VPS Solver Result File

ERF-HDF5 File Specification

ESI Software

Solver / Application	:	VPS Solver (PAM-CRASH)
Solver-Version	:	2012
ERF-Version	:	1.2
Release date	:	April 2012

Contents

1 Purpose of the File

The purpose of the described output file is to store simulation results for post-processing, result mapping or chaining of simulations. This file specification is complementary to the ERF-HDF5 format specification [1]. The file format is based on the open HDF5 binary format [2].

2 Filename Convention

The file name consists of three components: ROOT_*NATURE.SUFFIX*

with:		
	ROOT	users simulation project name
	NATURE	data indicator = RESULT
	SUFFIX	file extension = erfh5

Note: Eigenmode bases are stored in a special file with NATURE=EGM. For detailed information about Eigenmode files please refer to the VPS solver manual [3].

3 Space Dimensions and Coordinates

The model is embedded in the three-dimensional Cartesian space. The deformed geometry can be represented by:

- coordinates per state (ERF variable COORDINATE) or
- coordinates and displacements (variable COORDINATE and DISPLACEMENTS_NOD)

variable key	nature	ERF array storage order	description
`COORDINATE`	vector $\vec{x}$	$[x \quad y \quad z]$	Cartesian coordinates
`DISPLACEMENTS_NOD`	vector $\vec{u}$	$[u_x \quad u_y \quad u_z]$	displacements in Cartesian coordinates

4 Solver Options

The solver and analysis type options are stored in the ERF system block under the parameter 'solver_name'. The string consists of three qualifiers separated by colons (:) as follows:

solver_name = <Platform> : <Analysis_Type> : <Application_Option>

The following table lists all possible solver and analysis types.

ERF system block parameter solver_name	Description

`'PAM-CSM:Explicit_Transient'`
`'PAM-CSM:Explicit_Transient:FPM'`
`'PAM-CSM:Explicit_Transient:Stamp'`
	explicit nonlinear transient analysis
	FPM solver results embedded
	stamping analysis

`'PAM-CSM:Static_Linear'` `'PAM-CSM:Static_Linear:Composite'`	implicit linear static analysis with multi-layered output for composites
`'PAM-CSM:Static_Nonlinear'` `'PAM-CSM:Static_Nonlinear:Composite'`	implicit non-linear static implicit analysis with multi-layered output for composites
`'PAM-CSM:Heat_Steady_Linear'`	implicit heat linear steady analysis
`'PAM-CSM:Heat_Steady_Nonlinear'`	implicit heat non-linear steady analysis
`'PAM-CSM:Heat_Transient_Linear'`	implicit heat linear transient analysis
`'PAM-CSM:Heat_Transient_Nonlinear'`	implicit heat non-linear transient analysis
`'PAM-CSM:Acoustic_Eigen_Modes'`	acoustic mode extraction
`'PAM-CSM:Eigen_Modes'`	structure mode extraction
`'PAM-CSM:Frequency_Response'`	NVH and acoustics
`'PAM-CSM:Transient_Modal'`	dynamic analysis based on modal superposition
`'PAM-CSM:Buckling_Modes'`	buckling analysis

5 Entity Types

5.1 Introduction

An ERF entity can represent a structural element (e.g. a node, a finite-element or a finite-volume cell) as well as any other sort of objects or groups of entities (e.g. the whole model, a contact interface or a multi-body system). Each entity can be referenced by its entity type key.

Notes:

1) An entity key may be any string of ASCII characters not containing a slash or a dot ("/" and ".", which are reserved as HDF path separators) and starting with a non-space character. The use of punctuation, non-printing characters and spaces should be avoided, as they may create problems for other software.
2) All identifiers (e.g. nodal, elemental, part identifiers) are one-based (greater than zero, ID > 0).

5.2 Generic Entities

Entity type	Description	Visualization	Release
NODE	FEM nodal points	point	
PART	FEM parts	n/a	
MODEL	the whole model	n/a	
INPUT	stripped PAM-CRASH input deck	n/a	
CDATA	comments (PAM-CRASH input CDATA)	n/a	
FPMNODE	FPM points	point	
FPMPART	FPM parts	n/a	
SENPT	sensor points (attached to a node)	point	
CONTACT	contact interfaces (section "Finite Elements")	contact elements	2012
SECTION	sections for force output	n/a	
ELEMENT	any type of a finite element	depending on element type	
ACOUSTIC_NODE	acoustic node with 1 DOF 'acoustic pressure' in the fluid domain	point	2011
PANEL	interface between fluid and structure domains	facets	2011
POROUS_NODE	porous node with 2 DOF 'pressure' and 'displacement'	point	2012

5.3 Finite Elements

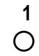
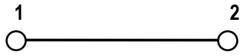
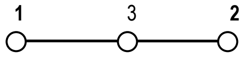
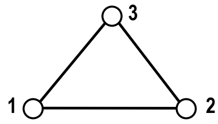
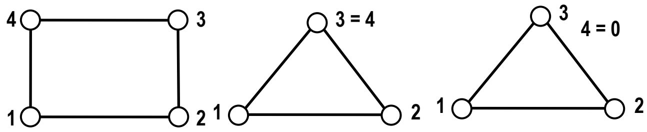
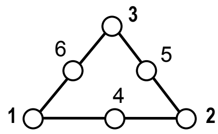
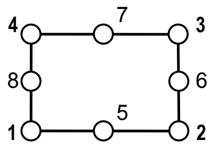
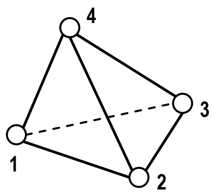
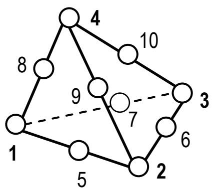
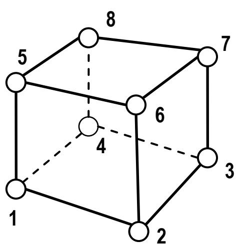
5.3.1 Overview

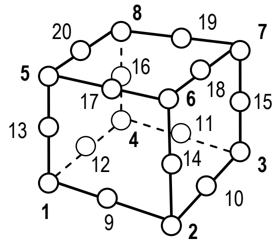
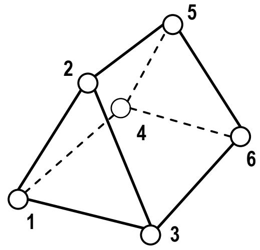
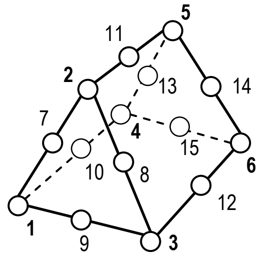
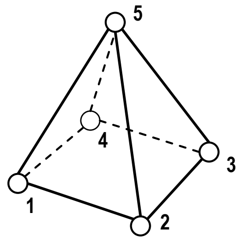
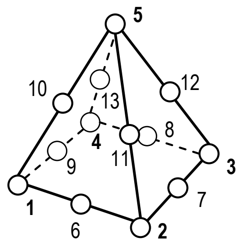
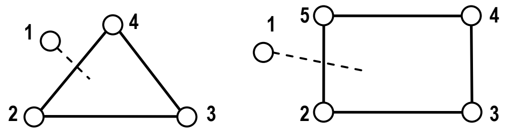
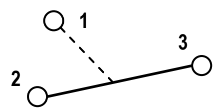
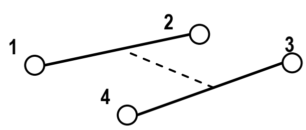
The finite elements are divided into four categories according to their dimensions in space. In addition to the classical bar and beam elements the structural link/joint elements are also considered as one-dimensional finite elements. The number of nodes and dimensions of each element type are stored as header parameter in the ERF connectivity blocks. It is highly recommended to use these parameters as primary information for visualization and extraction of faces. The purpose of the element key is to distinguish between different element formulations.

Note: The element identifiers (IDs) are unique over all types of elements. Therefore the generic type ELEMENT can also be used to refer to any type of element.

The following table shows all categories of elements used in CSM solvers. Note that an element family may consist of one or multiple element formulations, depending on the solver scheme.

Table: Types of elements

Category	Family	Nodes	Connectivity	Release
particles – 0D	point	1		
beams – 1D	line2	2		
	line3	3		
shells – 2D	tria3	3		
	quad4	4		
	tria6	6		
	quad8	8		
solids – 3D	tetra4	4		
	tetra10	10		
	hexa8	8		

	hexa20	20		
	penta6	6		
	penta15	15		
	pyramid5	5		
	pyramid13	13		
special purpose – 4D	node-to-segment	4 or 5		2012
	node-to-edge	3		2012
	edge-to-edge	4		2012

5.3.2 0D Elements

element entity type	family	description	visualization	Release
SPH	point	SPH particles	point	
FPM	point	FPM points	point	

5.3.3 1D Elements

element entity type	family	description	visualization	Release
BEAM	line2	beam C0	line	
BAR	line2	bar	line	
SPRING6DOF	line2	spring with 6 DOF	line	
SPHERICALJOINT	line2	spherical joint (mat 221)	line	
FLEXTORSJOINT	line2	flexion-torsion joint (mat 222)	line	
KJOINT	line2	kinematic joint	line	
PLINK	line2	point link	line	
MBSJOINT	line2	multi-body-system joint	line	
MBSSPRING	line2	multi-body-system spring	line	
SPRINGBEAM	line2	spring-beam	line	
DRAWBEAD	line2	stamping drawbead	line	
MTOJN	line2	multiple-to-one node kinematic joint	line	
MUSCLE	line2	muscle	line	
JET	line2	gas jet	line	
SLINK	line2	surface link	line	
MPCPLINK	line2	multi-point-constraint link	line	
GAP	line2	contact gap element (implicit only)	line	

5.3.4 2D Elements

element entity type	family	description	visualization	Release
MEMBR	quad4	membrane (mat 150, 151, 152)	facet	
THICKSHELL	quad4	thick shell (mat 161, 162)	facet-	
SHELL	quad4	4(3)-nodes shell element	facet	
SHEL6	tria6	6-nodes shell element	facet	2012
SHEL8	quad8	8-nodes shell element	facet	2012

5.3.5 3D Elements

element entity type	family	description	visualization	Release
HEXA8	hexa8	8-node hexahedral FI	facets	
BRICKSHELL	hexa8	8-node brick shell	facets	
TETRA10	tetra10	10-node tetrahedron	facets	
TETRA4	tetra4	4-node tetrahedron	facets	
PENTA6	penta6	6-node penta	facets	
PENTA15	penta15	15-node penta	facets	
HEXA20	hexa20	20-node hexahedral	facets	

5.3.6 4D Special Purpose Elements

element entity type	family	description	visualization	Release
CNT_ELE_NOD_SEG	node-to-segment	node-to-segment contact pair	point + facet	2012
CNT_ELE_NOD_EDG	node-to-edge	node-to-edge contact pair	point + line	2012
CNT_ELE_EDG_EDG	edge-to-edge	edge-to-edge contact pair	2 lines	2012

5.4 Collectors

5.4.1 Introduction

A collector is a compound of entities of different types including the collector itself. Collectors are used to visualize and access associated nodes and elements, e.g. of spotwelds, airbags or rigid body definitions. Since the collector is considered as a regular entity like an element or node, it can be used for result output (entityresults blocks).

5.4.2 Kinematic Constraints

collector entity type	description	visualization	Release
RBODY	rigid body	spider between master and slave nodes	
MTOCO_COLLECTOR	multiple-to-one kinematic constraint	spider between master and slave nodes	
OTMCO_COLLECTOR	one-to-multiple constraint	spider between slave and master nodes	

5.4.3 Airbags

collector entity type	description	visualization	Release
AIRBAG	Airbag	highlighted CHAMBER collectors	
CHAMBER	Airbag chamber	highlighted elements, VENTs, WALLs	
VENT	Airbag vent hole	highlighted elements and nodes	
WALL	Airbag chamber wall	highlighted elements	

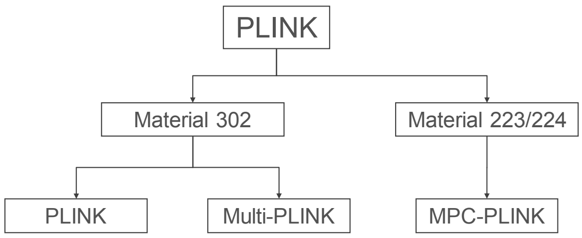
5.4.4 Muscle Collector

collector entity type	description	visualization	Release
MUSCLE_COLLECTOR	Human Model Muscle	highlighted MUSCLE elements	

5.4.5 PLINK Collectors

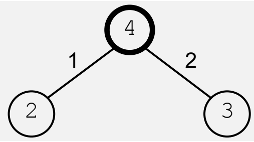
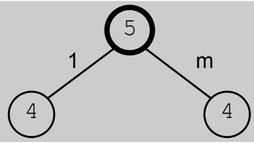
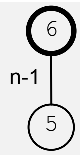
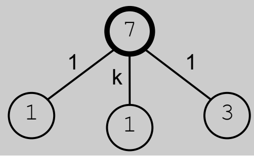
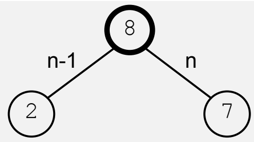
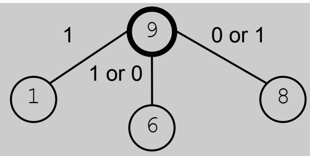
The introduction of PLINK-collectors was motivated by user requests to improve the MPC-PLINK output. The purpose of these collectors is to standardize the output of all PLINK entities and to improve their visualization in post-processors.

The family of PLINKS can be schematically presented as follows:



The following table shows all PLINK collector types and their dependencies.

Note: The collector IDs are only unique for a given collector type (i.e. not over all types).

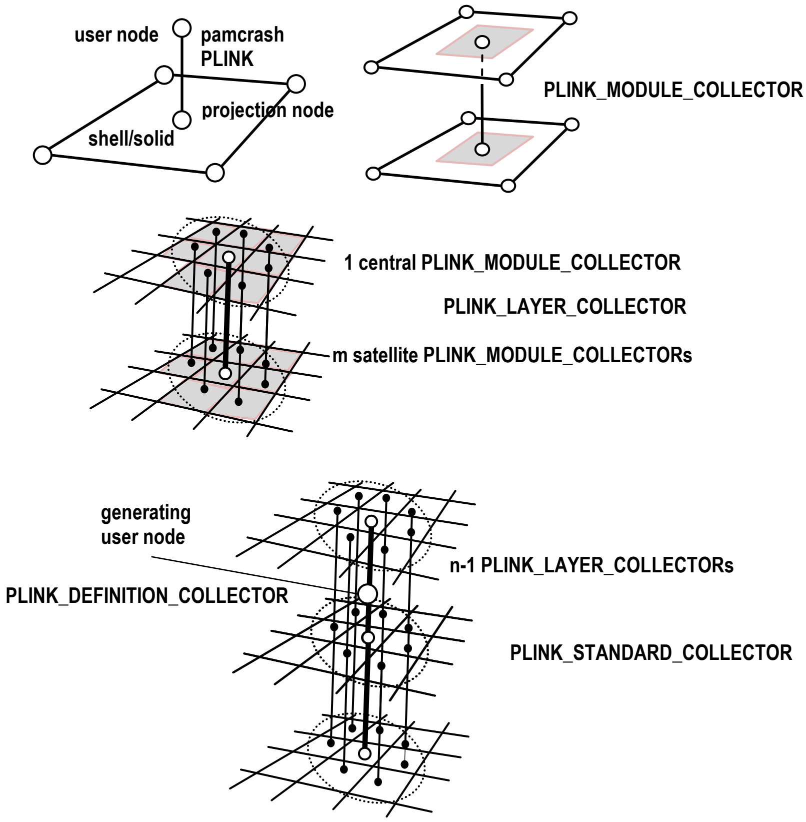
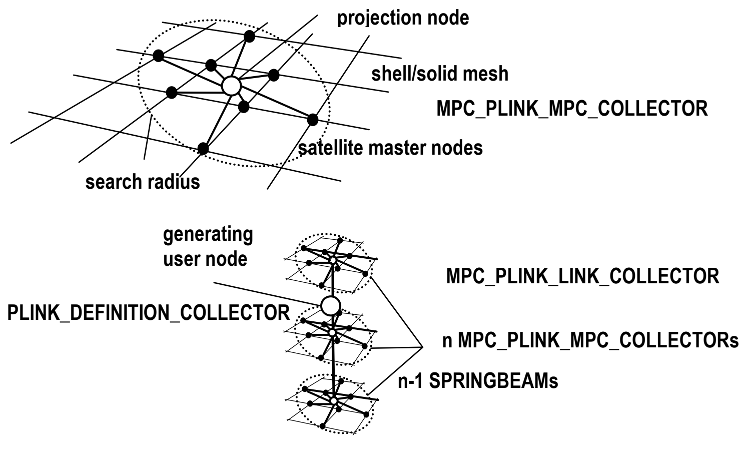
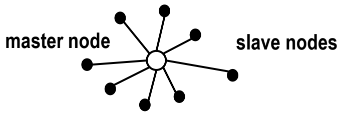
collector entity type	description	dependencies (see legend below)
PLINK_MODULE_COLLECTOR	PLINK Module Definition	
PLINK_LAYER_COLLECTOR	Layer of a Standard-PLINK Definition	
PLINK_STANDARD_COLLECTOR	Standard-PLINK Definition	
MPC_PLINK_MPC_COLLECTOR	MPC-PLINK Constraint Defintion	
MPC_PLINK_LINK_COLLECTOR	MPC-PLINK Link Definition	
PLINK_DEFINITION_COLLECTOR	MPC-/Multi-/Standard-PLINK Definition	

Legend for diagrams of collector dependencies:

symbol	entity type	definition	Identifier
1	NODE	Node	ID(1) = User NODE ID
2	ELEMENT	1D Element (beam or bar)	ID(2) = User ELEMENT ID
3	ELEMENT	2D or 3D Element (shell or solid)	ID(3) = User ELEMENT ID
4	PLINK_MODULE_COLLECTOR	1 connecting beam/bar ELEMENT 2 shell/solid ELEMENTs	ID(4) = ELEMENT ID(2)
5	PLINK_LAYER_COLLECTOR	1 central PLINK_MODULE_COLLECTOR m satellite PLINK_MODULE_COLLECTOR	ID(5) = central ID(4)
6	PLINK_STANDARD_COLLECTOR	n-1 PLINK_LAYER_COLLECTORs	ID(6) = ID(9)
7	MPC_PLINK_MPC_COLLECTOR	1 central NODE k satellite NODEs 1 ELEMENT containing central NODE	ID(7) = central NODE ID(1)
8	MPC_PLINK_LINK_COLLECTOR	n MPC_PLINK_MPC_COLLECTORs n-1 connecting beam ELEMENTs	ID(8) = ID(9)
9	PLINK_DEFINITION_COLLECTOR	1 generating NODE 0 or 1 PLINK_STANDARD_COLLECTOR 1 or 0 MPC_PLINK_LINK_COLLECTOR	ID(9) = PLINK User ID

The following table illustrates the visualization of collectors as compounds of different entity types.

Table: Visualization of collectors

PLINKs	
MPC_PLINKs	
RBODY, OTMCO_COLLECTOR	

5.5 Zones

The purpose of zones is to divide a single entity (normally an element) into multiple domains to attach results to each of these domains. A typical example is a multi-layered shell element that is used to model a composite structure.

zone type	description	visualization	Release
ZONE	default zone type – also used for single-zone entities	n/a	
LAYER	Composite applications	n/a	

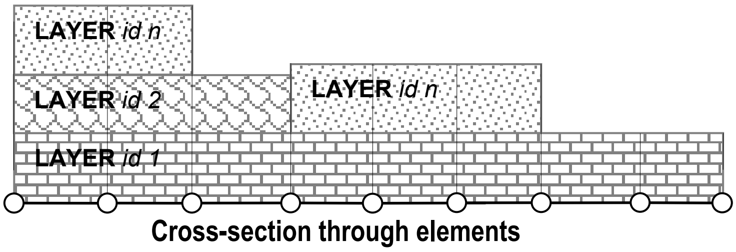


Figure: Multi-layered Shell Elements to model composite structures

Note: For detailed information about the simulation and output of composite structures please refer to the VPS solver manual [3] and the Functional Specification [4].

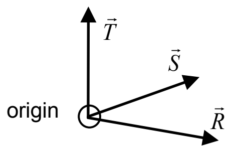
5.6 Local Frames

Vector or tensor coordinates may refer to local frames. A local frame can be referenced by the frame type key and an indentifier.

frame type	description	visualization	Release
FRAME	local frame	lines	
ELEMENT_FRAME	local frames for shell elements	lines	

The three-dimensional vector basis $\vec{R}, \vec{S}, \vec{T}$ of a local frame is stored as an entity result under the ERF variable BASIS3X3. In order to place the frame in space, the coordinates of the origin are stored under the ERF variable COORDINATE.

Note: Frames are not necessarily orthonormal or orthogonal.



variable key	nature	ERF array storage order	description
COORDINATE	vector $\vec{x}$	$[x \quad y \quad z]$	Cartesian coordinates to store the origin
BASIS3X3	matrix $\underline{B}$	$[R_x \quad R_y \quad R_z \quad S_x \quad S_y \quad S_z \quad T_x \quad T_y \quad T_z]$	vector basis in Cartesian coordinates

Figure: Local frame

6 File Storage Scheme

6.1 Introduction

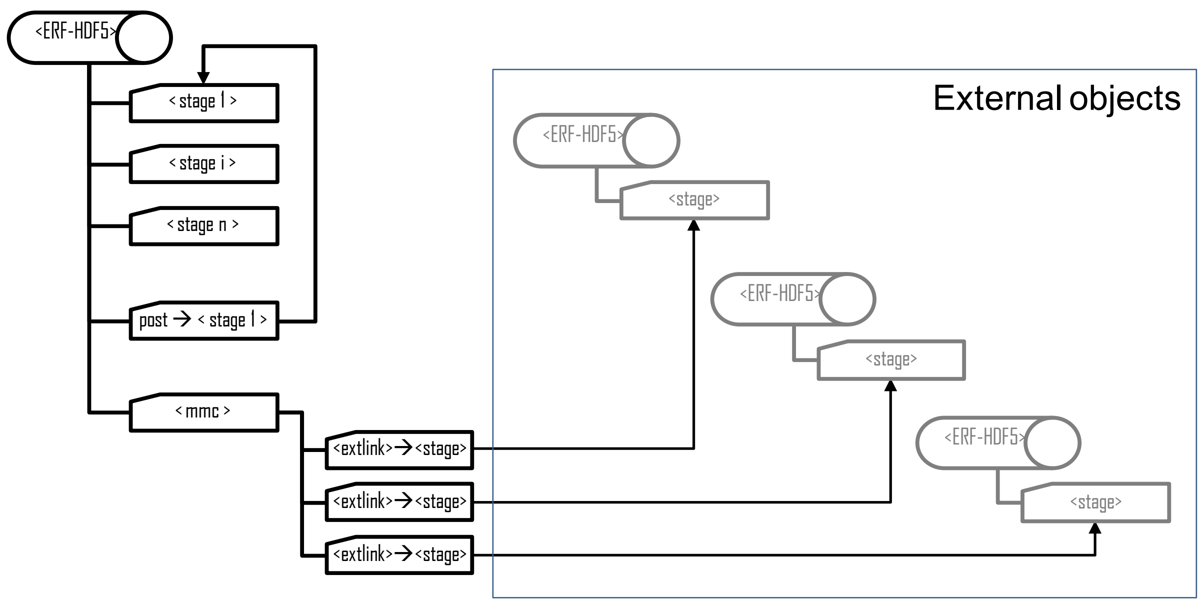
The file storage scheme defines the hierarchy of the data blocks being stored in the ERF file.

Note that the HDF5-group names – except 'post' and 'mmc', which are described below - are not part of the specification and could be changed. Therefore the reader should only rely on the block contents and not on the group/path names.

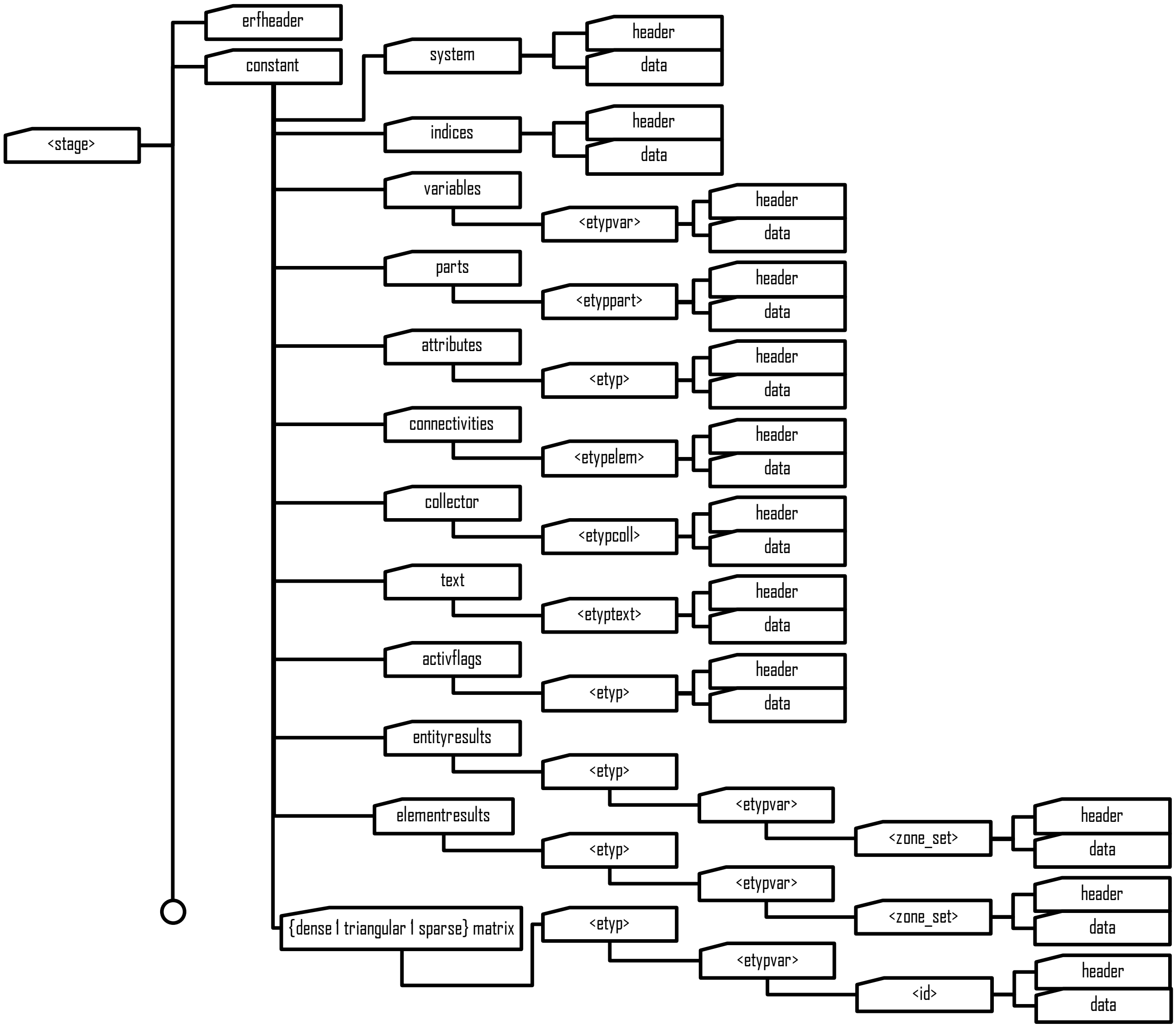
There are two optional top-level groups, 'post' and 'mmc':

- The group 'post' represents a symbolic link to a selected stage (default stage)
- The group 'mmc' (optional) contains external links to stage groups in other ERF files. These stages could be considered as sub-models of the same global model (MMC - Multi-Model Coupling). The post-processor may visualize these sub-models simultaneously (overlay mode).

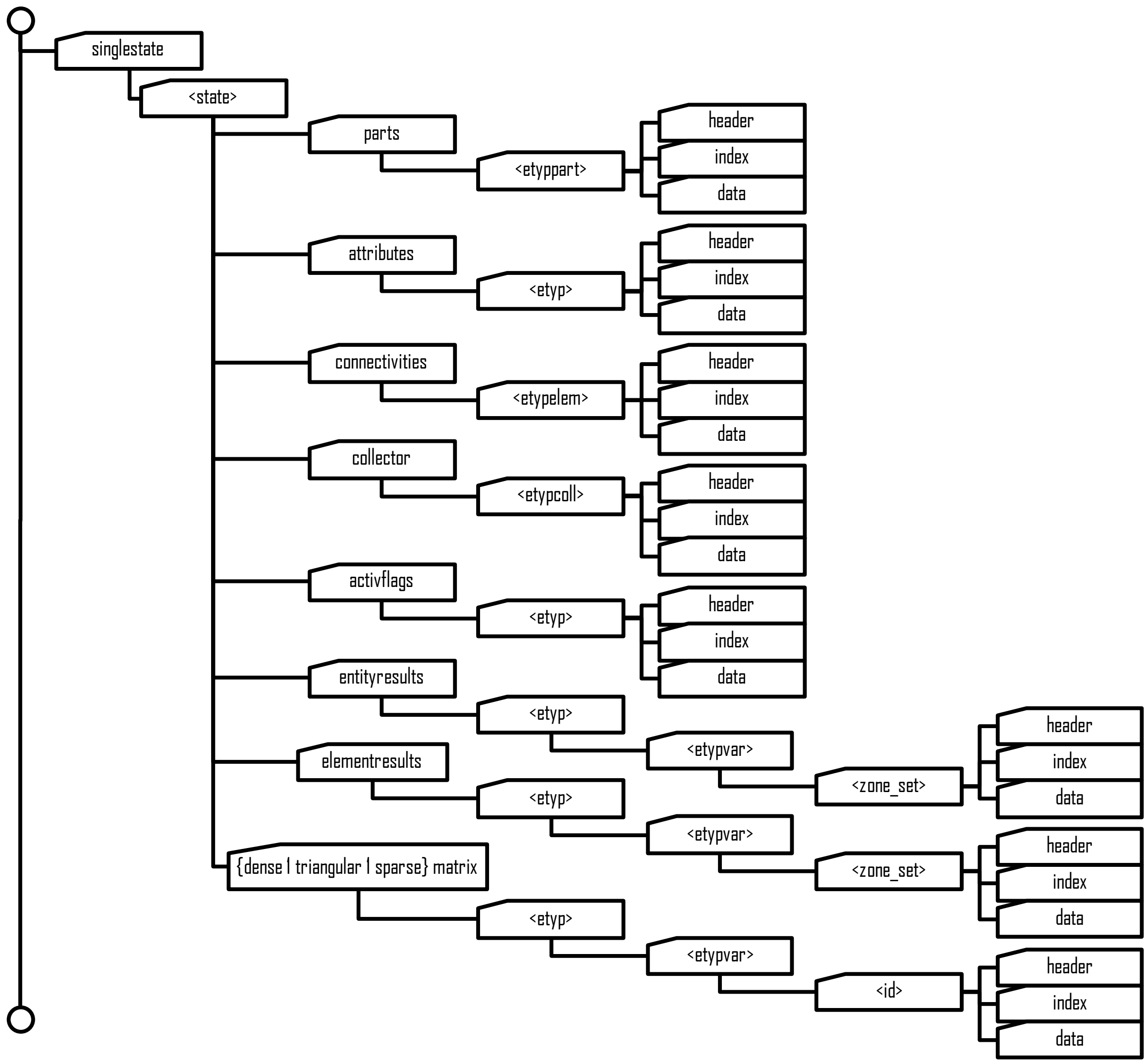
6.2 ERF File Top-Level Groups



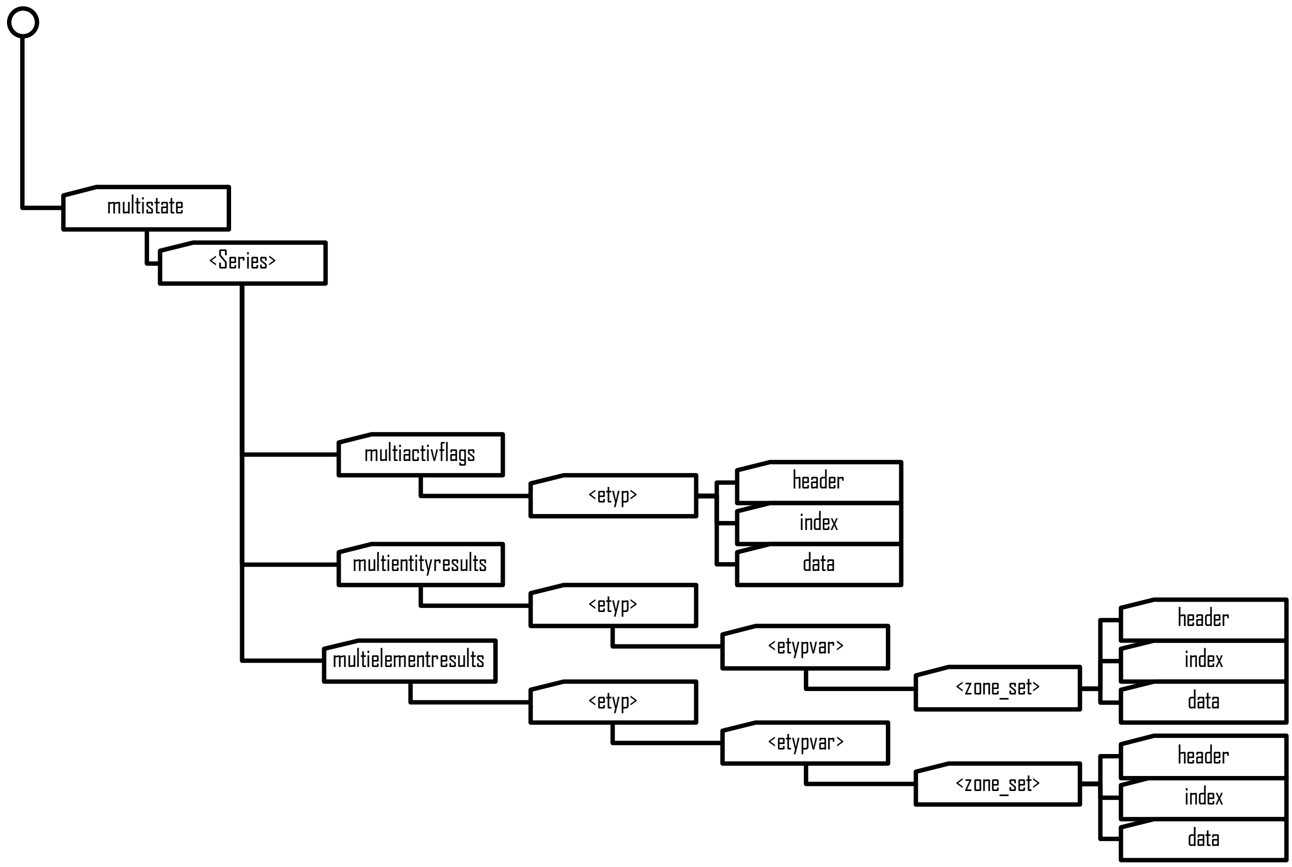
6.3 ERF-Header and Constant Blocks



6.4 Single-State Blocks



6.5 Multi-State Blocks



7 References

[1] ERF-HDF5 Specification Version 1.2, ESI Group, 2011

[2] http://www.hdfgroup.org/HDF5/, HDF Group

[3] VPS Solver Reference Manual Version 2012, ESI Group, 2012

[4] Functional specification COMP09 Visual Composite Solution Phase2, ESI Group, 2009