2"/>
                    <xsl:text>,</xsl:text>
                     <xsl:value-of select="$baseline - ($delta * $countComponents)
- 5"/>
                    <xsl:text>)</xsl:text>
                </xsl:attribute>
                <xsl:attribute name="d">
                    <xsl:text>M</xsl:text>
                    <xsl:value-of select="$Ax + ($delta * $countComponents) - 2"/>
                    <xsl:text>,</xsl:text>
                     <xsl:value-of select="$baseline - ($delta * $countComponents)
- 5"/>
                    <xsl:text>&#32;L</xsl:text>
                    <xsl:value-of select="$Ax - ($delta * 1.2)"/>
                    <xsl:text>,</xsl:text>
                     <xsl:value-of select="$baseline - ($delta * $countComponents)
- 5"/>
                </xsl:attribute>
            </path>
        </g>
    </xsl:template>

    <xsl:template name="sewnComponent-textHook">
        <xsl:param name="countComponents"/>
        <xsl:param name="baseline"/>
        <g xmlns="http://www.w3.org/2000/svg">
            <path>
                <xsl:attribute name="class">
                    <xsl:text>thread</xsl:text>
                </xsl:attribute>
                <xsl:attribute name="d">
                    <xsl:text>M</xsl:text>
                    <xsl:value-of select="$Ax + $delta + 10 - 3"/>
                    <xsl:text>,</xsl:text>
                    <xsl:value-of select="$baseline + 10"/>
                    <xsl:text>&#32;L</xsl:text>
                    <xsl:value-of select="$Ax - ($delta * 2)"/>
                    <xsl:text>,</xsl:text>
                    <xsl:value-of select="$baseline + 10"/>
                </xsl:attribute>
            </path>
        </g>
    </xsl:template>

    <xsl:template name="gluedComponent">
        <xsl:param name="baseline"/>
        <xsl:param name="certainty" select="100"/>
        <xsl:variable name="componentID" select="generate-id()"/>
```

```
            <mask xmlns="http://www.w3.org/2000/svg">
                <xsl:attribute name="id">
                    <xsl:value-of select="concat('fademask', $componentID)"/>
                </xsl:attribute>
                <rect>
                    <xsl:attribute name="width">
                        <xsl:value-of select="$delta * 2.5"/>
                    </xsl:attribute>
                    <xsl:attribute name="height">
                        <xsl:value-of select="$delta"/>
                    </xsl:attribute>
                    <xsl:attribute name="x">
                        <xsl:value-of select="$Ax + ($delta div 2)"/>
                    </xsl:attribute>
                    <xsl:attribute name="y">
                        <xsl:value-of select="$baseline - $delta "/>
                    </xsl:attribute>
                    <xsl:attribute name="fill">
                        <xsl:text>url(#radialFading)</xsl:text>
                    </xsl:attribute>
                    <xsl:attribute name="stroke-opacity">
                        <xsl:value-of select="0.0"/>
                    </xsl:attribute>
                </rect>
            </mask>
            <g xmlns="http://www.w3.org/2000/svg">
                <xsl:call-template name="certainty">
                    <xsl:with-param name="certainty" select="$certainty"/>
                    <xsl:with-param name="type" select="'4'"/>
                </xsl:call-template>
                <rect>
                    <xsl:attribute name="width">
                        <xsl:value-of select="$delta * 2.5"/>
                    </xsl:attribute>
                    <xsl:attribute name="height">
                        <xsl:value-of select="$delta"/>
                    </xsl:attribute>
                    <xsl:attribute name="x">
                        <xsl:value-of select="$Ax + ($delta div 2)"/>
                    </xsl:attribute>
                    <xsl:attribute name="y">
                        <xsl:value-of select="$baseline - $delta "/>
                    </xsl:attribute>
                    <xsl:attribute name="fill">
                        <xsl:text>url(#gluedPattern2)</xsl:text>
                    </xsl:attribute>
                    <xsl:attribute name="stroke-opacity">
                        <xsl:value-of select="0.0"/>
                    </xsl:attribute>
                    <xsl:attribute name="mask">
                          <xsl:value-of select="concat('url(#fademask', $componentID,
')')"/>
                    </xsl:attribute>
                </rect>
            </g>
        </xsl:template>

        <xsl:template name="gluedComponent-textHook">
            <xsl:param name="baseline"/>
            <xsl:variable name="componentID" select="generate-id()"/>
            <g xmlns="http://www.w3.org/2000/svg">
                <desc xmlns="http://www.w3.org/2000/svg">
                    <xsl:text>Glued component - textHook</xsl:text>
                </desc>
                <mask xmlns="http://www.w3.org/2000/svg">
```

```
                    <xsl:attribute name="id">
                        <xsl:value-of select="concat('fademask', $componentID)"/>
                    </xsl:attribute>
                    <rect xmlns="http://www.w3.org/2000/svg">
                        <xsl:attribute name="width">
                            <xsl:value-of select="$delta * 2.5"/>
                        </xsl:attribute>
                        <xsl:attribute name="height">
                            <xsl:value-of select="$delta"/>
                        </xsl:attribute>
                        <xsl:attribute name="x">
                            <xsl:value-of select="$Ax + 1.5*($delta)"/>
                        </xsl:attribute>
                        <xsl:attribute name="y">
                            <xsl:value-of select="$baseline + 20 "/>
                        </xsl:attribute>
                        <xsl:attribute name="fill">
                            <xsl:text>url(#radialFading)</xsl:text>
                        </xsl:attribute>
                        <xsl:attribute name="stroke-opacity">
                            <xsl:value-of select="0.0"/>
                        </xsl:attribute>
                    </rect>
                </mask>
                <rect xmlns="http://www.w3.org/2000/svg">
                    <xsl:attribute name="width">
                        <xsl:value-of select="$delta * 2.5"/>
                    </xsl:attribute>
                    <xsl:attribute name="height">
                        <xsl:value-of select="$delta"/>
                    </xsl:attribute>
                    <xsl:attribute name="x">
                        <xsl:value-of select="$Ax + 1.5*($delta)"/>
                    </xsl:attribute>
                    <xsl:attribute name="y">
                        <xsl:value-of select="$baseline + 20 "/>
                    </xsl:attribute>
                    <xsl:attribute name="fill">
                        <xsl:text>url(#gluedPattern2)</xsl:text>
                    </xsl:attribute>
                    <xsl:attribute name="stroke-opacity">
                        <xsl:value-of select="0.0"/>
                    </xsl:attribute>
                    <xsl:attribute name="mask">
                          <xsl:value-of select="concat('url(#fademask', $componentID,
')')"/>
                    </xsl:attribute>
                </rect>
            </g>
        </xsl:template>


        <!-- Titling -->
        <xsl:template name="title">
            <xsl:param name="detected" as="xs:integer" select="1"/>
            <xsl:param name="side" select="'left'"/>
            <xsl:param name="use" select="'use'"/>
            <text xmlns="http://www.w3.org/2000/svg">
                <xsl:attribute name="class">
                    <xsl:text>titleText</xsl:text>
                </xsl:attribute>
                <xsl:attribute name="x">
                    <xsl:value-of select="$Ax"/>
                </xsl:attribute>
                <xsl:attribute name="y">
```

```
                <xsl:value-of select="$Oy + 5"/>
            </xsl:attribute>
            <xsl:value-of select="$shelfmark"/>
            <xsl:text> - </xsl:text>
            <xsl:choose>
                <xsl:when test="$detected eq 0">
                    <xsl:text>endleaves not detected</xsl:text>
                </xsl:when>
                <xsl:otherwise>
                    <xsl:value-of
                        select="concat(upper-case(substring($side,1,1)),
                        substring($side, 2),
                        ' '[not(last())]
                        )"/>
                    <xsl:text> endleaves (</xsl:text>
                    <xsl:value-of
                        select="concat(upper-case(substring($use,1,1)),
                        substring($use, 2),
                        ' '[not(last())]
                        )"/>
                    <xsl:text>)</xsl:text>
                </xsl:otherwise>
            </xsl:choose>
        </text>
    </xsl:template>

    <!-- Description -->
    <xsl:template name="description">
        <xsl:param name="baseline"/>
        <xsl:param name="unknown"/>
        <xsl:variable name="baseline" select="$baseline - 10"/>
        <g xmlns="http://www.w3.org/2000/svg">
            <xsl:attribute name="class">
                <xsl:text>descText</xsl:text>
            </xsl:attribute>
            <xsl:choose>
                <xsl:when test="no | NC | NK | other">
                    <text xmlns="http://www.w3.org/2000/svg">
                        <xsl:attribute name="x">
                            <xsl:value-of select="$Ox + 20"/>
                        </xsl:attribute>
                        <xsl:attribute name="y">
                            <xsl:value-of select="$baseline"/>
                        </xsl:attribute>
                        <tspan xmlns="http://www.w3.org/2000/svg">
                            <xsl:choose>
                                <xsl:when test="no">
                                    <xsl:text>There are no endleaves</xsl:text>
                                </xsl:when>
                                <xsl:otherwise>
                                    <xsl:text>Endleaves not described</xsl:text>
                                </xsl:otherwise>
                            </xsl:choose>
                        </tspan>
                    </text>
                </xsl:when>
                <xsl:when test="$unknown eq 'other'">
                    <text xmlns="http://www.w3.org/2000/svg">
                        <xsl:attribute name="x">
                            <xsl:value-of select="$Ox + 20"/>
                        </xsl:attribute>
                        <xsl:attribute name="y">
                            <xsl:value-of select="$baseline"/>
                        </xsl:attribute>
                        <tspan xmlns="http://www.w3.org/2000/svg">
```

```
                            <xsl:value-of select="./yes/type/other/text()"/>
                        </tspan>
                    </text>
                </xsl:when>
                <xsl:otherwise>
                    <xsl:value-of select="$unknown"/>
                </xsl:otherwise>
            </xsl:choose>
            <xsl:for-each select="yes/type">
                <xsl:variable name="position" select="position()"/>
                <text xmlns="http://www.w3.org/2000/svg">
                    <xsl:attribute name="x">
                        <xsl:value-of select="$Ox + 20"/>
                    </xsl:attribute>
                    <xsl:attribute name="y">
                        <xsl:value-of select="$baseline"/>
                    </xsl:attribute>
                    <xsl:for-each select="separate">
                        <tspan xmlns="http://www.w3.org/2000/svg">
                            <xsl:attribute name="dy">
                                <xsl:value-of select="0 + 6 * ($position - 1)"/>
                            </xsl:attribute>
                            <xsl:text>Separate endleaves: </xsl:text>
                            <xsl:for-each select="units/unit">
                               <xsl:variable name="position" select="position()"/>

                                <tspan xmlns="http://www.w3.org/2000/svg">
                                    <xsl:attribute name="x">
                                        <xsl:value-of select="$Ox + 80"/>
                                    </xsl:attribute>
                                    <xsl:attribute name="dy">
                                        <xsl:value-of select="if ($position eq 1)
then 0 else 7"/>
                                    </xsl:attribute>
                                    <xsl:text>unit </xsl:text>
                                    <xsl:value-of select="position()"/>
                                    <xsl:text>: </xsl:text>
                                    <xsl:for-each select="components/component">
                                       <xsl:variable name="position" select="posi-
tion()"/>
                                        <tspan xmlns="http://www.w3.org/2000/svg">
                                            <xsl:attribute name="x">
                                                <xsl:attribute name="x">
                                                 <xsl:value-of select="$Ox + 100"/>

                                                </xsl:attribute>
                                            </xsl:attribute>
                                            <xsl:attribute name="dy">
                                                <xsl:value-of
                                                  select="if ($position eq 1) then
 0 else 7"/>
                                            </xsl:attribute>
                                            <xsl:text>component </xsl:text>
                                            <xsl:value-of select="position()"/>
                                            <xsl:text>: </xsl:text>
                                         <tspan xmlns="http://www.w3.org/2000/svg">

                                                <xsl:attribute name="dx">
                                                  <xsl:text>2pt</xsl:text>
                                                </xsl:attribute>
                                                <xsl:choose>
                                                  <xsl:when test="type/hook">
                                                  <xsl:value-of
                                                                                    se-
lect="type/hook/type/node()/name()"/>
```

```
                                                  <xsl:choose>
                                                                        <xsl:when
test="type/hook/node()/double/yes">
                                                  <xsl:text> double</xsl:text>
                                                  </xsl:when>
                                                  </xsl:choose>
                                                  </xsl:when>
                                                   <xsl:when test="type[NC | NK |
other]">
                                                  <xsl:text>type </xsl:text>
                                                  <xsl:choose>
                                                  <xsl:when test="type/NC">
                                                  <xsl:text>not checked</xsl:text>
                                                  </xsl:when>
                                                  <xsl:when test="type/NK">
                                                  <xsl:text>not known</xsl:text>
                                                  </xsl:when>
                                                  <xsl:when test="type/other">
                                                  <xsl:text>other</xsl:text>
                                                  </xsl:when>
                                                  </xsl:choose>
                                                  </xsl:when>
                                                  <xsl:otherwise>
                                                                <xsl:value-of se-
lect="type/node()/name()"/>
                                                  </xsl:otherwise>
                                                </xsl:choose>
                                                <xsl:text> (</xsl:text>
                                                <xsl:choose>
                                                  <xsl:when test="pastedown/yes">
                                                  <xsl:text> pastedown, </xsl:text>

                                                  </xsl:when>
                                                  <xsl:when test="pastedown[NC | NK
 | other]">
                                                  <xsl:text> pastedown: </xsl:text>

                                                                <xsl:value-of se-
lect="pastedown/node()/name()"/>
                                                  <xsl:text>, </xsl:text>
                                                  </xsl:when>
                                                </xsl:choose>
                                                <xsl:choose>
                                                   <xsl:when test="material[NC | NK
 | other]">
                                                  <xsl:text>material: </xsl:text>
                                                  <xsl:choose>
                                                  <xsl:when test="material/NC">
                                                  <xsl:text>not checked</xsl:text>
                                                  </xsl:when>
                                                  <xsl:when test="material/NK">
                                                  <xsl:text>not known</xsl:text>
                                                  </xsl:when>
                                                  <xsl:when test="material/other">
                                                <xsl:value-of select="material/oth-
er/text()"/>
                                                  </xsl:when>
                                                  </xsl:choose>
                                                  <xsl:text>, </xsl:text>
                                                  </xsl:when>
                                                  <xsl:otherwise>
                                                     <xsl:value-of select="materi-
al/node()/name()"/>
                                                                <tspan xmlns="ht-
tp://www.w3.org/2000/svg"
```

```
                                                dx="-1pt">
                                                <xsl:text>, </xsl:text>
                                                </tspan>
                                                </xsl:otherwise>
                                              </xsl:choose>
                                              <xsl:choose>
                                                <xsl:when test="attachment[NC |
NK | other]">
                                               <xsl:text>attachment: </xsl:text>

                                                <xsl:choose>
                                                <xsl:when test="attachment/NC">
                                                <xsl:text>not checked</xsl:text>
                                                </xsl:when>
                                                <xsl:when test="attachment/NK">
                                                <xsl:text>not known</xsl:text>
                                                </xsl:when>
                                               <xsl:when test="attachment/other">

                                            <xsl:value-of select="attachment/oth-
er/text()"/>
                                                </xsl:when>
                                                </xsl:choose>
                                                </xsl:when>
                                                <xsl:otherwise>
                                                   <xsl:value-of select="attach-
ment/node()/name()"/>
                                                </xsl:otherwise>
                                              </xsl:choose>
                                                              <tspan xmlns="ht-
tp://www.w3.org/2000/svg" dx="0pt">
                                                <xsl:text>)</xsl:text>
                                              </tspan>
                                          </tspan>
                                      </tspan>
                                  </xsl:for-each>
                              </tspan>
                          </xsl:for-each>
                      </tspan>
                  </xsl:for-each>
                  <xsl:for-each select="integral">
                      <xsl:variable name="position"
                                    select="if ($position eq 1) then 1 else 1.5 +
count(parent::type/preceding-sibling::type/separate/units/unit/components/compon-
ent)"/>
                      <tspan xmlns="http://www.w3.org/2000/svg">
                          <xsl:attribute name="dy">
                              <xsl:value-of select="concat(0 + 6 * ($position -
1), 'pt')"/>
                          </xsl:attribute>
                          <xsl:text>Integral endleaves: </xsl:text>
                          <tspan xmlns="http://www.w3.org/2000/svg">
                              <xsl:attribute name="x">
                                  <xsl:value-of select="$Ox + 80"/>
                              </xsl:attribute>
                              <xsl:value-of select="numberOfLeaves"/>
                              <xsl:choose>
                                  <xsl:when test="xs:integer(numberOfLeaves) gt
1">
                                      <xsl:text> leaves</xsl:text>
                                  </xsl:when>
                                  <xsl:otherwise>
                                      <xsl:text> leaf</xsl:text>
                                  </xsl:otherwise>
                              </xsl:choose>
```

```
                                <xsl:choose>
                                    <xsl:when test="pastedown/yes">
                                        <xsl:text> (pastedown)</xsl:text>
                                    </xsl:when>
                                </xsl:choose>
                            </tspan>
                        </tspan>
                    </xsl:for-each>
                </text>
            </xsl:for-each>
        </g>
    </xsl:template>

    <!-- Uncertainty template -->
    <xsl:template name="certainty">
        <xsl:param name="certainty" select="100" as="xs:integer"/>
        <xsl:param name="uncertaintyIncrement"/>
        <xsl:param name="type"/>
        <xsl:choose>
            <xsl:when test="$type = '1'">
                <xsl:choose>
                    <xsl:when test="$certainty lt 100">
                        <xsl:attribute name="filter">
                            <xsl:text>url(#f1)</xsl:text>
                        </xsl:attribute>
                    </xsl:when>
                </xsl:choose>
            </xsl:when>
            <xsl:when test="$type = '2'">
                <xsl:choose>
                    <xsl:when test="$certainty lt 100">
                        <xsl:attribute name="filter">
                            <xsl:text>url(#f2)</xsl:text>
                        </xsl:attribute>
                    </xsl:when>
                </xsl:choose>
            </xsl:when>
            <xsl:when test="$type = '3'">
                <xsl:choose>
                    <xsl:when test="$certainty lt 100">
                        <xsl:attribute name="filter">
                            <xsl:text>url(#f3)</xsl:text>
                        </xsl:attribute>
                    </xsl:when>
                </xsl:choose>
            </xsl:when>
            <xsl:when test="$type = '4'">
                <xsl:choose>
                    <xsl:when test="$certainty lt 100">
                        <xsl:attribute name="filter">
                            <xsl:text>url(#f4)</xsl:text>
                        </xsl:attribute>
                    </xsl:when>
                </xsl:choose>
            </xsl:when>
        </xsl:choose>
    </xsl:template>

</xsl:stylesheet>
```

Endleaf master SVG

This section contains the master SVG coding written for this project for endleaf transformations. This particular file makes available to the XSLT stylesheet a series of filters and shaped masks to be applied when needed to endleaf diagrams.

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet href="../CSS/style.css" type="text/css"?>
<!DOCTYPE svg
  PUBLIC "-//W3C//DTD SVG 1.1//EN" "http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd">
<svg xmlns="http://www.w3.org/2000/svg" xmlns:xlink="http://www.w3.org/1999/xlink"
 version="1.1"
    x="0" y="0" width="297mm" height="210mm" viewBox="0 0 297 210"
    preserveAspectRatio="xMinYMin meet">
    <defs>
        <filter id="f1" filterUnits="userSpaceOnUse">
            <feColorMatrix in="SourceGraphic" type="matrix" values="
                0.0 0.0 0.0 0.0 0.0
                0.0 0.0 0.0 0.0 0.0
                0.0 0.0 0.0 0.0 0.0
                0.0 0.0 0.0 0.6 0.0"/>
            <feGaussianBlur stdDeviation="0.5 0.1"/>
        </filter>
        <filter id="f2" filterUnits="userSpaceOnUse">
            <feColorMatrix in="SourceGraphic" type="matrix" values="
                0.0 0.0 0.0 0.0 0.0
                0.0 0.0 0.0 0.0 0.0
                0.0 0.0 0.0 0.0 0.0
                0.0 0.0 0.0 0.65 0.0"/>
            <feGaussianBlur stdDeviation="0.5 0.3"/>
        </filter>
        <filter id="f3" filterUnits="userSpaceOnUse">
            <feColorMatrix in="SourceGraphic" type="matrix" values="
                0.0 0.0 0.0 0.0 0.0
                0.0 0.0 0.0 0.0 0.0
                0.0 0.0 0.0 0.0 0.0
                0.0 0.0 0.0 0.65 0.0"/>
            <feGaussianBlur stdDeviation="0.5"/>
        </filter>
        <filter id="f4" filterUnits="userSpaceOnUse">
            <feColorMatrix in="SourceGraphic" type="matrix" values="
                0.0 0.0 0.0 0.0 0.0
                0.0 0.0 0.0 0.0 0.0
                0.0 0.0 0.0 0.0 0.0
                0.0 0.0 0.0 0.65 0.0"/>
            <feGaussianBlur stdDeviation="0 0.5"/>
        </filter>
        <linearGradient id="fading">
            <stop offset="0%" stop-color="#FFFFFF"/>
            <stop offset="50%" stop-color="#666666"/>
            <stop offset="90%" stop-color="#000000"/>
        </linearGradient>
        <linearGradient id="fading2">
            <stop offset="0%" stop-color="#000000"/>
            <stop offset="50%" stop-color="#666666"/>
            <stop offset="90%" stop-color="#FFFFFF"/>
        </linearGradient>
        <radialGradient id="radialFading">
            <stop offset="0%" stop-color="#FFFFFF"/>
            <stop offset="50%" stop-color="#666666"/>
            <stop offset="90%" stop-color="#000000"/>
        </radialGradient>
        <g id="outermostGL">
            <desc>Outermost gathering Left</desc>
            <path d="M 140,0 L 10,0 A 10,10 0 0,0 10,20 L 140,20 z"/>
        </g>
       <pattern id="gluedPattern" patternUnits="userSpaceOnUse" x="2" y="0" width="5"
 height="10"
            viewBox="0 0 10 10" xlink:type="simple" xlink:show="other" xlink:actu-
ate="onLoad"
```

```
            preserveAspectRatio="xMidYMid meet">
            <desc>Glue pattern</desc>
            <path d="M 0,5 L 3,0 " class="glued"/>
        </pattern>
           <pattern id="gluedPattern2" patternUnits="userSpaceOnUse" x="0" y="0"
width="8" height="8">
            <g fill="none" class="glued" stroke-linecap="round">
                <g stroke="#888888" stroke-width="0.5">
                    <path d="M0,0 L8,8"/>
                    <path d="M2,0 L8,6"/>
                    <path d="M4,0 L8,4"/>
                    <path d="M6,0 L8,2"/>
                    <path d="M8,0 L8,0"/>
                    <path d="M0,2 L6,8"/>
                    <path d="M0,4 L4,8"/>
                    <path d="M0,6 L2,8"/>
                    <path d="M0,8 L0,8"/>
                </g>
            </g>
        </pattern>
        <g id="pastedown" class="line"  stroke-linecap="round">
            <desc>pastedown</desc>
            <path d="M 0,0 L 130,0"/>
               <rect width="130" height="5" fill="url(#gluedPattern)" stroke-opa-
city="0.0"/>
        </g>
        <rect id="pasted" width="130" height="5" fill="url(#gluedPattern)" stroke-
opacity="0.0"/>
        <clipPath id="guardClip">
            <rect id="rect1" width="32.5" height="10" x="97.5" y="-2"
                style="stroke: gray; fill: none;"/>
        </clipPath>
        <clipPath id="outsideHookClip">
            <rect id="rect12" width="16.25" height="10" x="113.75" y="-2"
                style="stroke: gray; fill: none;"/>
        </clipPath>
    </defs>

   <!-- Call the definitions above inside the groups below by <use  xlink:href="[#+ID
 here]", use the X and Y attributes of the use element to move the components around
 -->

</svg>
```

Endleaf CSS

This section contains the CSS code written for this project to rule the appearance of lines within endleaf diagrams.

```
/* style.css */

/*
 *    GENERAL   *
 *
 * stroke=#000000; stroke-width=1; fill=none;
 */

.titleText{
    font-family: 'SimonciniGaramond LT', garamond, serif;
    font-style: normal;
    font-weight: bold;
    font-size: 10pt;
    dominant-baseline: alphabetic;
    text-anchor: middle;
}

.descText{
    font-family: 'SimonciniGaramond LT', garamond, serif;
    font-style: normal;
    font-weight: normal;
    font-size: 5pt;
    dominant-baseline: alphabetic;
    text-anchor: start;
}

.line{
    stroke:#000000;
    stroke-width:1pt;
    fill:none;
}

.line2{
    stroke:#000000;
    stroke-width:2pt;
    fill:none;
}

.line_ref{
    stroke:#999999;
    stroke-width:0.5pt;
    fill:none;
}

.thread{
    stroke:#000000;
    stroke-width:0.5;
    fill:none;
}
```