

**An Open Design for Computer-Aided Algorithmic Music Composition:
athenaCL**

by

Christopher Ariza

ISBN: 1-58112-292-6

DISSERTATION.COM



Boca Raton, Florida
USA • 2005

An Open Design for Computer-Aided Algorithmic Music Composition: athenaCL

Copyright © 2005 Christopher Ariza
All rights reserved.

Dissertation.com
Boca Raton, Florida
USA • 2005

ISBN: 1-58112-292-6

An Open Design for Computer-Aided Algorithmic Music Composition: athenaCL

by

Christopher Ariza

A dissertation submitted in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

Department of Music

New York University

September, 2005

Advisor: Elizabeth Hoffman

© Christopher Ariza

All Rights Reserved, 2005

Acknowledgments

Thanks foremost to my parents, Deborah and Augusto Ariza, for their boundless faith, encouragement, and support over the many years of my studies. Thanks also to my brother, Philip, and the rest of my family for their support. Much gratitude to Shruti Mahindrakar for her encouragement, solace, and patience over many years.

My work in computer music, and the existence of athenaCL, are directly the result of the teaching and encouragement of my advisor, Elizabeth Hoffman. Had she not given me access to new tools, today I would likely have only graphite and paper. Had she not encouraged the development and presentation of athenaCL, today I would have a bundle of incomprehensible code fragments. Her comments on my compositions, papers, and this text have been invaluable. For these reasons and more, I am particularly grateful for her dedication and support over the duration of my graduate education.

It is difficult to imagine how I could have completed this document had it not been for the teaching and experience of Paul Berg. His seminar in algorithmic composition, as well as our many conversations, helped me to refine many of my ideas, to lead me in better directions, and to ask better questions. His assistance with the literature, the history of the Institute of Sonology, and this text were a great benefit. His AC Toolbox provided numerous valuable ideas for refinements of athenaCL. I am very grateful for the time he devoted to helping me with this project.

Thanks to Robert Rowe for his assistance and advice with my previous research and this text. Additional thanks to Jairo Moreno, David Burrows, and Louis Karchin for serving as committee readers. Thanks to Ruth and Gottfried Michael Koenig for their hospitality, time, and conversations. Thanks to the following people for discussing their research: R. Lundbeck, Max Mathews, and John F. Sowa. Thanks to Nicole Falque (Shell Coordination

Centre S.A., Monnet Centre International Laboratory, Louvain-La-Neuve, Belgium), Lida van der Horst and Anneke Schelvis (Shell Global Solutions International B.V., Shell Research and Technology Centre, Amsterdam, the Netherlands), and Nicole Simpson (The New York Public Library, New York) for research assistance. I am also grateful to the anonymous reviewers of the 2005 International Computer Music Conference (ICMC) for comments on an article derived from components of Chapter 3 (Ariza 2005b).

Thanks to my composition teachers for wisdom and guidance over the course of my education: Mario Davidovsky, Michael Gandolfi, Elizabeth Hoffman, David Horne, Louis Karchin, and Jeff Nichols. Thanks to Joseph N. Straus for inspiring many of the theoretical components of athenaCL, as well as many valuable discussions concerning nomenclature, post-tonal theory, and his voice leading model. Much gratitude as well to Jacob T. Schwartz for suggesting, many years ago, that I work in Python rather than C. Thanks to the faculty and administration of the New York University Graduate School of Arts and Sciences Music Department for their flexibility in letting me pursue my academic interests. Particular thanks to Rena Mueller and Pauline Lum, as well as former and current department chairs Gage Averill and Michael Beckerman. Also, I am grateful for the department's support of the Washington Square Computer Music Studio, graduate course offerings in Computer Music, and the Electro-Acoustic Music concert series; all were essential for promoting my research and music.

Thanks to the United States Fulbright program, the Institute for International Education (IIE), and the Netherlands-America Foundation (NAF) for providing funds and support for my work at the Institute of Sonology in Den Haag, the Netherlands. My work and research at the Institute, as well as its students, faculty, and technical resources, were of great importance for this text, my software, and my work as a composer.

Thanks to the following people for reading early versions of this document and providing comments: Augusto Ariza, Rachel Coupe, Paula Matthusen, and Red Wierenga. Thanks to my colleagues at New York University, particularly those that took the trouble to, early on, investigate, learn, and provide comments on athenaCL: Sean Carson, Ryan Dorin, Paula Mattheson, Jonathan Saggau, Jesse Sklar, and Juliana Trivers. Special thanks to the numerous unnamed persons who have supported athenaCL over the course of its development, through downloads, links, and comments. Additional thanks to my friends around the world for putting up with my ramblings about algorithmic composition for so many years, and often offering valuable advice or respite.

I am grateful to the following libraries for providing research materials and quiet spaces necessary for the composition of this text: the Library of the Courant Institute of Math and Sciences and the Elmer Holmes Bobst Library, New York University, New York; the Langson Library and the Science Library, University of California Irvine, California; the Music Division of the New York Public Library for the Performing Arts, New York; the Centraal Bibliotheek Den Haag; the Openbare Bibliotheek Amsterdam; the Universiteit Leiden Bibliotheek; and the Koninklijk Conservatorium (Instituut voor Sonologie) Bibliotheek, Den Haag. Finally, I am grateful to the authors of the open-source software tools that were used in the production and typesetting of this text: DocBook, OpenJade, Python, and ImageMagick.

The following trademark information applies throughout this text. Apple, Macintosh, Mac OS, and QuickTime are trademarks or registered trademarks of Apple Computer, Inc. Finale is a trademark of MakeMusic! Inc. Java and JavaScript are trademarks or registered trademarks of Sun Microsystems, Inc. Linux is a trademark of Linus Torvalds. Max/MSP is a trademark of Cycling '74. Microsoft Windows and Visual Basic are trademarks or registered trademarks of Microsoft, Inc. PDP, VT05, and VT100 are trademarks of Compaq.

PDF and PostScript are trademarks of Adobe, Inc. Reaktor is a trademark of Native Instruments. SCORE is a trademark of San Andreas Press. Sibelius is a trademark of Sibelius Software Ltd. SourceForge.net is a trademark of VA Software Corporation. UML is a trademark of the Object Management Group. UNIX is a trademark of The Open Group.

Abstract

This dissertation introduces a new design for a computer-aided algorithmic music composition system. Rather than exploring specific algorithms, this study focuses on system and component design. The design introduced here is demonstrated through its implementation in athenaCL, a modular, polyphonic, poly-paradigm algorithmic music composition system in a cross-platform interactive command-line environment. The athenaCL system offers an open-source, object-oriented composition tool written in Python. The system can be scripted and embedded, and includes integrated instrument libraries, post-tonal and microtonal pitch modeling tools, multiple-format graphical outputs, and musical output in Csound, MIDI, audio file, XML, and text formats.

Software design analysis is framed within a broad historical and intertextual study of the themes, approaches, and systems of computer-aided algorithmic composition (CAAC). A detailed history of the earliest experiments, as well as analysis of the foundational CAAC systems, is provided. Common problems and interpretations of CAAC are then presented in a historical and intertextual context, drawn from the writings and systems of numerous composers and developers. Toward the goal of developing techniques of comparative software analysis, a survey of system design archetypes, based on seven descriptors of CAAC systems, is presented. With this foundation, athenaCL system components are analyzed in detail. System components are divided into abstractions of musical materials, abstractions of musical procedures, and system architecture. For each component, object models, Python examples, and diagrams are provided. Further, each component is given context in terms of its compositional implications and relation to alternative and related models from the history of CAAC.

Table of Contents

Acknowledgments	iii
Abstract	vii
List of Examples	xxvii
1. Principles and Methods.....	1
1.1. Introduction	1
1.1.1. An Open Design.....	1
1.1.2. Overview of the athenaCL System	7
1.1.3. Design Principles of the athenaCL System.....	8
1.1.4. Design Principles in Practice	11
1.1.5. Chapter Overview	16
1.2. Methods of Software Comparison and Analysis.....	17
1.2.1. Structures to be Compared and Analyzed	17
1.2.2. Object Modeling and Analysis.....	20
1.2.3. The Unified Modeling Language and Object Diagrams.....	23
1.2.4. Python and the Interactive Session.....	27
1.2.5. Conventions Used in this Text.....	34
2. Historical and Theoretical Foundations	36
2.1. Introduction	36
2.1.1. History of the First Experiments	36
2.1.2. The Early Systems of Hiller, Isaacson, and Baker.....	47
2.1.3. The Early System of Xenakis.....	53
2.1.4. The Early Systems of G. M. Koenig.....	55
2.2. The Problem of Music.....	59

2.2.1. The Unique Position of Music	59
2.2.2. The Constraint of Idiom	64
2.2.3. Nature and Musical Complexity.....	73
2.3. The Parameterization of Events.....	80
2.3.1. Music as Discrete Information.....	80
2.3.2. The Division of Musical Instruction and Production.....	83
2.4. Formalism and Algorithms	87
2.4.1. Empirical Music Theory.....	87
2.4.2. The Myth of Algorithmic Integrity.....	90
2.5. Problems of Mechanized Composition.....	94
2.5.1. Interpretation and Abundance	94
2.5.2. The Liberation of a Means of Compositional Production.....	100
2.5.3. Fear and Distributed Authorship.....	102
3. The Landscape of Composition Systems	108
3.1. Introduction	108
3.1.1. A Definition of a Computer-Aided Algorithmic Composition System	108
3.1.2. Research in Categorizing Composition Systems.....	112
3.2. Primary Descriptors	114
3.2.1. The Difficulty of Distinctions	114
3.2.2. Scale: Micro and Macro Structures	115
3.2.3. Process Model: Real-Time and Non-Real-Time.....	117
3.2.4. Idiom Affinity: Singular and Plural.....	118
3.2.5. Extensibility: Closed and Open.....	119
3.2.6. Event Production: Generation and Transformation.....	120
3.2.7. Sound Source: Internal, Exported, Imported, External	121

3.2.8. User Environment: Language, Batch, Interactive.....	122
3.3. A Classification of Systems	126
3.3.1. The Primacy of User Environment	126
3.3.2. Specialized Text Languages.....	126
3.3.3. Specialized Graphic Languages	129
3.3.4. Batch Processors	132
3.3.5. Interactive Text Interface Systems.....	134
3.3.6. Interactive Graphic Interface Systems	136
4. Abstractions of Musical Materials	138
4.1. Introduction	138
4.2. The Pitch Model	139
4.2.1. The Pitch	140
4.2.1.1. Model and Representation.....	140
4.2.1.2. Object Design.....	143
4.2.1.3. Diagrams and Examples.....	144
4.2.2. The Multiset	146
4.2.2.1. Model and Representation.....	146
4.2.2.2. Object Design.....	148
4.2.2.3. Diagrams and Examples.....	150
4.2.3. The Path.....	152
4.2.3.1. Model and Representation.....	152
4.2.3.2. Object Design.....	155
4.2.3.3. Diagrams and Examples.....	156
4.2.4. Compositional Implications.....	159
4.2.5. Alternative and Related Models	160

4.3. The Rhythm Model	163
4.3.1. The Pulse	163
4.3.1.1. Model and Representation.....	163
4.3.1.2. Object Design.....	165
4.3.1.3. Diagrams and Examples.....	166
4.3.2. The Rhythm	167
4.3.2.1. Model and Representation.....	167
4.3.2.2. Object Design.....	168
4.3.2.3. Diagrams and Examples.....	168
4.3.3. Compositional Implications.....	170
4.3.4. Alternative and Related Models	172
4.4. The Event, SubEvent, and EventSequence Model	175
4.4.1. The Event and SubEvent.....	176
4.4.2. The EventSequence	178
4.4.2.1. Model and Representation.....	178
4.4.2.2. Object Design.....	178
4.4.2.3. Diagrams and Examples.....	179
4.4.3. Compositional Implications.....	181
4.4.4. Alternative and Related Models	183
4.5. The Instrument and Orchestra Model	187
4.5.1. The Instrument.....	190
4.5.1.1. Model and Representation.....	190
4.5.1.2. Object Design.....	190
4.5.1.3. Diagrams and Examples.....	191
4.5.2. The Orchestra.....	194

4.5.2.1. Model and Representation.....	194
4.5.2.2. Object Design.....	194
4.5.2.3. Diagrams and Examples.....	195
4.5.3. Compositional Implications.....	197
4.5.4. Alternative and Related Models	198
5. Abstractions of Musical Procedures	201
5.1. Introduction	201
5.2. The One-Dimensional Generator and Transformer.....	205
5.2.1. The Selector.....	207
5.2.1.1. Model and Representation.....	207
5.2.1.2. Object Design.....	208
5.2.1.3. Diagrams and Examples.....	208
5.2.2. The Generator ParameterObject	210
5.2.2.1. Model and Representation.....	210
5.2.2.2. Object Design.....	210
5.2.2.3. Diagrams and Examples.....	212
5.2.3. The Rhythm Generator ParameterObject	214
5.2.3.1. Model and Representation.....	214
5.2.3.2. Object Design.....	214
5.2.3.3. Diagrams and Examples.....	215
5.2.4. The Filter ParameterObject	217
5.2.4.1. Model and Representation.....	217
5.2.4.2. Object Design.....	217
5.2.4.3. Diagrams and Examples.....	218
5.2.5. The Static ParameterObject.....	219

5.2.5.1. Model and Representation.....	219
5.2.5.2. Object Design.....	220
5.2.5.3. Diagrams and Examples.....	220
5.2.6. Compositional Implications.....	222
5.2.7. Alternative and Related Models	225
5.3. The Multi-Dimensional Generator and Transformer	227
5.3.1. The Temperament.....	228
5.3.1.1. Model and Representation.....	228
5.3.1.2. Object Design.....	229
5.3.1.3. Diagrams and Examples.....	229
5.3.2. The Texture.....	231
5.3.2.1. Model and Representation.....	231
5.3.2.2. Object Design.....	232
5.3.2.3. Diagrams and Examples.....	237
5.3.3. The Clone.....	242
5.3.3.1. Model and Representation.....	242
5.3.3.2. Object Design.....	243
5.3.3.3. Diagrams and Examples.....	246
5.3.4. Compositional Implications.....	249
5.3.5. Alternative and Related Models	251
6. System Architecture	258
6.1. Introduction	258
6.2. The Architecture of athenaCL.....	260
6.2.1. The External.....	260
6.2.2. The AthenaObject.....	261

6.2.2.1. Model and Representation.....	261
6.2.2.2. Object Design.....	261
6.2.2.3. Diagrams and Examples.....	262
6.2.3. The Performer and PolySequence.....	265
6.2.4. The OutputFormat.....	265
6.2.5. The OutputEngine.....	266
6.2.5.1. Model and Representation.....	266
6.2.5.2. Object Design.....	267
6.2.5.3. Diagrams and Examples.....	270
6.2.6. The EventMode.....	272
6.2.6.1. Model and Representation.....	272
6.2.6.2. Object Design.....	273
6.2.6.3. Diagrams and Examples.....	274
6.2.7. Compositional Implications.....	276
6.3. The Interface of athenaCL.....	277
6.3.1. The Command Object.....	279
6.3.1.1. Model and Representation.....	279
6.3.1.2. Object Design.....	279
6.3.1.3. Diagrams and Examples.....	281
6.3.2. The Terminal.....	284
6.3.3. The Interpreter.....	285
6.3.3.1. Model and Representation.....	285
6.3.3.2. Object Design.....	285
6.3.3.3. Diagrams and Examples.....	286
6.3.4. Compositional Implications.....	288

7. Postscript.....	291
A. Notes on Included Compositions	295
B. ParameterObject Reference and Examples	296
B.1. Generator ParameterObjects	296
B.1.1. accumulator (a).....	296
B.1.2. analysisSelect (as)	297
B.1.3. basketGen (bg).....	297
B.1.4. breakPointLinear (bpl)	298
B.1.5. breakPointPower (bpp).....	299
B.1.6. constant (c)	301
B.1.7. constantFile (cf)	301
B.1.8. cyclicGen (cg).....	301
B.1.9. directorySelect (ds)	302
B.1.10. fibonacciSeries (fs).....	303
B.1.11. henonBasket (hb).....	304
B.1.12. iterateGroup (ig)	306
B.1.13. iterateHold (ih).....	307
B.1.14. iterateWindow (iw)	309
B.1.15. lorenzBasket (lb).....	310
B.1.16. logisticMap (lm)	312
B.1.17. mask (m).....	313
B.1.18. markovGeneratorAnalysis (mga).....	315
B.1.19. markovValue (mv)	317
B.1.20. noise (n).....	317
B.1.21. operatorAdd (oa)	319

B.1.22. operatorDivide (od).....	320
B.1.23. operatorMultiply (om).....	320
B.1.24. operatorPower (op)	321
B.1.25. operatorSubtract (os).....	321
B.1.26. pathRead (pr).....	322
B.1.27. quantize (q)	323
B.1.28. randomBeta (rb).....	324
B.1.29. randomBilateralExponential (rbe).....	325
B.1.30. randomCauchy (rc).....	326
B.1.31. randomExponential (re).....	327
B.1.32. randomGauss (rg).....	329
B.1.33. randomInverseExponential (rie)	330
B.1.34. randomInverseLinear (ril).....	331
B.1.35. randomInverseTriangular (rit)	332
B.1.36. randomLinear (rl).....	332
B.1.37. randomTriangular (rt)	333
B.1.38. randomUniform (ru)	334
B.1.39. randomWeibull (rw)	335
B.1.40. sieveFunnel (sf)	336
B.1.41. sieveList (sl)	338
B.1.42. sampleSelect (ss)	338
B.1.43. valueSieve (vs)	339
B.1.44. waveCosine (wc).....	341
B.1.45. wavePulse (wp).....	342
B.1.46. wavePowerDown (wpd).....	343

B.1.47. wavePowerUp (wpu).....	344
B.1.48. waveSine (ws).....	345
B.1.49. waveSawDown (wsd).....	346
B.1.50. waveSawUp (wsu).....	347
B.1.51. waveTriangle (wt).....	348
B.2. Rhythm ParameterObjects	349
B.2.1. binaryAccent (ba).....	349
B.2.2. convertSecond (cs).....	349
B.2.3. convertSecondTriple (cst).....	350
B.2.4. gaRhythm (gr).....	351
B.2.5. loop (l).....	352
B.2.6. markovPulse (mp).....	353
B.2.7. markovRhythmAnalysis (mra).....	354
B.2.8. pulseSieve (ps).....	356
B.2.9. pulseTriple (pt).....	358
B.2.10. rhythmSieve (rs).....	359
B.3. Filter ParameterObjects	360
B.3.1. bypass (b).....	360
B.3.2. filterAdd (fa).....	361
B.3.3. filterMultiplyAnchor (fma).....	362
B.3.4. orderBackward (ob).....	363
B.3.5. orderRotate (or).....	364
B.3.6. pipeLine (pl).....	364
B.3.7. replace (r).....	365
B.4. TextureStatic ParameterObjects	366

B.4.1. levelFieldMonophonic (lfm).....	366
B.4.2. levelFieldPolyphonic (lfp).....	366
B.4.3. levelOctaveMonophonic (lom).....	367
B.4.4. levelOctavePolyphonic (lop).....	367
B.4.5. loopWithinSet (lws)	367
B.4.6. maxTimeOffset (mto)	368
B.4.7. nonRedundantSwitch (nrs).....	368
B.4.8. ornamentLibrarySelect (ols)	368
B.4.9. ornamentMaxDensity (omd).....	369
B.4.10. parallelMotionList (pml).....	369
B.5. CloneStatic ParameterObjects.....	369
B.5.1. retrogradeMethodToggle (rmt).....	369
B.5.2. timeReferenceSource (trs)	370
C. Temperament and TextureModule Reference	371
C.1. Temperaments.....	371
C.1.1. Temperament Interleave24Even	371
C.1.2. Temperament Interleave24Odd	371
C.1.3. Temperament Just	371
C.1.4. Temperament MeanTone.....	371
C.1.5. Temperament NoiseHeavy	371
C.1.6. Temperament NoiseLight	371
C.1.7. Temperament NoiseMedium.....	371
C.1.8. Temperament Pythagorean	371
C.1.9. Temperament Split24Lower.....	372
C.1.10. Temperament Split24Upper.....	372

C.1.11. Temperament TwelveEqual.....	372
C.2. TextureModules	372
C.2.1. TextureModule LineGroove.....	372
C.2.2. TextureModule LineCluster.....	372
C.2.3. TextureModule MonophonicOrnament	372
C.2.4. TextureModule LiteralHorizontal.....	373
C.2.5. TextureModule DroneArticulate.....	373
C.2.6. TextureModule LiteralVertical.....	373
C.2.7. TextureModule IntervalExpansion.....	373
C.2.8. TextureModule DroneSustain	373
D. OutputFormat and OutputEngine Reference.....	374
D.1. OutputFormats.....	374
D.1.1. acToolbox	374
D.1.2. audioFile	374
D.1.3. csoundBatch.....	374
D.1.4. csoundData.....	374
D.1.5. csoundOrchestra.....	374
D.1.6. csoundScore.....	374
D.1.7. maxColl.....	374
D.1.8. midiFile.....	374
D.1.9. textSpace.....	375
D.1.10. textTab.....	375
D.1.11. xmlAthenaObject.....	375
D.2. OutputEngines	375
D.2.1. EngineAcToolbox.....	375

D.2.2. EngineAudioFile	375
D.2.3. EngineCsoundExternal	375
D.2.4. EngineCsoundNative	376
D.2.5. EngineCsoundSilence.....	376
D.2.6. EngineMaxColl.....	376
D.2.7. EngineMidiFile	376
D.2.8. EngineText.....	376
E. Command Reference.....	377
E.1. AthenaHistory Commands.....	377
E.1.1. AH	377
E.1.2. AHexe.....	377
E.1.3. AHls.....	377
E.1.4. AHrm	377
E.2. AthenaObject Commands.....	377
E.2.1. AO	377
E.2.2. AOals.....	377
E.2.3. AOI	378
E.2.4. AOmg.....	378
E.2.5. AOrm	378
E.2.6. AOw	378
E.3. AthenaPreferences Commands	378
E.3.1. AP	378
E.3.2. APcc	378
E.3.3. APcurs.....	379
E.3.4. APdir	379

E.3.5. APdlg.....	379
E.3.6. APgfx.....	379
E.3.7. APr.....	380
E.3.8. APwid.....	380
E.4. AthenaScript Commands.....	380
E.4.1. ASexe.....	380
E.5. AthenaUtility Commands.....	380
E.5.1. AU.....	380
E.5.2. AUbeat.....	381
E.5.3. AUbug.....	381
E.5.4. AUdoc.....	381
E.5.5. AUlog.....	381
E.5.6. AUpc.....	381
E.5.7. AUSys.....	382
E.5.8. AUup.....	382
E.6. CsoundPreferences Commands.....	382
E.6.1. CP.....	382
E.6.2. CPapp.....	382
E.6.3. CPauto.....	382
E.6.4. CPch.....	383
E.6.5. CPff.....	383
E.7. EventList Commands.....	383
E.7.1. EL.....	383
E.7.2. ELh.....	383
E.7.3. ELn.....	383

E.7.4. ELr.....	383
E.7.5. ELv	384
E.7.6. ELw	384
E.8. EventMode Commands.....	384
E.8.1. EM.....	384
E.8.2. EMi.....	384
E.8.3. EMIs	384
E.8.4. EMo.....	384
E.8.5. EMv.....	385
E.9. EventOutput Commands.....	385
E.9.1. EO	385
E.9.2. EOIs.....	385
E.9.3. EOo.....	385
E.9.4. EOrm	385
E.10. MapClass Commands.....	385
E.10.1. MC.....	385
E.10.2. MCcm.....	386
E.10.3. MCgrid.....	386
E.10.4. MCnet.....	386
E.10.5. MCOpt.....	386
E.10.6. MCv.....	386
E.11. PathInstance Commands.....	387
E.11.1. PI.....	387
E.11.2. PIals	387
E.11.3. PIcp	387

E.11.4. PIdf.....	387
E.11.5. PIe.....	387
E.11.6. PIh.....	387
E.11.7. PIl.....	388
E.11.8. PImv.....	388
E.11.9. PIn.....	388
E.11.10. PIo.....	389
E.11.11. PIOpt.....	389
E.11.12. PIret.....	389
E.11.13. PIrm.....	389
E.11.14. PIrot.....	390
E.11.15. PIslc.....	390
E.11.16. PIV.....	390
E.12. PathSet Commands.....	390
E.12.1. PS.....	390
E.12.2. PScma.....	390
E.12.3. PScmb.....	390
E.13. PathVoice Commands.....	391
E.13.1. PV.....	391
E.13.2. PVan.....	391
E.13.3. PVauto.....	391
E.13.4. PVcm.....	391
E.13.5. PVcp.....	391
E.13.6. PVe.....	391
E.13.7. PVls.....	392