ISO TC 184/SC4/ WG12_ N2175

Date: 2003/07/18

Supersedes ISO TC 184/SC4/WG12_ N ___

ISO/CD 10303-110

Product data representation and exchange: Integrated application resource: Mesh-based computational fluid dynamics

ABSTRACT:

This document provides a general application independent means of representing computational fluid dynamics numerical analyses on structured and unstructured grids.

KEYWORDS: Mesh, Numerical analysis, Computational fluid dynamics

COMMENTS TO READER:

Recipients of this draft are invited to submit, with their comments, notification of any relevant patent rights of which they are aware and to provide supporting documentation. The formal modeling uses EXPRESS edition 2. This document has been reviewed using the internal review checklist (see WG12 N2177), the project leader checklist (see WG12 N2178), and the convenor checklist (see WG12 N2179).

Project Leader: Ray Cosner
Address: Boeing, Phantom Works
PO Box 516,
M/S S106–7126
St. Louis, MO 63166

Telephone: +1 (314) 233–6481
Facsimile: +1 (314) 777–1328
E-mail: raymond.r.cosner@boeing.com

Project Editor: Peter Wilson
Address: Boeing Commercial Airplane
PO Box 3707, M/S 2R–97
Seattle, WA 98124–2207

Telephone: +1 (206) 544–0589
Facsimile: +1 (206) 544–5889
E-mail: peter.r.wilson@boeing.com

ISO copyright office
Case postale 56. CH-1211 Geneva 20
Tel. +41 22 749 01 11
Fax +41 22 734 10 79
E-mail `copyright@iso.ch`
Web `www.iso.ch`

Contents

Page

Figures

Tables

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of technical committees is to prepare International Standards. Draft International Standards adopted by the technical committees are circulated to the member bodies for voting. Publication as an International Standard requires approval by at least 75% of the member bodies casting a vote.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO shall not be held responsible for identifying any or all such patent rights.

ISO 10303–110 was prepared by Technical Committee ISO/TC 184, *Industrial automation systems and integration*, Subcommittee SC4, *Industrial data*.

This International Standard is organized as a series of parts, each published separately. The structure of this International Standard is described in ISO 10303–1.

Each part of this International Standard is a member of one of the following series: description methods, implementation methods, conformance testing methodology and framework, integrated generic resources, integrated application resources, application protocols, abstract test suites, application interpreted constructs, and application modules. This part is a member of the integrated application resource series. The integrated generic resources and the integrated application resources specify a single conceptual product data model.

A complete list of parts of ISO 10303 is available from the Internet:
`<http://www.nist.gov/sc4/editing/step/titles/>`

Annexes A and B are a normative part of this part of ISO 10303. Annexes C, D, E and F are for information only.

Introduction

ISO 10303 is an International Standard for the computer-interpretable representation of product information and for the exchange of product data. The objective is to provide a neutral mechanism capable of describing products throughout their life cycle. This mechanism is suitable not only for neutral file exchange, but also as a basis for implementing and sharing product databases, and as a basis for archiving.

This part of ISO 10303 is a member of the integrated resources series. Major subdivisions of this part of ISO 10303 are:

— **basis_schema**;

— **hierarchy_schema**;

— **domain_schema**;

— **conditions_schema**;

— **equations_schema**;

— **results_schema**.

The relationships of the schemas in this part of ISO 10303 to other schemas that define the integrated resources of this International Standard are illustrated in Figure 1 using the EXPRESS-G notation. EXPRESS-G is defined in annex D of ISO 10303-11. The schemas identified in the bold boxes are specified in this part of ISO 10303. The **external_reference_schema**, **product_property_definition_schema**, **product_property_representation_schema**, and the **support_resource_schema** schemas are specified in part 41 of ISO 10303. The **representation_schema** schema is specified in part 43 of ISO 10303. The **mathematical_context_schema** and the **mathematical_description_of_distribution_schema** schemas are specified in part 51 of ISO 10303. The **mesh_connectivity_schema** and the **mesh_topology_schema** schemas are specified in part 52 of ISO 10303. The **analysis_schema** schema is specified in part 53 of ISO 10303. The schemas illustrated in Figure 1 are components of the integrated resources.

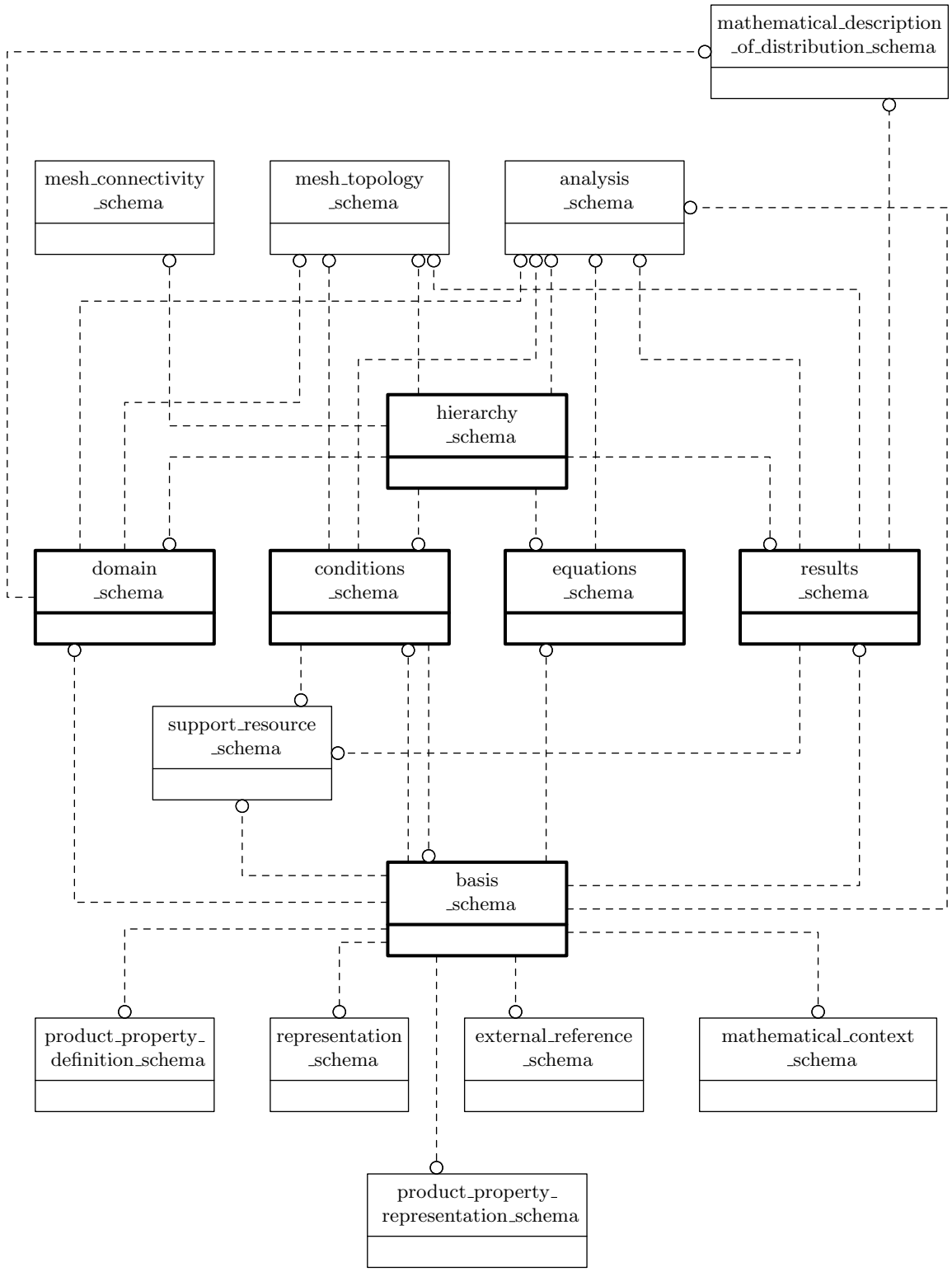

Figure 1 – Schema relationships

Industrial automation systems and integration — Product data representation and exchange — Part 110 : Integrated application resource: Mesh-based computational fluid dynamics

1 Scope

The following are within the scope of this part of ISO 10303:

— digital data on structured and unstructured grids describing steady or unsteady fluid dynamics flowfields;

— data describing the fluid dynamics model including grid description, grid inter-connectivity, boundary conditions, and modeling parameters;

— data from solutions of equation sets commonly used in fluid dynamics analysis: Navier-Stokes equations, Euler equations, linear and nonlinear potential flow equations, small-disturbance equations, boundary layer equations, and stream function equations;

— data at any point in the analysis activity;

— single-phase flow of a liquid or a gas;

— laminar flow, transitional flow, turbulent flow (direct representation of turbulence, or represented by Reynolds-averaged data);

— incompressible or compressible flow;

— unsteady flow;

— perfect gas, or variable chemical composition (equilibrium flow, frozen flow, or finite-rate chemical reactions);

— data regarding the exchange of energy by molecular transport including convection, conduction, and advectation;

— rotating flowfields (e.g., turbomachinery);

— inertial and rotating frames of reference;

— Newtonian transport laws;

— reference to product geometry;

— administrative information necessary to track the approval and configuration control of the analysis of a product;

The following are outside the scope of this part of ISO 10303:

— representations of geometry;

— gross flow in networks (e.g., piping and ducting);

— the use that application programs may make of the data;

— the means by which application programs modify the data;

— the form in which the data is stored internal to an application.

The validity, accuracy and completeness of the data for a particular purpose are determined entirely by the applications' software.

NOTE 1 The following are outside the scope of this edition of this part of ISO 10303 but are expected to be inside the scopes of later editions of this part:

— two- and three-phase flow;

— free surface flow;

— non-continuum flow (e.g., direct simulation of Monte Carlo data);

— data from non-analytical sources (e.g., experimental simulation such as wind tunnel or water tank testing, and product test such as flight test or sea trials);

— data regarding the exchange of energy by radiation;

— non-Newtonian transport laws;

— electro-magnetic interactions with a fluid;

— plasmas.

2 Normative references

The following normative documents contain provisions which, through reference in this text, constitute provisions of this international standard. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply. However, parties to agreements based on this international standard are encouraged to investigate the possibility of applying the most recent editions of the normative documents indicated below. For undated references, the latest edition of the normative document referred to applies. Members of ISO and IEC maintain registers of currently valid International Standards.

ISO 10303-1:1994, *Industrial automation systems and integration — Product data representation and exchange — Part 1: Overview and fundamental principles.*

ISO 10303-11:2003, *Industrial automation systems and integration — Product data representation and exchange — Part 11: Description method: The EXPRESS language reference manual.*

ISO 10303-41:2000, *Industrial automation systems and integration — Product data representation and exchange — Part 41: Integrated generic resource: Fundamentals of product description and support.*

ISO 10303-43:2000, *Industrial automation systems and integration — Product data representation and exchange — Part 43: Integrated generic resource: Representation structures.*

ISO 10303-51:—[1)], *Industrial automation systems and integration — Product data representation and exchange — Part 51: Integrated generic resource: Mathematical description.*

ISO 10303-52:—[1)], *Industrial automation systems and integration — Product data representation and exchange — Part 52: Integrated generic resource: Mesh-based topology.*

ISO 10303-53:—[1)], *Industrial automation systems and integration — Product data representation and exchange — Part 53: Integrated generic resource: Numerical analysis.*

ISO/IEC 8824-1:1998, *Information technology — Abstract Syntax Notation One (ASN.1): Specification of basic notation.*

3 Terms, definitions, abbreviations, and symbols

3.1 Terms defined in ISO 10303-1

— application protocol (AP)

— integrated resource (IR)

3.2 Other terms and definitions

3.2.1
BC patch
the subrange of a face of a zone where a given boundary-condition is applied

3.2.2

[1)] To be published.

Table 1 – Symbols for dimensional units

Symbol	Description
M	mass unit
L	length unit
T	time unit
Θ	temperature unit
α	angle unit

Table 2 – Symbols for coordinate systems

Symbol	Description
x, y, z	coordinates in a Cartesian system
r, θ, z	coordinates in a Cylindrical system
r, θ, ϕ	coordinates in a Spherical system
ξ, η, ζ	coordinates in an auxiliary system

computational fluid dynamics
the set of knowledge and tools used to generate exact or approximate solutions to the mathematical equations governing the motion of a fluid (gas or liquid).

NOTE 1 The underlying knowledge is implemented in computing codes or application programs.

3.2.3
global BC data
boundary-condition data applied globally to a BC patch; for example, specifying a uniform total pressure.

3.2.4
local BC data
boundary-condition data applied at each grid point of a BC patch; an example of this is varying total pressure specified at each vertex of a BC patch.

3.3 Abbreviations

CFD computational fluid dynamics

3.4 Symbols

Symbols for dimensional units are given in Table 1.

EXAMPLE 1 A length has dimensions $\mathbf{L}$, an area has dimensions $\mathbf{L}^2$, and a velocity has dimensions $\mathbf{L}/\mathbf{T}$ (alternatively written as $\mathbf{L}\mathbf{T}^{-1}$).

Symbols for coordinate systems are given in Table 2.

Associated with the coordinate systems are unit vectors, the symbols for which are given in Table 3.

Symbols for physical properties are given in Table 4.

Symbols for nondimensional parameters are given in Table 5

Table 3 – Symbols for unit vectors

Symbol	Direction	Symbol	Direction	Symbol	Direction
$\hat{e}_x$	x-direction	$\hat{e}_r$	r-direction	$\hat{e}_\xi$	ξ-direction
$\hat{e}_y$	y-direction	$\hat{e}_\theta$	θ-direction	$\hat{e}_\eta$	η-direction
$\hat{e}_z$	z-direction	$\hat{e}_\phi$	ϕ-direction	$\hat{e}_\zeta$	ζ-direction

Table 4 – Symbols for physical properties

Symbol	Description
ρ	static density
p	static pressure
T	static temperature
e	static internal energy per unit mass
h	static enthalpy per unit mass
s	entropy
ρ_0	stagnation density
p_0	stagnation pressure
T_0	stagnation temperature
e_0	stagnation energy per unit mass
h_0	stagnation enthalpy per unit mass
ρe_0	stagnation energy per unit volume
ν	kinematic viscosity ($\nu = \mu/\rho$)
μ	molecular viscosity
ν_t	kinematic eddy viscosity
μ_t	eddy viscosity
k	thermal conductivity coefficient
R	ideal gas constant ($R = c_p - c_v$)
c_p	specific heat at constant pressure
c_v	specific heat at constant volume

Table 5 – Symbols for nondimensional parameters and related scales

Symbol	Description
M	Mach number ($M = q/c$)
q	Mach velocity scale
c	Mach speed of sound scale
R_e	Reynolds number ($R_e = VL/\nu$)
V	Reynolds velocity scale
L	Reynolds length scale
ν	Reynolds kinematic viscosity scale
Pr	Prandtl number ($Pr = \mu c_p/k$)
k	Prandtl thermal conductivity scale
μ	Prandtl molecular viscosity scale
c_p	Prandtl specific heat scale
γ	specific heat ratio ($\gamma = c_p/c_v$)
c_p	specific heat at constant pressure
c_v	specific heat at constant volume

4 Basis

The following EXPRESS declaration begins the **basis_schema** and identifies the necessary external references.

EXPRESS specification:

```
*)
SCHEMA basis_schema;
  REFERENCE FROM analysis_schema                          -- ISO 10303-53
                (model_property_distribution);
  REFERENCE FROM conditions_schema                        -- ISO 10303-110
                (Riemann_1D_data_name);
  REFERENCE FROM domain_schema                            -- ISO 10303-110
                (coordinate_data_name);
  REFERENCE FROM equations_schema                         -- ISO 10303-110
                (force_moment_data_name,
                 gas_model_data_name,
                 thermal_conductivity_model_data_name,
                 turbulence_closure_data_name,
                 turbulence_model_data_name,
                 viscosity_model_data_name);
  REFERENCE FROM external_reference_schema                -- ISO 10303-41
                (externally_defined_item);
  REFERENCE FROM mathematical_context_schema              -- ISO 10303-51
                (value_context_select);
  REFERENCE FROM mathematical_description_of_distribution_schema -- ISO 10303-51
                (property_distribution_select);
  REFERENCE FROM product_property_definition_schema       -- ISO 10303-41
                (general_property);
  REFERENCE FROM product_property_representation_schema -- ISO 10303-41
                (shape_representation);
  REFERENCE FROM representation_schema                    -- ISO 10303-43
                (representation_context);
  REFERENCE FROM results_schema                           -- ISO 10303-110
                (flow_solution_data_name);
  REFERENCE FROM support_resource_schema                  -- ISO 10303-41
                (text);
(*
```

NOTE The schemas referenced above can be found in the following parts of ISO 10303:

analysis_schema	part 53
conditions_schema	this part of ISO 10303
domain_schema	this part of ISO 10303
equations_schema	this part of ISO 10303
external_reference_schema	part 41
mathematic_context_schema	part 51
mathematical_description_of_distribution_schema	part 51
product_property_definition_schema	part 41
product_property_representation_schema	part 41
representation_schema	part 43
results_schema	this part of ISO 10303
support_resource_schema	part 41

Abbreviated names are used in identifiers of elements declared in this schema. Prefixes used in these identifiers have the following meanings:

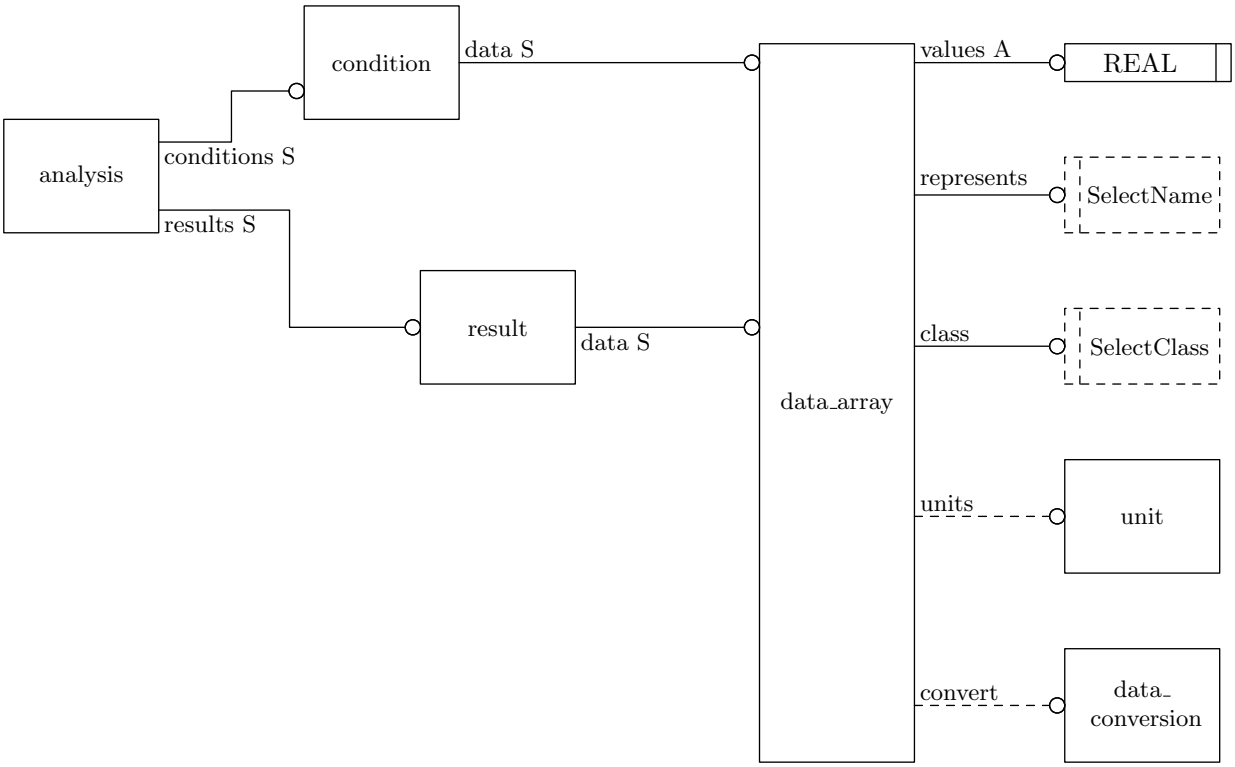

Figure 2 – EXPRESS-G **partial model sketching a skeleton for an analysis information model**

fd fluid_dynamics;

mbna mesh_based_numerical_analysis.

4.1 Introduction

This schema defines and describes low-level structures and types that are used in the definition of more complex structures in the hierarchy.

4.2 Fundamental concepts and assumptions

One practical aspect of some kinds of numerical analyses is that they involve massive amounts of numerical data. For example in CFD analyses the data includes coordinate values of meshes and the calculated results at each cell or node in the mesh; the numerical data may be Gigabytes in size.

Conceptually an information model for numerical analyses of the type considered here is simple.

NOTE 1 A sketch of the skeleton of such a model is shown in Figure 2.

An **analysis** has elements like boundary **condition**s and calculated **result**s. In turn, a **result** or a **condition** has numerical data which is contained in a **data_array**. A **data_array** has an array of data values and information about the class of data, the units of measure, data conversion information, and what the values represent.

Conceptually the structure type **data_array** is a general purpose structure that may be used

for holding data arrays and scalars. In the context of mesh-based numerical analysis it is used to describe grid coordinates, solution data, governing parameters, boundary-condition data, and other information. Information such as the number of dimensions in a multidimensional arry, the array sizes for each dimension, the dimensional units (if any), may be maintained by the **data_array**.

Several classes of data need to be addressed with the **data_array** structure type, for example:

a) dimensional data (e.g., velocity in units of m/s);

b) nondimensional data normalized by dimensional reference quantities;

c) nondimensional data with associated nondimensional reference quantities;

d) nondimensional parameters (e.g., Reynolds number, pressure coefficient);

e) pure constants (e.g., π, e).

Each of the classes of data requires different information to describe dimensional units or normalization associated with the data.

Identifiers or data-names need to be associated with instances of **data_array** entities to identify and describe the quantity being stored; as examples, one instance may be holding data related to coordinate values while another holds data related to density values. To facilitate communication between different application codes, an initial set of standardized data-name identifiers with fairly precise definitions are provided. For any identifier in this set, the associated data should be unambiguously understood. In essence, this schema supplies standardized terminology for labeling analysis-related data such as grid coordinates.

All standardized identifiers denote scalar quantities; this is consistent with the intended use of the **data_array** structure type to describe an array of scalars. For quantities that are vectors, such as velocity, their components are listed.

In the conceptual **data_array** the *data name* for the data is specified by the associated **SelectName** and the *data class* of the data is specified by the associated **SelectClass**.

The other attributes of **data_array** provide information for manipulating the data, including changing units or normalization. Within a given instance of **data_array** where the data class is specified by a **defined_representation_context_identifier**, all information required for manipulations may be completely and precisely specified by the data class and the values of **units**, **exponents** and **conversion**.

— **dimensional:** When the data class is **dimensional**, the data is dimensional. The optional qualifier **units** describe the dimensional units associated with the data.

  EXAMPLE 1 If the data is the x-component of velocity, then **units** will state that the pertinent dimensional units are, say, **metre** and **second**.

— **normalized_by_dimensional:** When the data class is **normalized_by_dimensional**, the data is nondimensional and is normalized by dimensional reference quantities. All optional entities in **data_array** are used. **conversion** contains factors to convert the nondimensional data to 'raw' dimensional data; these factors are **scale** and **offset**. The conversion process is as follows:


```
Data(raw) = Data(nondimensional)*scale + offset
```

where `Data(nondimensional)` is the original nondimensional data, and `Data(raw)` is the converted raw data. This converted raw data is dimensional, and the qualifier **units** describes the appropriate dimensional units. The **units** also describes the units for **scale** and **offset**.

— **normalized_by_unknown_dimensional:** When the data class is **normalized_by_unknown_dimensional**, the data is nondimensional and is normalized by some unspecified dimensional quantities. This type of data is typical of a completely nondimensional test case, where all field data and all reference quantities are nondimensional.

Rather than providing qualifiers to describe the normalization of the data, all data of type **normalized_by_unknown_dimensional** in a given object base shall be nondimensionalized consistently. This is done by picking one set of mass, length, time and temperature scales and normalizing all appropriate data by these scales. This process is described in detail in the following.

NOTE 2 The practice of nondimensionalization within flow solvers and other application codes is quite popular. The problem with this practice is that to manipulate the data from a given code, one must often know the particulars of the nondimensionalization used. This largely results from what can be termed inconsistent normalization — more than the minimum required scales are used to normalize data within the code.

EXAMPLE 2 In one CFD flow solver, the following nondimensionalization is used:

$$\begin{array}{lll} \tilde{x} = x/L, & \tilde{u} = u/c_\infty, & \tilde{\rho} = \rho/\rho_\infty, \\ \tilde{y} = y/L, & \tilde{v} = v/c_\infty, & \tilde{p} = p/(\rho_\infty c_\infty^2), \\ \tilde{z} = z/L, & \tilde{w} = w/c_\infty, & \tilde{\mu} = \mu/\mu_\infty, \end{array}$$

where (x, y, z) are the cartesian coordinates, (u, v, w) are the cartesian components of velocity, ρ is static density, p is static pressure, c is the static speed of sound, and μ is the molecular viscosity. In this example, tilde quantities ($\tilde{}$) are nondimensional and all others are dimensional. Four dimensional scales are used for normalization: L (a unit length), ρ_∞, c_∞ and μ_∞. However, only three fundamental dimensional units are represented: mass, length and time. The extra normalizing scale leads to inconsistent normalization. The primary consequence of this is additional nondimensional parameters, such as Reynolds number, appearing in the nondimensionalized governing equations where none are found in the original dimensional equations. Many definitions, including skin friction coefficient, also have extra terms appearing in the nondimensionalized form. This adds unnecessary complication to any data or equation manipulation associated with the flow solver.

Consistent normalization avoids many of these problems. Here the number of scales used for normalization is the same as the number fundamental dimensional units represented by the data. Using consistent normalization, the resulting nondimensionalized form of equations and definitions is identical to their original dimensional formulations. One piece of evidence to support this assertion is that it is not possible to form any nondimensional parameters from the set of dimensional scales used for normalization.

An important fallout of consistent normalization is that the actual scales used for normalization become immaterial for all data manipulation processes. To illustrate this consider the following nondimensionalization procedure: Let M (mass), L (length) and T (time) be arbitrary dimensional scales by which all data is normalized (neglect temperature data for the present). The nondimensional data follows:

$$\begin{array}{lll} x' = x/L, & u' = u/(L/T), & \rho' = \rho/(M/L^3), \\ y' = y/L, & v' = v/(L/T), & p' = p/(M/(LT^2)), \\ z' = z/L, & w' = w/(L/T), & \mu' = \mu/(M/(LT)), \end{array}$$

where primed quantities are nondimensional and all others are dimensional.

Consider an existing object base where all field data and all reference data is nondimensional and normalized as shown. Assume the object base has a single reference state given by,

$$\begin{array}{lll} x'_{\text{ref}} = x_{\text{ref}}/L, & u'_{\text{ref}} = u_{\text{ref}}/(L/T), & \rho'_{\text{ref}} = \rho_{\text{ref}}/(M/L^3), \\ y'_{\text{ref}} = y_{\text{ref}}/L, & v'_{\text{ref}} = v_{\text{ref}}/(L/T), & p'_{\text{ref}} = p_{\text{ref}}/(M/(LT^2)) \\ z'_{\text{ref}} = z_{\text{ref}}/L, & w'_{\text{ref}} = w_{\text{ref}}/(L/T), & \mu'_{\text{ref}} = \mu_{\text{ref}}/(M/(LT)). \end{array}$$

If a user wanted to change the nondimensionalization of grid-point pressures, the procedure is straightforward. Let the desired new normalization be given by $p''_{ijk} = p_{ijk}/(\rho_{\text{ref}} c^2_{\text{ref}})$, where all terms on the right-hand-side are *dimensional*, and as such they are unknown to the object base user. However, the desired manipulation is possible using only nondimensional data provided in the object base,

$$\begin{aligned} p''_{ijk} &\equiv p_{ijk}/(\rho_{\text{ref}} c^2_{\text{ref}}) \\ &= \frac{p_{ijk}}{M/(LT^2)} \frac{M/L^3}{\rho_{\text{ref}}} \left[\frac{L/T}{c_{\text{ref}}}\right]^2 \\ &= p'_{ijk}/(\rho'_{\text{ref}} (c'_{\text{ref}})^2) \end{aligned}$$

Thus, the desired renormalization is possible using the object base's nondimensional data as if it were actually dimensional. There is, in fact, a high degree of equivalence between dimensional data and consistently normalized nondimensional data. The procedure shown in this example should extend to any desired renormalization, provided the needed reference-state quantities are included in the object base.

This example points out two requirements for data in the class **normalized_by_unknown_dimensional**,

a) All nondimensional data within a given object base that is of class **normalized_by_unknown_dimensional** shall be consistently normalized.

b) Any nondimensional reference state appearing in an object base should be sufficiently populated with reference quantities to allow for renormalization procedures.

These two stipulations lead to the following:

— The dimensional scales used to nondimensionalize all data are immaterial, and there is no need to identify these quantities in an object base.

— The dimensional scales need not be reference-state quantities provided in the object base.

EXAMPLE 3 A CFD analysis could contain freestream reference state conditions, but all the data is normalized by sonic conditions (which are not provided).

— All renormalization procedures can be carried out treating the data as if it were dimensional with a consistent set of units.

— Any application code that internally uses consistent normalization can use the data provided in an object base without modification or transformation to the code's internal normalization.

EXAMPLE 4 A CFD application code that internally uses inconsistent normalization could easily read and write data to a nondimensional object base that conforms to the above stipulations. On output, the code could renormalize data so it is consistently normalized. Probably, the easiest method would be to remove the molecular viscosity scale (μ_∞), and only use L, ρ_∞ and c_∞ for all normalizations (recall these are dimensional scales). The only change from the above example would

be the nondimensionalization of viscosity, which would become, $\tilde{\tilde{\mu}} = \mu/(\rho_\infty c_\infty L)$. The code could then output all field data as,

$$\begin{array}{lll} \tilde{x}_{ijk} = x_{ijk}/L, & \tilde{u}_{ijk} = u_{ijk}/c_\infty, & \tilde{\rho}_{ijk} = \rho_{ijk}/\rho_\infty, \\ \tilde{y}_{ijk} = y_{ijk}/L, & \tilde{v}_{ijk} = v_{ijk}/c_\infty, & \tilde{p}_{ijk} = p_{ijk}/(\rho_\infty c_\infty^2), \\ \tilde{z}_{ijk} = z_{ijk}/L, & \tilde{w}_{ijk} = w_{ijk}/c_\infty, & \tilde{\tilde{\mu}}_{ijk} = \mu_{ijk}/(\rho_\infty c_\infty L), \end{array}$$

and output the freestream reference quantities,

$$\begin{array}{ll} \tilde{u}_\infty = u_\infty/c_\infty, & \tilde{\rho}_\infty = \rho_\infty/\rho_\infty = 1, \\ \tilde{v}_\infty = v_\infty/c_\infty, & \tilde{p}_\infty = p_\infty/(\rho_\infty c_\infty^2) = 1/\gamma, \\ \tilde{w}_\infty = w_\infty/c_\infty, & \tilde{\tilde{\mu}}_\infty = \mu_\infty/(\rho_\infty c_\infty L) \sim O(1/Re), \end{array}$$

where γ is the specific heat ratio (assumes a perfect gas) and Re is the Reynolds number.

On input, the flow solver should be able to recover its internal normalizations from the data in a nondimensional object base by treating the data as if it were dimensional.

— **nondimensional_parameter:** When the data class is **nondimensional_parameter**, the data is a nondimensional parameter (or array of nondimensional parameters). Examples include Mach number, Reynolds number and pressure coefficient. These parameters are prevalent in CFD, although their definitions tend to vary between different application codes.

Nondimensional parameters are distinguished from other data classes by the fact that they are *always* dimensionless. In a completely nondimensional object base, they are distinct in that their normalization is not necessarily consistent with other data.

Typically, the **units** and **conversion** qualifiers are not used for nondimensional parameters; although, there are a few situations where they may be used (these are discussed below). Rather than rely on optional qualifiers to describe the normalization, any nondimensional parameter shall be accompanied by their defining scales;

EXAMPLE 5 An example is Reynolds number defined as $Re = VL/\nu$, where V, L and ν are velocity, length and viscosity scales, respectively. Note that these defining scales may be dimensional or nondimensional data.

In certain situations, it may be more convenient to use the optional qualifiers of **data_array** to describe the normalization used in nondimensional parameters. These situations must satisfy two requirements: First, the defining scales are dimensional; and second, the nondimensional parameter is a normalization of a single 'raw' data quantity and it is clear what this raw data is.

EXAMPLE 6 Examples that satisfy this second constraint are pressure coefficient, where the raw data is static pressure, and lift coefficient, where the raw data is the lift force. Conversely, Reynolds number is a parameter that violates the second requirement — there are three pieces of raw data rather than one that make up Re.

For nondimensional parameters that satisfy these two requirements, the qualifiers **units** and **conversion** may be used to recover the raw dimensional data.

— **dimensionless_constant:** When the data class is **dimensionless_constant**, the data is a constant (or array of constants) with no associated dimensional units. The **units** and **conversion** qualifiers are not used.

NOTE 3 The number of dimensions in a multidimensional array, the array sizes for each dimension, and

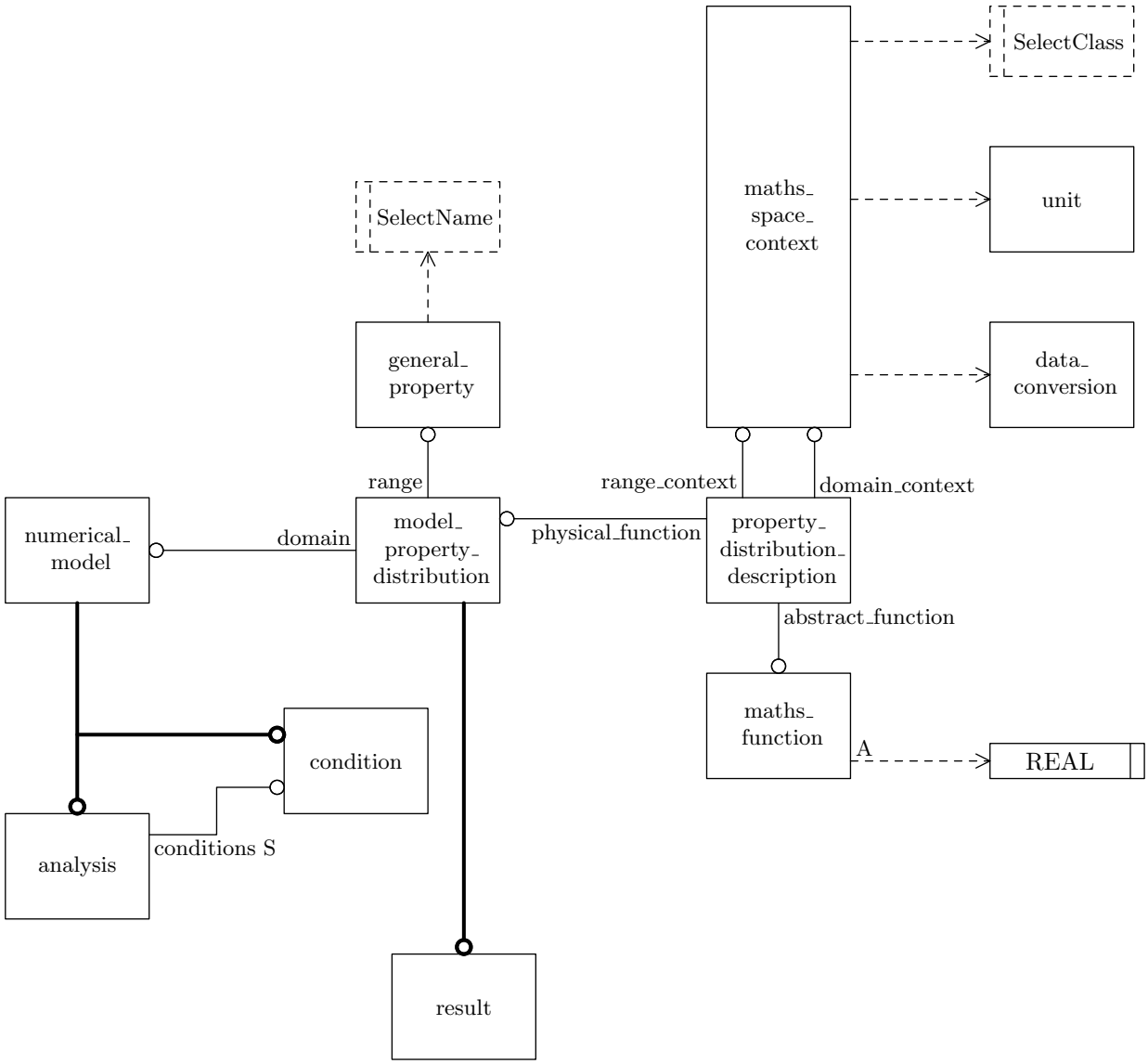

Figure 3 – EXPRESS-G **partial model sketching the realisation of the conceptual data_array**

the dimensional units (if any) of the data is maintained as parts of, or associations with, the **property_distribution_description**. Details are given in ISO 10303 part 51, and in particular, in part 50.

NOTE 4 The basis for the normalisation is specified by a **maths_space_context** which references a **value_context_select** as its normalisation basis. The **maths_space_context** is applied to the **maths_space** which is referenced by the **property_distribution_function** (which is a SUPERTYPE of **data_array**) whose attribute **property_distribution_description.abstract_function** is a **maths_function** that provides the data representation for the **data_array**.

The realisation of the conceptual **data_array** is more opaque than the above description.

NOTE 5 A sketch illustrating the realisation of the conceptual **data_array** is shown in Figure 3.

The **analysis** entity is a particular kind (subtype) of **numerical_analysis**, as is the **condition** entity. The **result** entity is a kind (subtype) of **model_property_distribution**. Instead of an **analysis** directly specifying the results, the **model_property_distribution** specifies the **numerical_analysis**.

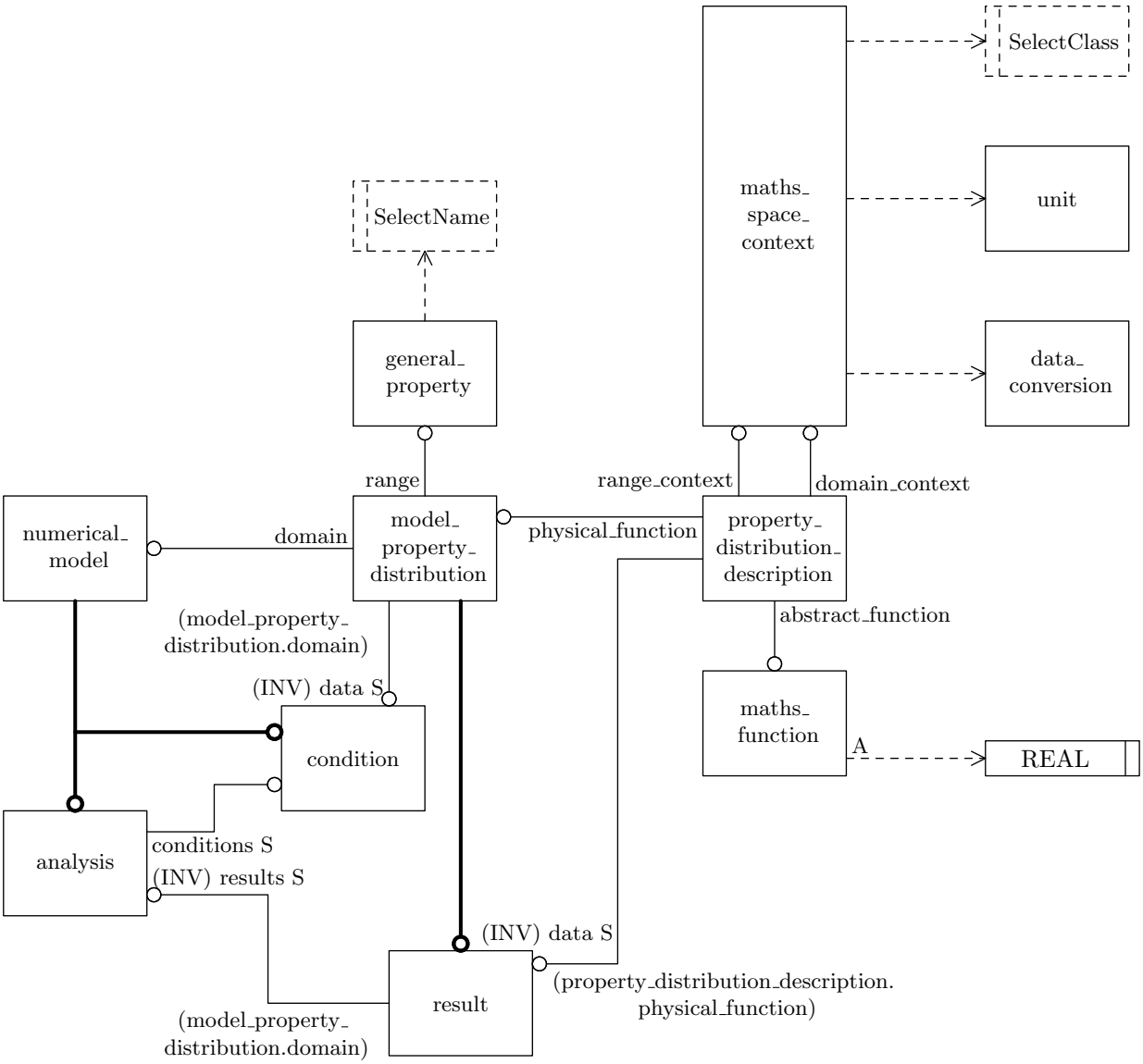

Figure 4 – EXPRESS-G **partial model sketching the skeleton of the analysis information model**

The information associated with the conceptual **data_array** is scattered among several entities. The data values are stored in a **maths_function**; the **property_distribution_description** relates a **maths_function** and a **model_property_distribution**. The **data_array**'s data name is accessible via the **model_property_distribution**. The **unit**, **data_conversion** and data class information is accessible via the **property_distribution_description**.

NOTE 6 Comparing the illustrations, there are explicit attributes shown in Figure 2 that are absent from Figure 3. In some cases these explicit attributes can be replaced by equivalent inverse attributes. In other cases, which are not shown, such as when an entity has multiple attributes that reference the conceptual **data_array**, this is not immediately possible because there is no means of distinguishing the different roles. Thus in general the model sketched in Figure 3 has to be modified to the model illustrated in Figure 4 to fully capture the semantics of Figure 2.

4.3 basis_schema type definitions

4.3.1 choose_general_property_identifier

A **choose_general_property_identifier** is a defined identifier for a **specified_general_property**. It is a superset of at least **externally_defined_item** and **defined_data_name**, and is the basis of a set of standardised data-names.

NOTE An **externally_defined_item** enables data-names to be accessed from an externally defined catalogues of names.

EXPRESS specification:

```
*)
TYPE choose_general_property_identifier = EXTENSIBLE SELECT
     (externally_defined_item,
      defined_data_name);
END_TYPE;
(*
```

4.3.2 choose_geometry_location

A **choose_geometry_location** is a location of some geometry data. It is a superset of at least **externally_defined_item** and **shape_representation**.

EXPRESS specification:

```
*)
TYPE choose_geometry_location = EXTENSIBLE SELECT
    (externally_defined_item,
     shape_representation);
END_TYPE;
(*
```

4.3.3 choose_representation_context_identifier

A **choose_representation_context_identifier** is a defined identifier for a **specified_representation_context**. It is a superset of at least **externally_defined_item** and **defined_data_class** and is the basis of a set of standardised identifiers of classes of data.

NOTE An **externally_defined_item** enables data class names to be accessed from an externally defined catalogues of names.

EXPRESS specification:

```
*)
TYPE choose_representation_context_identifier = EXTENSIBLE SELECT
     (externally_defined_item,
      defined_data_class);
END_TYPE;
(*
```


4.3.4 defined_data_class

A **defined_data_class** is a kind of defined identifier for classes of data.

EXPRESS specification:

```
*)
TYPE defined_data_class = EXTENSIBLE ENUMERATION OF
     (unspecified,
      application_defined,
      dimensional,
      normalised_by_dimensional,
      normalise_by_unknown_dimensional,
      dimensionless_parameter,
      dimensionless_constant);
END_TYPE;
(*
```

Enumerated item definitions:

unspecified: no identification;

application_defined: meaning is assigned by agreement external to this standard;

dimensional: identifies dimensional data;

normalised_by_dimensional: identifies nondimensional data that is normalised by dimensional reference quantities;

normalised_by_unknown_dimensional: identifies nondimensional data typically found in a completely nondimensional object base where all field and reference data is nondimensional;

nondimensional_parameter: identifies nondimensional parameters;

dimensionless_constant: identifies a constant value.

4.3.5 defined_data_name

A **defined_data_name** is a kind of defined identifier for data names. It is a superset of at least **coordinate_data_name** and **fd_defined_data_name**.

EXPRESS specification:

```
*)
TYPE defined_data_name = EXTENSIBLE SELECT
     (coordinate_data_name,
      fd_defined_data_name);
END_TYPE;
(*
```

4.3.6 fd_defined_data_name

A **fd_defined_data_name** is a kind of defined identifier for CFD data names. It is a superset of at least **fd_nondimensional_parameter_name**, **Riemann_1D_data_name**, **force_mo-**

ment_data_name, **gas_model_data_name**, **viscosity_model_data_name**, **thermal_conductivity_model_data_name**, **turbulence_closure_data_name**, **turbulence_model_data_name**, and **flow_solution_data_name**.

EXPRESS specification:

```
*)
TYPE fd_defined_data_name = EXTENSIBLE SELECT
     (fd_nondimensional_parameter_name,
      Riemann_1D_data_name,
      force_moment_data_name,
      gas_model_data_name,
      viscosity_model_data_name,
      thermal_conductivity_model_data_name,
      turbulence_closure_data_name,
      turbulence_model_data_name,
      flow_solution_data_name);
END_TYPE;
(*
```

4.3.7 fd_nondimensional_parameter_name

An **fd_nondimensional_parameter_name** is an enumeration of standardized nondimensional parameter data names.

CFD codes are rich in nondimensional governing parameters, such as Mach number and Reynolds number, and nondimensional flowfield coefficients, such as pressure coefficient. The problem with these parameters is that their definitions and conditions that they are evaluated at can vary from code to code. Reynolds number is particularly notorious in this respect.

These parameters have posed us with a difficult dilemma. Either we impose a rigid definition for each and force all database users to abide by it, or we develop some methodology for describing the particular definition that the user is employing. The first limits applicability and flexibility, and the second adds complexity. We have opted for the second approach, but we include only enough information about the definition of each parameter to allow for conversion operations. For example, the Reynolds number includes velocity, length and kinematic viscosity scales in its definition (i.e. $Re = VL/\nu$). The database description of Reynolds number includes these different scales. By providing these 'definition components', any code that reads Reynolds number from the database can transform its value to an appropriate internal definition.

Definitions for nondimensional flowfield coefficients follow: The pressure coefficient is defined as,

$$c_p = \frac{p - p_{\text{ref}}}{\frac{1}{2}\rho_{\text{ref}} q_{\text{ref}}^2},$$

where $\frac{1}{2}\rho_{\text{ref}} q_{\text{ref}}^2$ is the dynamic pressure evaluated at some reference condition, and p_{ref} is some reference pressure. The skin friction coefficient is,

$$\vec{c}_f = \frac{\vec{\tau}}{\frac{1}{2}\rho_{\text{ref}} q_{\text{ref}}^2},$$

where $\vec{\tau}$ is the shear stress or skin friction vector. Usually, $\vec{\tau}$ is evaluated at the wall surface.

Table 6 – Nondimensional parameter name identifiers

Data name identifier	Description	Units
Mach	Mach number: $M = q/c$	-
Mach_velocity	velocity scale (q)	$\mathbf{L/T}$
Mach_velocity_sound	speed of sound scale (c)	$\mathbf{L/T}$
Reynolds	Reynolds number: $Re = VL/\nu$	-
Reynolds_velocity	velocity scale (V)	$\mathbf{L/T}$
Reynolds_length	length scale (L)	$\mathbf{L}$
Reynolds_viscosity_kinematic	kinematic viscosity scale (ν)	$\mathbf{L^2/T}$
Prandtl	Prandtl number: $Pr = \mu c_p/k$	-
Prandtl_thermal_conductivity	thermal conductivity scale (k)	$\mathbf{ML}/(\mathbf{T}^3\Theta)$
Prandtl_viscosity_molecular	molecular viscosity scale (μ)	$\mathbf{M}/(\mathbf{LT})$
Prandtl_specific_heat_pressure	specific heat scale (c_p)	$\mathbf{L}^2/(\mathbf{T}^2\Theta)$
specific_heat_ratio	specific heat ratio: $\gamma = c_p/c_v$	-
specific_heat_ratio_pressure	specific heat at constant pressure (c_p)	$\mathbf{L}^2/(\mathbf{T}^2\Theta)$
specific_heat_ratio_volume	specific heat at constant volume (c_v)	$\mathbf{L}^2/(\mathbf{T}^2\Theta)$
coef_pressure	c_p	-
coef_pressure_dynamic	$1/2\rho_{\text{ref}}q_{\text{ref}}^2$	$\mathbf{M}/(\mathbf{LT}^2)$
coef_pressure_reference	p_{ref}	$\mathbf{M}/(\mathbf{LT}^2)$
coef_skin_friction_x	$\vec{c}_f \cdot \hat{e}_x$	-
coef_skin_friction_y	$\vec{c}_f \cdot \hat{e}_y$	-
coef_skin_friction_z	$\vec{c}_f \cdot \hat{e}_z$	-
length_reference	L_{ref}	$\mathbf{L}$

EXPRESS specification:

```
*)
TYPE fd_nondimensional_parameter_name = EXTENSIBLE ENUMERATION OF
     (Mach,
      Mach_velocity,
      Mach_velocity_sound,
      Reynolds,
      Reynolds_velocity,
      Reynolds_length,
      Reynolds_viscosity_kinematic,
      Prandtl,
      Prandtl_thermal_conductivity,
      Prandtl_viscosity_molecular,
      Prandtl_specific_heat_pressure,
      specific_heat_ratio,
      specific_heat_ratio_pressure,
      specific_heat_ratio_volume,
      coef_pressure,
      coef_skin_friction_x,
      coef_skin_friction_y,
      coef_skin_friction_z,
      coef_pressure_dynamic,
      coef_pressure_reference,
      length_reference);
END_TYPE;
(*
```

The meanings of the identifiers are given in Table 6.

4.3.8 mbna_property_distribution_select

An **mbna_property_distribution_select** is a **property_distribution_select** that includes **model_property_description**.

EXPRESS specification:

```
*)
TYPE mbna_property_distribution_select = EXTENSIBLE SELECT
     BASED_ON property_distribution_select WITH
     (model_property_distribution);
END_TYPE;
(*
```

4.3.9 mbna_value_context_select

An **mbna_value_context_select** is a **value_context_select** that includes **choose_representation_context_identifier**.

EXPRESS specification:

```
*)
TYPE mbna_value_context_select = EXTENSIBLE SELECT
     BASED_ON value_context_select WITH
     (choose_representation_context_identifier);
END_TYPE;
(*
```

4.4 basis_schema entity definitions

4.4.1 geometry_reference

A **geometry_reference** is a reference to geometric data with means of describing the purpose or meaning of the reference.

EXPRESS specification:

```
*)
ENTITY geometry_reference;
  description : text;
  data        : choose_geometry_location;
END_ENTITY;
(*
```

Attribute definitions:

description: annotation;

data: the geometry data reference.

4.4.2 specified_general_property

A **specified_general_property** is a kind of **general_property** where the identifier is one of a pre-defined set.

EXPRESS specification:

```
*)
ENTITY specified_general_property
  SUBTYPE OF (general_property);
  name_specifier : choose_general_property_identifier;
END_ENTITY;
(*
```

Attribute definitions:

name_specifier: the defined identifier for the **general_property**.

4.4.3 specified_representation_context

A **specified_representation_context** is a kind of **representation_context** where the identifier is one of a pre-defined set.

EXPRESS specification:

```
*)
ENTITY specified_representation_context
  SUBTYPE OF (representation_context);
  class_specifier : choose_representation_context_identifier;
END_ENTITY;
(*
```

Attribute definitions:

class_specifier: the defined identifier for the **representation_context**.

EXPRESS specification:

```
*)
END_SCHEMA; -- end of basis_schema
(*
```

5 Hierarchy

The following EXPRESS declaration begins the **hierarchy_schema** and identifies the necessary external references.

EXPRESS specification:

```
*)
SCHEMA hierarchy_schema;
  REFERENCE FROM analysis_schema                    -- ISO 10303-53
                (model_action_domain,
                 model_product_domain_with_mesh,
                 numerical_model);
  REFERENCE FROM conditions_schema                  -- ISO 10303-110
                (fd_zone_bc,
                 mbna_family,
                 mbna_integral_data,
                 mbna_reference_state,
                 mbna_zone_bc);
  REFERENCE FROM domain_schema                      -- ISO 10303-110
                (grid_coordinates);
  REFERENCE FROM equations_schema                   -- ISO 10303-110
                (flow_equation_set,
                 mbna_equation_set);
  REFERENCE FROM mesh_connectivity_schema           -- ISO 10303-52
                (multiple_mesh_block);
  REFERENCE FROM mesh_topology_schema               -- ISO 10303-52
                (structured_mesh,
                 unstructured_mesh);
  REFERENCE FROM results_schema                     -- ISO 10303-110
                (mbna_discrete_data,
                 mbna_history,
                 mbna_solution);

(*
```

NOTE The schemas referenced above can be found in the following parts of ISO 10303:

analysis_schema	part 53
conditions_schema	this part of ISO 10303
domain_schema	this part of ISO 10303
equations_schema	this part of ISO 10303
mesh_connectivity_schema	part 52
mesh_topology_schema	part 52
results_schema	this part of ISO 10303

Abbreviated names are used in identifiers of elements declared in this schema. Prefixes used in these identifiers have the following meanings:

fd fluid_dynamics;

mbna mesh_based_numerical_analysis.

5.1 Introduction

This schema defines and describes the top level structures for describing mesh-based CFD numerical analyses.

The major goal of this part of ISO 10303 is a high level, comprehensive and unambiguous description of the intellectual content of information that must be passed from code to code in mesh-based analysis systems. This information includes grids, solutions, multiblock interface connectivity, boundary-conditions, reference states, and dimensional units or normalization as-

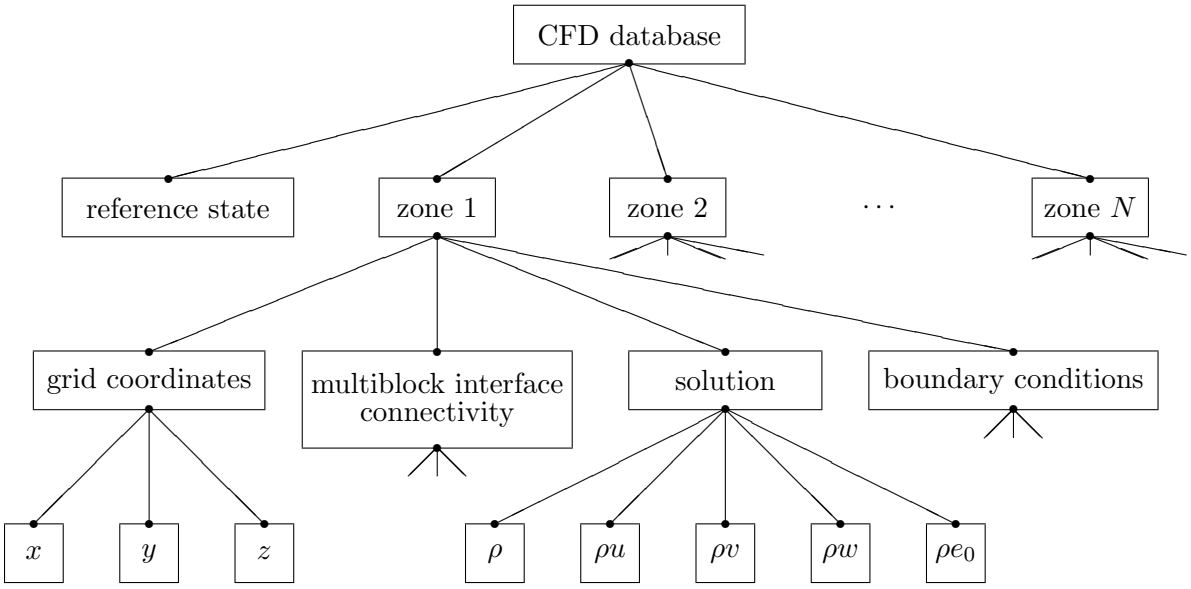

Figure 5 – Topologically based analysis hierarchy

sociated with data. The data sets typical of these kinds of analysis tend to contain a small number of extremely large data arrays.

The model describes a hierarchical database, precisely defining both the data and their hierarchical relationships.

There are two major alternatives to organizing an analysis hierarchy: topologically based and data-type based. In a topologically based graph, overall organization is by multiblock zones; information pertaining to a particular zone, including its grid coordinates or solution, hangs off the zone. In a data-type based graph, organization is by related data. For example, there would be two nodes at the same level, one for grid coordinates and another for the solution. Hanging off each of these nodes would be separate lists of the zones.

The hierarchy described here is topologically based; a simplified illustration of the database hierarchy is shown in Figure 5. Hanging off the root 'node' of the database is a node containing global reference-state information, such as general conditions, and a list of nodes for each zone. The figure shows the nodes that hang off the first zone; similar nodes would hang off of each zone in the database. Nodes containing the physical-coordinate data arrays (x, y and z) for the first zone are shown hanging off the 'grid coordinates' node. Likewise, nodes containing the first zone's solution data arrays hang off the 'solution' node. The figure also depicts nodes containing multiblock interface connectivity and boundary condition information for the first zone; subnodes hanging off each of these are not pictured.

Conceptually, an analysis can be thought of as having four components, as in this extremely brief model:

```
ENTITY analysis;
  domain     : computational_space;
  equations  : SET OF equation;
  conditions : SET OF condition;
  results    : SET OF solution;
END_ENTITY;
```


The **domain** is the representation of the computational space (typically a discrete approximation to a continuum). The **equations** are the mathematical formulations of the physics of the problem at hand and the **conditions** are the conditions (e.g., boundary conditions or constraints) that apply to the particular analysis. Finally, the **results** are the calculated solutions to the equations subject to the conditions over the domain, and possibly other data resulting from the analysis.

5.2 Fundamental concepts and assumptions

All information pertaining to a given zone, including grid coordinates, solution, and other related data, is contained within that zone's structure entity.

Structures for describing or annotating the database are provided; these same descriptive mechanisms are provided at all levels of the hierarchy.

Certain information, such as units of measure, may be defined at a high level in the hierarchy and apply to all lower levels, unless overriden at one of the lower levels. In turn, the lower level information applies to the subsequent even lower levels, unless overriden again. That is, default information of this kind specified at a low level takes precedence over that specified at a higher level.

5.3 hierarchy_schema entity definitions

5.3.1 fd_step

An **fd_step** is a single step, or one of a series of steps, in a CFD analysis.

EXPRESS specification:

```
*)
ENTITY fd_step
  SUBTYPE OF (mbna_step);
  SELF\mbna_step.equations : SET OF flow_equation_set;
  SELF\mbna_step.zones     : LIST OF fd_zone;
END_ENTITY;
(*
```

Attribute definitions:

equations: (inherited) description of the governing flow equations associated with the step.

zones: (inherited) data specific to each zone or block in a multiblock case.

5.3.2 fd_zone

An **fd_zone** is a kind of **mbna_zone** specifically for a CFD analysis.

EXPRESS specification:

```
*)
```



```
ENTITY fd_zone
  SUBTYPE OF (mbna_zone);
  SELF\mbna_zone.conditions : SET OF fd_zone_bc;
  SELF\mbna_zone.equations  : SET OF flow_equation_set;
END_ENTITY;
(*
```

Attribute definitions:

conditions: boundary-condition information;

equations: flow equations.

5.3.3 mbna_model

An **mbna_model** is the highest level structure in an analysis database. It contains a set of equations for solution and a list of the analysis steps performed. There may be associated globally applicable information.

EXPRESS specification:

```
*)
ENTITY mbna_model
  SUBTYPE OF (numerical_model);
  data      : SET OF mbna_integral_data;
  equations : SET OF mbna_equation_set;
  steps     : LIST OF mbna_step;
INVERSE
  history : SET OF mbna_history for domain;
END_ENTITY;
(*
```

Attribute definitions:

data: miscellaneous data; candidates for inclusion are global forces and moments;

equations: the description of the governing equations associated with the entire database. This structure contains information on the general class of governing equations, equations of state, and constants associated with the equations;

steps: the sequence of analysis steps performed;

history: globally relevant convergence history. The convergence information includes total configuration forces, global parameters, and global residual and solution-change norms taken over all the zones.

5.3.4 mbna_step

An **mbna_action** is a single step, or one of a series of steps, in an analysis. It contains a list of the zones making up the domain for the step. There may be associated globally applicable information.

EXPRESS specification:

```
*)
ENTITY mbna_step
  SUBTYPE OF (model_action_domain);
  data               : SET OF mbna_integral_data;
  equations          : SET OF mbna_equation_set;
  families           : SET OF mbna_family;
  refstate           : SET OF mbna_reference_state;
  zones              : LIST OF mbna_zone;
INVERSE
  history : SET OF mbna_history for domain;
END_ENTITY;
(*
```

Attribute definitions:

data: miscellaneous data; candidates for inclusion are global forces and moments;

equations: description of the governing equations associated with the step. This structure contains information on the general class of governing equations, equations of state, and constants associated with the equations;

families: global family information;

refstate: reference data applicable to the entire step;

zones: data specific to each zone or block in a multiblock case. The size of the list defines the number of zones or blocks in the domain;

history: globally relevant convergence history. The convergence information includes total configuration forces, global parameters, and global residual and solution-change norms taken over all the zones.

5.3.5 mbna_structured_zone

An **mbna_structured_zone** contains the information pertinent to an individual structured multiblock zone.

EXPRESS specification:

```
*)
ENTITY mbna_structured_zone
  SUBTYPE OF (mbna_zone);
  SELF\model_product_domain_with_mesh.model_mesh : structured_mesh;
END_ENTITY;
(*
```

Attribute definitions:

model_mesh: the **structured_mesh** describing the topology of the **mbna_zone**.

5.3.6 mbna_unstructured_zone

An **mbna_unstructured_zone** contains the information pertinent to an individual unstructured zone.

EXPRESS specification:

```
*)
ENTITY mbna_unstructured_zone
  SUBTYPE OF (mbna_zone);
  SELF\model_product_domain_with_mesh.model_mesh : unstructured_mesh;
END_ENTITY;
(*
```

Attribute definitions:

model_mesh: the **unstructured_mesh** describing the topology of the **mbna_zone**.

5.3.7 mbna_zone

An **mbna_zone** collects the information pertinent to an individual multiblock zone. This information includes the number of cells and vertices making up the grid, the physical coordinates of the grid vertices, the solution, multiblock interface connectivity, boundary-conditions, and zonal convergence-history data. In addition this structure contains a reference state and a set of governing equations that apply to the zone.

EXPRESS specification:

```
*)
ENTITY mbna_zone
  SUBTYPE OF (model_product_domain_with_mesh);
  cell_dimension     : INTEGER;
  conditions         : SET OF mbna_zone_bc;
  equations          : SET OF mbna_equation_set;
  families           : SET OF mbna_family;
  global_data        : SET OF mbna_integral_data;
  grid_connectivity  : OPTIONAL multiple_mesh_block;
  nindices           : INTEGER;
  physical_dimension : INTEGER;
  rstate             : SET OF mbna_reference_state;
INVERSE
  coordinates : SET OF grid_coordinates FOR domain;
  field_data  : SET OF mbna_discrete_data FOR domain;
  history     : SET OF mbna_history FOR domain;
  solution    : SET OF mbna_solution FOR domain;
END_ENTITY;

SUBTYPE_CONSTRAINT sc1_mbna_zone FOR mbna_zone;
  ABSTRACT SUPERTYPE;
  ONEOF(mbna_structured_zone,
        mbna_unstructured_zone);
END_SUBTYPE_CONSTRAINT;
```



```
(*
```

Attribute definitions:

cell_dimension: dimension of cells in the mesh;

conditions: boundary-condition information;

equations: the set of equations for the **zone**;

families: a family may be used to specify geometry of certain boundary-conditions associated with the **zone**;

global_data: miscellaneous zone-specific global data, other than reference-state data and convergence history information;

grid_connectivity: multiblock interface-connectivity information;

model_mesh: (inherited) mesh describing the topological shape of the **zone**;

nindices: number of indices required to identify uniquely a vertex or a cell in the grid. It is the indexical dimensionality of the computational grid. For structured-grid calculations, **nindices** is usually the same as the spatial problem being solved.

EXAMPLE 1 Usually **nindices**=3 for a 3-D problem.

For lower-dimensional flowfields, such as quasi 3-D flow, **nindices** may not be the same as the dimensionality of the position vector or the velocity vector. For unstructured grids, usually **nindices**=1 since all the grid points and flow solution variables are stored in 1-D arrays. However, there are instances, such as prismatic boundary-layer grids, where **nindices** may be 2.

physical_dimension: number of coordinates required to define a node position;

rstate: reference-state data;

coordinates: physical coordinates of the grid vertices. This structure defines the physical shape of the zone; it may optionally contain physical coordinates of rind or ghost points;

field_data: miscellaneous field data. Candidate information includes residuals, fluxes and other discrete data that is considered auxiliary to the solution;

history: the convergence history of the zone; this includes residual and solution-change norms;

solution: solution quantities. Each instance of **mbna_solution** shall only contain data at a single grid location (vertices, cell-centers, etc.); therefore, multiple **mbna_solution** structures are provided to store solution data at different grid locations. These structures may optionally contain solution data defined at rind points.

EXPRESS specification:

```
*)
END_SCHEMA;  -- end of hierarchy_schema
(*
```


6 Domain

The following EXPRESS declaration begins the **domain_schema** and identifies the necessary external references.

EXPRESS specification:

```
*)
SCHEMA domain_schema;
  REFERENCE FROM analysis_schema                    -- ISO 10303-53
                (model_product_domain_with_mesh,
                 model_property_distribution);
  REFERENCE FROM basis_schema                       -- ISO 10303-110
                (specified_general_property);
  REFERENCE FROM mathematical_description_of_distribution_schema  -- ISO 10303-51
                (property_distribution_description);
  REFERENCE FROM mesh_topology_schema               -- ISO 10303-52
                (rind);
(*
```

NOTE The schemas referenced above can be found in the following parts of ISO 10303:

analysis_schema	part 53
basis_schema	this part of ISO 10303
mathematical_description_of_distribution_schema	part 51
mesh_topology_schema	part 52

6.1 Introduction

This schema defines and describes the structure types for describing the grid coordinates pertaining to a grid.

6.2 domain_schema type definitions

6.2.1 coordinate_data_name

A **coordinate_data_name** is an enumeration of standardized coordinate systems data.

Coordinate systems for identifying physical location are as follows:

System	3–D	2–D
Cartesian	(x, y, z)	(x, y) or (x, z) or (y, z)
Cylindrical	(r, θ, z)	(r, θ)
Spherical	(r, θ, ϕ)	
Auxiliary	(ξ, η, ζ)	(ξ, η) or (ξ, ζ) or (η, ζ)

Associated with these coordinate systems are the following unit vector conventions:

x-direction $\hat{e}_x$	r-direction $\hat{e}_r$	ξ-direction $\hat{e}_\xi$
y-direction $\hat{e}_y$	θ-direction $\hat{e}_\theta$	η-direction $\hat{e}_\eta$
z-direction $\hat{e}_z$	ϕ-direction $\hat{e}_\phi$	ζ-direction $\hat{e}_\zeta$

Note that $\hat{e}_r$, $\hat{e}_\theta$ and $\hat{e}_\phi$ are functions of position.

Table 7 – Coordinate data name identifiers

Data name identifier	Description	Units
coordinate_x	x	**L**
coordinate_y	y	**L**
coordinate_z	z	**L**
coordinate_r	r	**L**
coordinate_theta	θ	α
coordinate_phi	ϕ	α
coordinate_normal	coordinate in direction of $\hat{e}_n$	**L**
coordinate_tangential	tangential coordinate (2–D only)	**L**
coordinate_xi	ξ	**L**
coordinate_eta	η	**L**
coordinate_zeta	ζ	**L**
coordinate_transform	transformation matrix (**T**)	-

It is expected that one of the 'standard' coordinate systems (cartesian, cylindrical or spherical) will be used within a zone (or perhaps the entire database) to define grid coordinates and other related data. The auxiliary coordinates will be used for special quantities, including forces and moments, which may not be defined in the same coordinate system as the rest of the data. When auxiliary coordinates are used, a transformation must also be provided to uniquely define them.

EXAMPLE 1 The transform from cartesian to orthogonal auxiliary coordinates is,

$$\begin{pmatrix} \hat{e}_\xi \\ \hat{e}_\eta \\ \hat{e}_\zeta \end{pmatrix} = \mathbf{T} \begin{pmatrix} \hat{e}_x \\ \hat{e}_y \\ \hat{e}_z \end{pmatrix},$$

where **T** is an orthonormal matrix (2×2 in 2–D and 3×3 in 3–D).

In addition, normal and tangential coordinates are often used to define boundary conditions and data related to surfaces. The normal coordinate is identified as n with the unit vector $\hat{e}_n$.

EXPRESS specification:

```
*)
TYPE coordinate_data_name = EXTENSIBLE ENUMERATION OF
     (coordinate_x,
      coordinate_y,
      coordinate_z,
      coordinate_r,
      coordinate_theta,
      coordinate_phi,
      coordinate_normal,
      coordinate_tangential,
      coordinate_xi,
      coordinate_eta,
      coordinate_zeta,
      coordinate_transform);
END_TYPE;
(*
```

The meanings of the enumerated coordinate data identifiers are given in Table 7.

6.3 domain_schema entity definitions

6.3.1 grid_coordinates

A **grid_coordinates** is the physical coordinates of the vertices of a grid.

EXPRESS specification:

```
*)
ENTITY grid_coordinates
  SUBTYPE OF (model_property_distribution);
  SELF\model_property_distribution.domain : model_product_domain_with_mesh;
  SELF\model_property_distribution.range  : specified_general_property;
INVERSE
  data : SET OF property_distribution_description FOR physical_function;
WHERE
  wr1 : is_coordinate_property(range);
END_ENTITY;
(*
```

Attribute definitions:

domain: a **model_product_domain_with_mesh**;

range: a **specified_general_property**;

data: the grid-coordinate data.

Formal propositions:

wr1: the model property distribution range shall be coordinate data.

6.3.2 grid_coordinates_with_rind

A **grid_coordinates_with_rind** is the physical coordinates of the vertices of a grid with a rind.

EXPRESS specification:

```
*)
ENTITY grid_coordinates_with_rind
  SUBTYPE OF (grid_coordinates);
  rind_planes : rind;
END_ENTITY;
(*
```

Attribute definitions:

rind_planes: the rind planes included in the data.

6.4 domain_schema function definitions

6.4.1 is_coordinate_property

The function **is_coordinate_property** returns TRUE if its argument denotes a coordinate.

EXPRESS specification:

```
*)
FUNCTION is_coordinate_property(arg : specified_general_property) : BOOLEAN;
  IF ('EXTERNAL_REFERENCE_SCHEMA.EXTERNALLY_DEFINED_ITEM' IN
      TYPEOF(arg) ) THEN
    RETURN(TRUE);
  END_IF;
  RETURN ('DOMAIN_SCHEMA.COORDINATE_DATA_NAME' IN TYPEOF(arg));
END_FUNCTION;
(*
```

Argument definitions:

arg: a **specified_general_property**;

RETURNS: TRUE if the **arg** is an **externally_defined_item** or a **coordinate_data_name**, otherwise FALSE.

EXPRESS specification:

```
*)
END_SCHEMA;  -- end of domain_schema
(*
```

7 Conditions

The following EXPRESS declaration begins the **conditions_schema** schema and identifies the necessary external references.

EXPRESS specification:

```
*)
SCHEMA conditions_schema;
  REFERENCE FROM analysis_schema               -- ISO 10303-53
                (model_property_distribution,
                 model_state_domain);
  REFERENCE FROM basis_schema                  -- ISO 10303-110
                (geometry_reference);
  REFERENCE FROM mesh_topology_schema          -- ISO 10303-52
                (indices_group,
                 indices_list,
                 mesh_location,
```



```
                 vertex_defined_cell);
  REFERENCE FROM support_resource_schema       -- ISO 10303-41
                (label,
                 text);
(*
```

NOTE The schemas referenced above can be found in the following parts of ISO 10303:

analysis_schema	part 53
basis_schema	this part of ISO 10303
mesh_topology_schema	part 52
support_resource_schema	part 41

Abbreviated names are used in identifiers of elements declared in this schema. Abbreviations used in these identifiers have the following meanings:

bc boundary_condition;

fd fluid_dynamics;

mbna mesh_based_numerical_analysis.

7.1 Introduction

This schema defines and describes the boundary-condition specifications for mesh-based analysis codes.

7.2 Fundamental concepts and assumptions

This model is an attempt to unify boundary-condition specifications within mesh-based numerical analysis codes. The structures and conventions presented are a compromise between simplicity and generality.

The difficulty with boundary-conditions is that there is such a wide variety used, and even a single boundary-condition equation is often implemented differently in different codes. Some boundary-conditions, such as a symmetry plane, are fairly well defined. Other boundary-conditions are much looser in their definition and implementation.

EXAMPLE 1 In CFD analyses an inflow boundary is an example of a loosely defined condition. It is generally accepted how many solution quantities should be specified at an inflow boundary (from mathematical well-posedness arguments), but what those quantities are will change with the class of flow problems (e.g., internal flows *vs.* external flows), and they will also change from code to code.

EXAMPLE 2 An additional difficulty for CFD analysis is that in some situations different boundary-condition equations are applied depending on local flow conditions. Any boundary where the flow can change from inflow to outflow or supersonic to subsonic is a candidate for flow-dependent boundary-condition equations.

These difficulties have moulded the design of the boundary-condition structures and conventions. Boundary-condition types are defined (**bc_type_simple**, **bc_type_compound**, and **bc_type**) that establish the equations to be enforced. However, for those more loosely defined boundary-conditions, such as inflow/outlow, the boundary-condition type merely establishes general guidelines on the equations to be imposed. Augmenting (and superseding) the information provided by the boundary-condition type is precisely defined boundary-condition solution data. Data-name conventions are used to identify the exact quantities involved in the boundary-conditions.

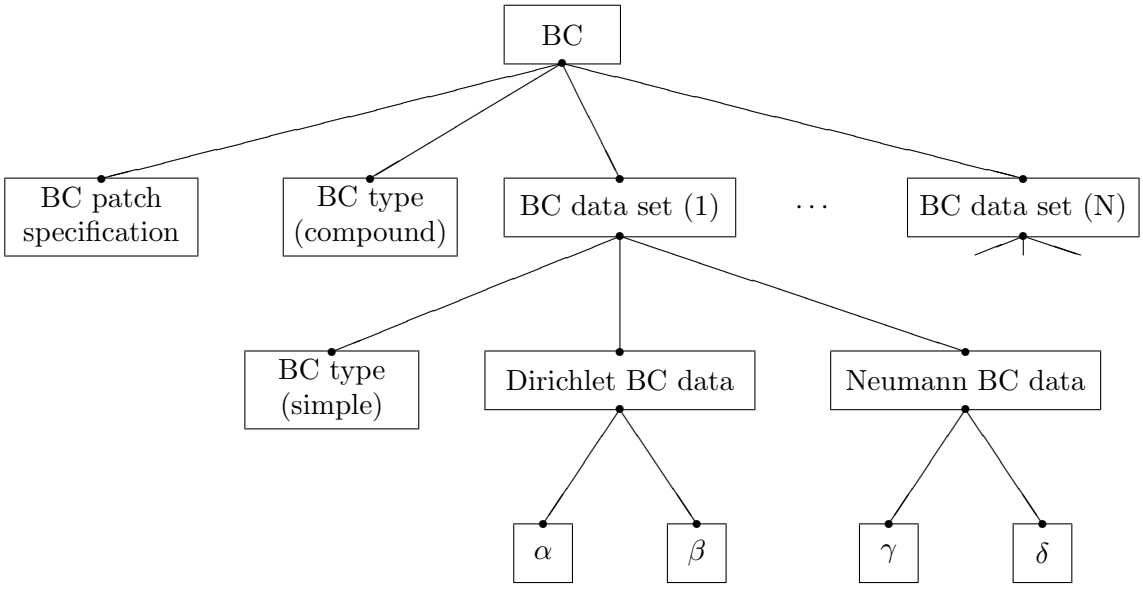

Figure 6 – Hierarchy for boundary-condition structures

One flexibility that is provided by this approach is that boundary-condition information can easily be built during the course of analysis. For example, during grid-generation phases minimal information (e.g., only the boundary-condition type) may be given. Then, prior to running of the solver, more specific boundary-condition information, such as Dirichlet or Neumann data, may be added.

Boundary-conditions are classified as either fixed or dependent. Fixed boundary-conditions enforce a given set of boundary-condition equations regardless of conditions; dependent boundary-conditions enforce different sets of boundary-condition equations depending on local conditions. Both fixed and dependent boundary-conditions are incorporated into a uniform framework, which allows all boundary-conditions to be described in a similar manner.

Figure 6 depicts the hierarchy used for prescribing a single boundary-condition. Each boundary-condition includes a type that describes the general equations to enforce, a patch specification, and a collection of data sets. The minimum required information for any boundary-condition is the patch specification and the boundary-condition type. This minimum information is similar to that used in many existing (flow) solvers.

Generality in prescribing equations to enforce and their associated boundary-condition data is provided in the optional data sets. Each data set contains all boundary-condition data required for a given fixed or simple boundary-condition. Each data set is also tagged with a boundary-condition type. For fixed boundary-conditions, the hierarchical tree contains a single data set, and the two boundary-condition types shown in Figure 6 are identical. Dependent or compound boundary-conditions contain multiple data sets, each to be applied separately depending on local conditions. The compound boundary-condition type describes the general dependent boundary-conditions, and each data set contains associated simple boundary-condition types.

EXAMPLE 3 In CFD analyses a farfield boundary condition would contain four data sets, where each applies to the different combinations of subsonic and supersonic inflow and outflow.

Within a single data set, boundary-condition data is grouped by equation type into Dirichlet and Neumann data. The lower leaves of Figure 6 show data for generic solution quantities α and β to be applied in Dirichlet conditions, and data for γ and δ to be applied in Neumann

boundary-conditions. **data_array** entities are employed to store these data and to identify the specific variables they are associated with.

In situations where the data sets (or any information contained therein) are absent from a given boundary-condition hierarchy, solvers are free to impose any appropriate boundary-conditions. Although not pictured in Figure 6, it is also possible to specify the reference state from which the solver should extract the boundary-condition data.

7.3 conditions_schema type definitions

7.3.1 fd_bc_type_compound

An **fd_bc_type_compound** is an enumeration type that identifies the CFD compound boundary-condition at a boundary location.

EXPRESS specification:

```
*)
TYPE fd_bc_type_compound = EXTENSIBLE ENUMERATION
     BASED_ON mbna_bc_type_compound WITH
     (bc_inflow,
      bc_outflow,
      bc_farfield);
END_TYPE;
(*
```

Enumerated item definitions:

unspecified: (from **mbna_bc_type_compound**) condition is unspecified;

application_defined: (from **mbna_bc_type_compound**) condition is specified via an external agreement between the data creator and the data user;

bc_inflow: inflow, arbitrary normal Mach
test on normal Mach, then perform one of: **bc_inflow_subsonic**, **bc_inflow_supersonic**;

bc_outflow: outflow, arbitrary normal Mach
test on normal Mach, then perform one of: **bc_outflow_subsonic**, **bc_outflow_supersonic**;

bc_farfield: farfield inflow/outflow, arbitrary normal Mach
test on normal velocity and normal Mach, then perform one of: **bc_inflow_subsonic**, **bc_inflow_supersonic**, **bc_outflow_subsonic**, **bc_outflow_supersonic**.

7.3.2 fd_bc_type_simple

An **fd_bc_type_simple** is an enumeration type that identifies the simple CFD boundary-condition at a boundary location.

In the descriptions below, Q is the solution vector, $\vec{q}$ is the velocity vector whose magnitude is q, the unit normal to the boundary is $\hat{n}$, and $\partial()/\partial n = \hat{n} \cdot \nabla$ is differentiation normal to the boundary.

For inflow/outflow boundary-condition descriptions, 3-D inviscid compressible flow is assumed;

the 2-D equivalent should be obvious. These same boundary-conditions are typically used for viscous cases also. This '3-D Euler' assumption will be noted wherever used.

EXPRESS specification:

```
*)
TYPE fd_bc_type_simple = EXTENSIBLE ENUMERATION
     BASED_ON mbna_bc_type_simple WITH
     (bc_wall_inviscid,
      bc_wall_viscous_heat_flux,
      bc_wall_viscous_isothermal,
      bc_wall_viscous,
      bc_wall,
      bc_inflow_subsonic,
      bc_inflow_supersonic,
      bc_outflow_subsonic,
      bc_outflow_supersonic,
      bc_tunnel_inflow,
      bc_tunnel_outflow,
      bc_degenerate_line,
      bc_degenerate_point,
      bc_symmetry_polar,
      bc_axissymmetric_wedge);
END_TYPE;
(*
```

Enumerated item definitions:

unspecified: (from **mbna_bc_type_simple**) condition is unspecified;

application_defined: (from **mbna_bc_type_simple**) condition is specified via an external agreement between the data creator and the data user;

bc_general: (from **mbna_bc_type_simple**) arbitrary conditions on Q or $\partial Q/\partial n$;

bc_Dirichlet: (from **mbna_bc_type_simple**) Dirichlet condition on Q vector;

bc_Neumann: (from **mbna_bc_type_simple**) Neumann condition on $\partial Q/\partial n$;

bc_extrapolate: (from **mbna_bc_type_simple**) extrapolate Q from interior;

bc_symmetry_plane: (from **mbna_bc_type_simple**) symmetry plane; face should be coplanar

— density, pressure: $\partial()/\partial n = 0$

— tangential velocity: $\partial(\vec{q} \times \hat{n})/\partial n = 0$

— normal velocity: $\vec{q} \cdot \hat{n} = 0$

bc_wall_inviscid: inviscid (slip) wall

— normal velocity specified (default: $\vec{q} \cdot \hat{n} = 0$)

bc_wall_viscous_heat_flux: viscous no-slip wall with heat flux

— velocity Dirichlet (default: $q = 0$)

— temperature Neumann (default: adiabatic, $\partial T/\partial n = 0$)

bc_wall_viscous_isothermal: viscous no-slip, isothermal wall

— velocity Dirichlet (default: $q = 0$)

— temperature Dirichlet

bc_wall_viscous: viscous no-slip wall; special cases are **bc_wall_viscous_heat_flux** and **bc_wall_viscous_isothermal**.

— velocity Dirichlet (default: $q = 0$)

— Dirichlet or Neumann on temperature

bc_wall: general wall condition; special cases are **bc_wall_inviscid**, **bc_wall_viscous**, **bc_wall_viscous_heat_flux**, and **bc_wall_viscous_isothermal**

bc_inflow_subsonic: inflow with subsonic normal velocity

— specify 4; extrapolate 1 (3-D Euler)

bc_inflow_supersonic: inflow with supersonic normal velocity

— specify 5; extrapolate 0 (3-D Euler)

same as **bc_Dirichlet**

bc_outflow_subsonic: outflow with subsonic normal velocity

— specify 1; extrapolate 0 (3-D Euler)

bc_outflow_supersonic: outflow with supersonic normal velocity

— specify 0; extrapolate 5 (3-D Euler)

same as **bc_Extrapolate**

bc_tunnel_inflow: tunnel inlet (subsonic normal velocity)

— specify cross-flow velocity, stagnation enthalpy, entropy

— extrapolate 1 (3-D Euler)

bc_tunnel_outflow: tunnel exit (subsonic normal velocity)

— specify static pressure

— extrapolate 4 (3-D Euler)

bc_degenerate_line: face degenerated to a line;

bc_degenerate_point: face degenerated to a point;

bc_symmetry_polar: polar-coordinate singularity line; special case of **bc_degenerate_line** where degenerate face is a straight line and flowfield has polar symmetry; $\hat{s}$ is singularity line tangential unit vector

— normal velocity: $\vec{q} \times \hat{s} = 0$

— all others: $\partial()/\partial n = 0$, n normal to $\hat{s}$

bc_axissymmetric_wedge: axisymmetric wedge; special case of **bc_degenerate_line** where degenerate face is a straight line

7.3.3 ijk_minmax

An **ijk_minmax** is an enumeration of i-min through k-max.

NOTE See Table 10 for one use of this type.

EXPRESS specification:

```
*)
TYPE ijk_minmax = ENUMERATION OF
     (i_min,
      j_min,
      k_min,
      i_max,
      j_max,
      k_max);
END_TYPE;
(*
```

7.3.4 mbna_bc_type

An **mbna_bc_type** is a superset of at least two kinds of boundary-condition types.

Boundary-condition types identify the equations that should be enforced at a given boundary location. The boundary-condition types are described by **mbna_bc_type**. Some members of **mbna_bc_type** completely identify the equations to impose, while others identify a general description of the class of boundary-condition equations to impose.

The subdivision of **mbna_bc_type** is based on function. For simple boundary-conditions, the equations and data are fixed; whereas, for compound boundary-conditions different sets of equations are imposed depending on local conditions at the boundary.

mbna_bc_type is a superset of **mbna_bc_type_simple** and **mbna_bc_type_compound**. It identifies the boundary-condition (simple or compound) at a boundary location.

EXPRESS specification:

```
*)
TYPE mbna_bc_type = EXTENSIBLE SELECT
     (mbna_bc_type_simple,
      mbna_bc_type_compound);
END_TYPE;
(*
```

7.3.5 mbna_bc_type_compound

An **mbna_bc_type_compound** is an enumeration type that identifies the compound boundary-condition at a boundary location.

EXPRESS specification:

```
*)
TYPE mbna_bc_type_compound = EXTENSIBLE ENUMERATION OF
     (unspecified,
      application_defined);
END_TYPE;
(*
```

Enumerated item definitions:

unspecified: condition is unspecified;

application_defined: condition is specified via an external agreement between the data creator and the data user;

7.3.6 mbna_bc_type_simple

An **mbna_bc_type_simple** is an enumeration type that identifies the simple boundary-condition at a boundary location.

In the descriptions below, Q is the solution vector, the unit normal to the boundary is $\hat{n}$, and $\partial()/\partial n = \hat{n} \cdot \nabla$ is differentiation normal to the boundary.

EXPRESS specification:

```
*)
TYPE mbna_bc_type_simple = EXTENSIBLE ENUMERATION OF
     (unspecified,
      application_defined,
      bc_general,
      bc_Dirichlet,
      bc_Neumann,
      bc_extrapolate,
      bc_symmetry_plane);
END_TYPE;
(*
```

Enumerated item definitions:

unspecified: condition is unspecified;

application_defined: condition is specified via an external agreement between the data creator and the data user;

bc_general: arbitrary conditions on Q or $\partial Q/\partial n$;

bc_Dirichlet: Dirichlet condition on Q vector;

bc_Neumann: Neumann condition on $\partial Q/\partial n$;

bc_extrapolate: extrapolate Q from interior;

bc_symmetry_plane: symmetry plane; face should be coplanar

Table 8 – Riemann 1-D data name identifiers

Data name identifier	Description	Units
Riemann_invariant_plus	$u + 2c/(\gamma - 1)$	$\mathbf{L/T}$
Riemann_invariant_minus	$u - 2c/(\gamma - 1)$	$\mathbf{L/T}$
characteristic_entropy	$p' - \rho'/\bar{c}^2$	$\mathbf{M/(LT^2)}$
characteristic_vorticity1	v'	$\mathbf{L/T}$
characteristic_vorticity2	w'	$\mathbf{L/T}$
characteristic_acoustic_plus	$p' + u'/(\bar{\rho}\bar{c})$	$\mathbf{M/(LT^2)}$
characteristic_acoustic_minus	$p' - u'/(\bar{\rho}\bar{c})$	$\mathbf{M/(LT^2)}$

7.3.7 Riemann_1D_data_name

A **Riemann_1D_data_name** is an enumeration of standardized Riemann data for 1-D flow.

Boundary condition specification for inflow/outflow or farfield boundaries often involves Riemann invariants or characteristics of the linearized inviscid flow equations. For an ideal compressible gas, these are typically defined as follows: Riemann invariants for an isentropic 1–D flow are,

$$\left[\frac{\partial}{\partial t} + (u \pm c)\frac{\partial}{\partial x}\right]\left(u \pm \frac{2}{\gamma - 1}c\right) = 0.$$

Characteristic variables for the 3–D Euler equations linearized about a constant mean flow are,

$$\left[\frac{\partial}{\partial t} + \bar{\Lambda}_n \frac{\partial}{\partial x}\right] W'_n(x,t) = 0, \qquad n = 1, 2, \ldots 5,$$

where the characteristics and corresponding characteristic variables are

Characteristic	$\bar{\Lambda}_n$	W'_n
'entropy'	$\bar{u}$	$p' - \rho'/\bar{c}^2$
'vorticity'	$\bar{u}$	v'
'vorticity'	$\bar{u}$	w'
'acoustic'	$\bar{u} \pm \bar{c}$	$p' \pm u'/(\bar{\rho}\bar{c})$

Barred quantities are evaluated at the mean flow, and primed quantities are linearized perturbations. The only non-zero mean-flow velocity component is $\bar{u}$.

The following data-name identifiers are defined for Riemann invariants and characteristic variables:

EXPRESS specification:

```
*)
TYPE Riemann_1D_data_name = EXTENSIBLE ENUMERATION OF
     (Riemann_invariant_plus,
      Riemann_invariant_minus,
      characteristic_entropy,
      characteristic_vorticity1,
      characteristic_vorticity2,
      characteristic_acoustic_plus,
      characteristic_acoustic_minus);
END_TYPE;
(*
```


The required identifiers and their meanings are given in Table 8.

7.4 conditions_schema entity definitions

7.4.1 elements_bc

An **elements_bc** is an **mbna_bc** containing boundary-condition information for a group of elements in an unstructured mesh.

EXPRESS specification:

```
*)
ENTITY elements_bc
  SUBTYPE OF (mbna_bc);
  elements : LIST OF vertex_defined_cell;
END_ENTITY;
(*
```

Attribute definitions:

elements: the elements (in an unstructured mesh).

7.4.2 fd_bc

An **fd_bc** is an **mbna_bc** containing boundary-condition information for a single BC surface patch of a CFD zone.

EXPRESS specification:

```
*)
ENTITY fd_bc
  SUBTYPE OF (mbna_bc);
  SELF\mbna_bc.datasets : SET OF fd_bc_dataset;
END_ENTITY;
(*
```

Attribute definitions:

datasets: (inherited) a list of boundary-condition data sets. In general, the proper **fd_bc_dataset** instance(s) to impose on the BC patch is determined by the value of **the_type**.

7.4.3 fd_bc_dataset

An **fd_bc_dataset** is an **mbna_bc_dataset** for CFD. Its intended use is for simple boundary-condition types, where the equations imposed do not depend on local flow conditions.

Boundary-condition data is separated by equation type into Dirichlet and Neumann conditions. Dirichlet boundary-conditions impose the value of the given variables, whereas Neumann boundary-conditions impose the normal derivative of the given variables.

Table 9 – Associated boundary-condition types and usage rules

bc_type Identifier	**Associated bc_type_simple identifiers and usage rules**
bc_inflow	**bc_inflow_supersonic** **bc_inflow_subsonic** *usage rule:* if supersonic normal Mach, choose **bc_inflow_supersonic**, else choose **bc_inflow_subsonic**.
bc_Outflow	**bc_outflow_supersonic** **bc_outflow_subsonic** *usage rule:* if supersonic normal Mach, choose **bc_outflow_supersonic**, else choose **bc_outflow_subsonic**.
bc_farfield	**bc_inflow_supersonic** **bc_inflow_subsonic** **bc_outflow_supersonic** **bc_outflow_subsonic** *usage rule:* if inflow and supersonic normal Mach, choose **bc_inflow_supersonic**, else if inflow, choose **bc_inflow_subsonic**, else if outflow and supersonic normal Mach, choose **bc_outflow_supersonic**, else, choose **bc_outflow_subsonic**.
bc_inflow_supersonic	**bc_inflow_supersonic** **bc_Dirichlet** *usage rule:* choose either; **bc_inflow_supersonic** takes precedence.
bc_outflow_supersonic	**bc_outflow_supersonic** **bc_Extrapolate** *usage rule:* choose either; **bc_outflow_supersonic** takes precedence.
all others	self-matching

The **bc** structure (see 7.4.7) allows for an arbitrary list of boundary-condition data sets, described by the **bc_dataset** structure. For simple boundary-conditions, a single data set must be chosen from a list that may contain more than one element. Likewise, for a compound boundary-condition, a limited number of data sets must be chosen and applied appropriately. The mechanism for proper choice of data sets is controlled by the **the_type** attribute of the **bc** structure, the **the_type** attribute of the **bc_dataset** structure, and the boundary-condition type association table (Table 9).

EXPRESS specification:

```
*)
ENTITY fd_bc_dataset
  SUBTYPE OF (mbna_bc_dataset);
```



```
  SELF\mbna_bc_dataset.the_type : fd_bc_type_simple;
END_ENTITY;
(*
```

Attribute definitions:

the_type: the boundary-condition type, which gives general information on the boundary-condition equations to be enforced.

7.4.4 fd_zone_bc

An **fd_zone_bc** is the boundary-condition information pertaining to a zone.

EXPRESS specification:

```
*)
ENTITY fd_zone_bc
  SUBTYPE OF (mbna_zone_bc);
  SELF\mbna_zone_bc.conditions : SET OF fd_bc;
END_ENTITY;
(*
```

Attribute definitions:

conditions: (inherited) the boundary-conditions for a zone, on a patch by patch basis.

7.4.5 indexed_elements_bc

An **element_list_bc** is an **mbna_bc** containing boundary-condition information for cells in a structured mesh.

EXPRESS specification:

```
*)
ENTITY indexed_elements_bc
  SUBTYPE OF (mbna_bc);
  element_indices : indices_group;
END_ENTITY;
(*
```

Attribute definitions:

element_indices: the element indices.

7.4.6 indexed_points_bc

An **indexed_points_bc** is an **mbna_bc** containing boundary-condition information for points in a structured mesh.

EXPRESS specification:

```
*)
ENTITY indexed_points_bc
  SUBTYPE OF (mbna_bc);
  point_indices : indices_group;
END_ENTITY;
(*
```

Attribute definitions:

point_indices: a group of indices; by convention the indices refer to vertices.

Informal propositions:

ip1: if **inward_normal_list** is specified, then an ordering convention is needed for indices on the BC patch. An ordering is also needed if local data is present in the **bc_dataset** substructures. FORTRAN multidimensional array ordering shall be used.

7.4.7 mbna_bc

An **mbna_bc** is boundary-condition information for a single BC surface patch of a zone. A BC patch is the subrange of the face of a zone where a given boundary-condition is applied.

The structure contains a boundary-condition type, as well as one or more sets of boundary-condition data that are used to define the boundary-condition equations to be enforced on the BC patch. For most boundary-conditions, a single data set is all that is needed. The structure also contains information describing the normal vector to the BC surface patch.

EXPRESS specification:

```
*)
ENTITY mbna_bc
  SUBTYPE OF (mbna_condition);
  datasets            : SET OF mbna_bc_dataset;
  family              : SET OF mbna_family;
  gridloc             : mesh_location;
  inward_normal_index : OPTIONAL ijk_minmax;
  inward_normal_list  : OPTIONAL indices_list;
  rstate              : SET OF mbna_reference_state;
  the_type            : mbna_bc_type;
END_ENTITY;

SUBTYPE_CONSTRAINT sc1_mbna_bc FOR mbna_bc;
  ONEOF(elements_bc,
        indexed_elements_bc,
        indexed_points_bc);
END_SUBTYPE_CONSTRAINT;
(*
```


Table 10 – inward_normal_index values

Face	Value	Face	Value
i_min	$[+1, 0, 0]$	**i_max**	$[-1, 0, 0]$
j_min	$[0, +1, 0]$	**j_max**	$[0, -1, 0]$
k_min	$[0, 0, +1]$	**k_max**	$[0, 0, -1]$

Attribute definitions:

datasets: a list of boundary-condition data sets. In general, the proper **bc_dataset** instance(s) to impose on the BC patch is determined by the value of **the_type**.

For a few boundary-conditions, such as a symmetry plane or polar singularity, the value of **the_type** completely describes the equations to impose, and no instances of **bc_dataset** are needed. For 'simple' boundary-conditions, where a single set of Dirichlet and/or Neumann data is applied a single **bc_dataset** will likely be used (although this is not a requirement). For 'compound' boundary-conditions, where the equations to impose are dependent on local conditions, several instances of **bc_dataset** will likely be used.

family: a collection of family boundary-condition;

gridloc: the location of the conditions, which are normally either at the cell vertices or on the cell faces.

Some boundary-conditions require a normal direction to be specified in order to be properly imposed. A computational-coordinate normal can be derived from **point_range** or **point_list** by examining redundant index components. Alternatively, this information can be provided directly by **inward_normal_index**. For exterior faces of a zone in 3-D, **inward_normal_index** takes one of the values given in Table 10.

inward_normal_list: a list of vectors normal to the BC patch pointing into the interior of the zone; the vectors are not required to be unit vectors. By convention the vectors are located at the vertices of the BC patch.

The physical-space normal vectors of the BC patch may be described by **inward_normal_list**; these are located at vertices, consistent with **point_range** and **point_list**. **inward_normal_list** is specified as an optional attribute because it is not always needed to enforce boundary-conditions, and the physical-space normals of a BC patch can usually be constructed from the grid. However, there are some situations, such as grid-coordinate singularity lines, where **inward_normal_list** becomes a required attribute because the normals cannot be generated from other information.

rstate: reference data applicable to the conditions of the BC patch;

the_type: the type of the boundary-condition.

7.4.8 mbna_bc_data

An **mbna_bc_data** is boundary-condition data.

By convention all data specified in a given instance of **mbna_bc_data** is to be used in the same *type* of boundary-condition equation.

EXAMPLE The Dirichlet and Neumann conditions in **mbna_bc_dataset** use separate **mbna_bc_data** structures.

EXPRESS specification:

```
*)
ENTITY mbna_bc_data
  SUBTYPE OF (mbna_condition);
END_ENTITY;

SUBTYPE_CONSTRAINT sc1_mbna_bc_data FOR mbna_bc_data;
  ABSTRACT SUPERTYPE;
END_SUBTYPE_CONSTRAINT;
(*
```

7.4.9 mbna_bc_data_global

An **mbna_bc_data_global** is global boundary-condition data.

EXAMPLE If a Dirichlet condition consists of a uniform stagnation pressure but with with a non-uniform velocity profile, then the stagnation pressure can be described by **mbna_bc_data_global**.

EXPRESS specification:

```
*)
ENTITY mbna_bc_data_global
  SUBTYPE OF (mbna_bc_data);
INVERSE
  data_global : SET OF model_property_distribution FOR domain;
END_ENTITY;
(*
```

Attribute definitions:

data_global: global data.

7.4.10 mbna_bc_data_local

An **mbna_bc_data_local** is local boundary-condition data.

EXAMPLE If a Dirichlet condition consists of a uniform stagnation pressure but with with a non-uniform velocity profile, then the velocity profile can be described by **mbna_bc_data_local**.

EXPRESS specification:

```
*)
ENTITY mbna_bc_data_local
  SUBTYPE OF (mbna_bc_data);
INVERSE
  data_local : SET OF model_property_distribution FOR domain;
END_ENTITY;
(*
```


Attribute definitions:

data_local: local data.

7.4.11 mbna_bc_dataset

An **mbna_bc_dataset** is Dirichlet or Neumann data for a single set of boundary-condition equations. Its intended use is for simple boundary-condition types, where the equations imposed do not depend on local conditions.

Boundary-condition data is separated by equation type into Dirichlet and Neumann conditions. Dirichlet boundary-conditions impose the value of the given variables, whereas Neumann boundary-conditions impose the normal derivative of the given variables.

The **mbna_bc** structure (see 7.4.7) allows for an arbitrary list of boundary-condition data sets, described by the **mbna_bc_dataset** structure. For simple boundary-conditions, a single data set must be chosen from a list that may contain more than one element. Likewise, for a compound boundary-condition, a limited number of data sets must be chosen and applied appropriately. The mechanism for proper choice of data sets can be controlled by the **the_type** attribute of the **mbna_bc** structure together with the **the_type** attribute of the **mbna_bc_dataset** structure.

mbna_bc is used for both simple and compound boundary-conditions; hence, its attribute **the_type** is of type **mbna_bc_type**. Conversely, the structure **mbna_bc_dataset** is intended to enforce a single set of boundary-condition equations independent of local flow conditions (i.e., it is appropriate only for simple boundary-conditions). That is why its attribute **the_type** is of type **mbna_bc_type_simple** and not **mbna_bc_type**.

NOTE 1 The appropriate choice of data sets may be specified by listing appropriate pairings of the values of **mbna_bc.the_type** and **mbna_bc_dataset.the_type**.

Although the model has a strict division between the two categories of boundary-condition types, in practice some overlap may exist. These complications require further guidelines on appropriate definition and use of boundary-condition types. The real distinctions between **mbna_bc_type_simple** and **mbna_bc_type_compound** are as follows:

— **mbna_bc_type_simple** identifiers always match themselves; **mbna_bc_type_compound** never match themselves.

— **mbna_bc_type_simple** identifiers always produce a single match; **mbna_bc_type_compound** produce multiple matches.

— The usage rule for **mbna_bc_type_simple** identifiers is always trivial — apply the single matching data set regardless of local conditions.

Therefore, any boundary-condition that involves application of different data sets depending on local conditions should be classified as **mbna_bc_type_compound**.

NOTE 2 If a type that is classified **mbna_bc_type_simple** is desired to be used as a compound, somehow be reclassified. One option is to define a new **mbna_bc_type_compound** identifier and provide associated **mbna_bc_type_simple** types and a usage rule. Another option may be to allow some identifiers to be both **mbna_bc_type_simple** and **mbna_bc_type_compound** and let their appropriate use be based on context.

For a given simple boundary-condition the database provides a set of boundary-condition equations to be enforced through the definitions of **mbna_bc_dataset** and **mbna_bc_data**. Apart from the boundary-condition type, the precise equations to be enforced are described by boundary-condition solution data. These specified solution data are arranged by 'equation type':
Dirichlet: $Q = (Q)_{\text{specified}}$
Neumann: $\partial Q/\partial n = (\partial Q/\partial n)_{\text{specified}}$

The **Dirichlet_data** and **Neumann_data** attributes (of type **mbna_bc_data**) list both the solution variables involved in the equations (through data-name conventions) and the specified solution data.

Two issues need to be addressed for specifying Dirichlet or Neumann boundary-condition data. The first is whether the data is global or local.

The global versus local issue can easily be handled by storing a scalar for the global BC data case, and storing an array for the local BC data case.

By convention, if the **Dirichlet_data** and **Neumann_data** are not present in an instance of **mbna_bc_dataset**, then application codes are free to enforce appropriate boundary-conditions for the given type of **mbna_bc_type_simple**. Furthermore, if insufficient data is present then application codes are free to fill out the boundary-condition data as appropriate for the **mbna_bc_type_simple** identifier.

To facilitate implementation of boundary-conditions into existing solvers, if no boundary-condition data is specified, solvers are free to enforce any appropriate boundary-condition equations. This includes situations where instances of **mbna_bc_dataset**, **mbna_bc_data** or **data_array** are absent within the boundary-condition hierarchy. In this case the **mbna_reference_state** specifies the reference-state conditions from which the solver should extract the boundary-condition data. Within the boundary-condition hierarchy, **mbna_reference_state** instances may be present at any of the **zone_bc**, **mbna_bc** or **mbna_bc_dataset** levels with the lowest taking precedence.

EXPRESS specification:

```
*)
ENTITY mbna_bc_dataset
  SUBTYPE OF (mbna_condition);
  gridloc  : mesh_location;
  rstate   : SET OF mbna_reference_state;
  the_type : mbna_bc_type_simple;
END_ENTITY;

SUBTYPE_CONSTRAINT sc1_mbna_bc_dataset FOR mbna_bc_dataset;
  ONEOF(mbna_Dirichlet_bc_dataset,
        mbna_Neumann_bc_dataset);
END_SUBTYPE_CONSTRAINT;
(*
```


Attribute definitions:

gridloc: the location of local data arrays (if any) provided in **mbna_Dirichlet_bc_dataset** or **mbna_Neumann_bc_dataset**; local boundary-condition data may be defined either at vertices or boundary face centers;

rstate: reference data applicable to the set of boundary-condition data;

the_type: the boundary-condition type, which gives general information on the boundary-condition equations to be enforced.

7.4.12 mbna_condition

An **mbna_condition** represents the concept of conditions or constraints pertinent to to an analysis.

EXPRESS specification:

```
*)
ENTITY mbna_condition
  SUBTYPE OF (model_state_domain);
END_ENTITY;

SUBTYPE_CONSTRAINT sc1_mbna_condition FOR mbna_condition;
  ABSTRACT SUPERTYPE;
  ONEOF(mbna_zone_bc,
        mbna_bc,
        mbna_bc_dataset,
        mbna_bc_data,
        mbna_reference_state,
        mbna_integral_data);
END_SUBTYPE_CONSTRAINT;
(*
```

7.4.13 mbna_Dirichlet_bc_dataset

An **mbna_Dirichlet_bc_dataset** is an **mbna_bc_dataset** that contains Dirichlet boundary condition data.

EXPRESS specification:

```
*)
ENTITY mbna_Dirichlet_bc_dataset
  SUBTYPE OF (mbna_bc_dataset);
  data : mbna_bc_data;
END_ENTITY;
(*
```

Attribute definitions:

data: boundary-condition data for Dirichlet conditions which may be constant over the BC patch or defined locally at each point of the patch.

7.4.14 mbna_family

An **mbna_family** is an association of set of boundary-condition with a set of geometric elements.

EXPRESS specification:

```
*)
ENTITY mbna_family;
  description  : text;
  id           : label;
  conditions   : SET OF mbna_bc_type;
  geometry     : SET OF geometry_reference;
END_ENTITY;
(*
```

Attribute definitions:

description: annotation;

id: user-specified instance identifier;

conditions: the family's boundary conditions;

geometry: the family's geometric information.

7.4.15 mbna_integral_data

An **mbna_integral_data** is a description of generic global or integral data that may be associated with a particular zone or an entire database. In contrast to **mbna_discrete_data**, integral data is not associated with any specific field location.

EXPRESS specification:

```
*)
ENTITY mbna_integral_data
  SUBTYPE OF (mbna_condition);
INVERSE
  data : SET OF model_property_distribution FOR domain;
END_ENTITY;
(*
```

Attribute definitions:

data: the data.

7.4.16 mbna_Neumann_bc_dataset

An **mbna_Neumann_bc_dataset** is an **mbna_bc_dataset** that contains Neumann boundary condition data.

Table 11 – Data that may be associated with mbna_reference_state

data_name	Description	Units
Mach	Mach number, $M = q/c$	—
Mach_velocity	Velocity scale, q	**L/T**
Mach_velocity_sound	Speed of sound scale, c	**L/T**
Reynolds	Reynolds number, $Re = VL_R/\nu$	—
Reynods_velocity	Velocity scale, V	**L/T**
Reynolds_length	Length scale, L_R	**L**
Reynolds_viscosity_kinematic	Kinematic viscosity scale, ν	$\mathbf{L^2/T}$
length_reference	Reference length, L_{ref}	**L**

EXPRESS specification:

```
*)
ENTITY mbna_Neumann_bc_dataset
  SUBTYPE OF (mbna_bc_dataset);
  data : mbna_bc_data;
END_ENTITY;
(*
```

Attribute definitions:

data: boundary-condition data for Neumann conditions which may be constant over the BC patch or defined locally at each point of the patch.

7.4.17 mbna_reference_state

An **mbna_reference_state** is a reference state, which is a list of geometric or state quantities defined at a common location or condition.

EXAMPLE Typical reference states associated with CFD calculations are freestream, plenum, stagntation, inlet and exit.

Data that may be associated with an **mbna_reference_state** is listed in Table 11. The meaning of the identifiers is described in 4.3.7.

EXPRESS specification:

```
*)
ENTITY mbna_reference_state
  SUBTYPE OF (mbna_condition);
INVERSE
  data : SET OF model_property_distribution FOR domain;
END_ENTITY;
(*
```


Attribute definitions:

data: the reference state data.

7.4.18 mbna_zone_bc

An **mbna_zone_bc** is the boundary-condition information for a zone.

EXPRESS specification:

```
*)
ENTITY mbna_zone_bc
  SUBTYPE OF (mbna_condition);
  conditions : SET OF mbna_bc;
  rstate     : SET OF mbna_reference_state;
END_ENTITY;
(*
```

Attribute definitions:

conditions: the boundary-conditions for a zone, on a patch by patch basis. Boundary-condition information for a single patch is contained in the **bc** structure. If a zone contains N boundary-condition patches, then N separate instances of **bc** shall be provided in the **zone_bc** entity for the zone.

rstate: reference data. Reference data is applicable to all the boundary-condition of the zone. Reference state data is useful for situations where boundary-condition data is not provided, and solvers are free to enforce any appropriate boundary-condition equations.

EXAMPLE An engine nozzle exit boundary-condition usually imposes a stagnation pressure (or some other stagnation quantity) different from freestream. The nozzle-exit stagnation quantities could be specified by reference data at this level or below in lieu of providing explicit Dirichlet or Neumman data.

EXPRESS specification:

```
*)
END_SCHEMA;  -- end of conditions_schema
(*
```

8 Equations

The following EXPRESS declaration begins the **equations_schema** and identifies the necessary external references.

EXPRESS specification:

```
*)
SCHEMA equations_schema;
```



```
  REFERENCE FROM analysis_schema               -- ISO 10303-53
                (model_product_distribution,
                 model_product_domain,
                 model_property_distribution);
(*
```

NOTE The schemas referenced above can be found in the following parts of ISO 10303:
analysis_schema part 53

Abbreviated names are used in identifiers of elements declared in this schema. Prefixes used in these identifiers have the following meanings:

fd fluid_dynamics;

mbna mesh_based_numerical_analysis.

8.1 Introduction

This schema defines and describes governing equation set associated with a mesh-based analysis.

8.2 Fundamental concepts and assumptions

The description of the equation set includes the general class of governing equations and mathematical models. Included with each equation description are associated constants.

The description of the flow-equation set includes the general class of governing equations, the turbulent closure equations, the gas model, and the viscosity and thermal-conductivity models. Included with each equation description are associated constants.

The structure definitions attempt to balance the opposing requirements of initial ease of implementation against requirements for extensibilty and future growth.

The intended use of these structures is primarily for archival purposes and to provide additional documentation of the solution.

NOTE 1 As an exampler, one small set of data name identifiers is specified for force and moment data which may be common to both structural and CFD analyses. It is expected that this set will be extended, and other sets introduced, for specific kinds of analyses.

NOTE 2 As an exampler, one small model for relating thermal properties is specified, which may cover both thermal and CFD analyses. It is expected that this model may have to be extended, and other kinds of model introduced, for specific kinds of analyses.

8.3 equations_schema type definitions

8.3.1 fd_behaviour_models

An **fd_behaviour_models** is a selection among CFD behavioural models.

EXPRESS specification:

```
*)
```



```
TYPE fd_behaviour_models = EXTENSIBLE SELECT
  (gas_model,
   turbulence_closure,
   turbulence_model,
   viscosity_model);
END_TYPE;
(*
```

8.3.2 fd_governing_equation_type

An **fd_governing_equation_type** is an enumeration of the classes of flow equations.

EXPRESS specification:

```
*)
TYPE fd_governing_equation_type = EXTENSIBLE ENUMERATION
     BASED_ON mbna_governing_equation_type WITH
     (full_potential,
      Euler,
      NS_laminar,
      NS_turbulent,
      NS_laminar_incompressible,
      NS_turbulent_incompressible);
END_TYPE;
(*
```

Enumerated item definitions:

unspecified: (from **governing_equations_type**) is unspecified;

application_defined: (from **governing_equations_type**) is specified via an external agreement between the data creator and the data user;

full_potential: is full potential flow;

Euler: is Euler flow;

NS_laminar: is Navier-Stokes laminar flow;

NS_turbulent: is Navier-Stokes turbulent flow;

NS_laminar_incompressible: is Navier-Stokes laminar incrompressible flow;

NS_turbulent_incompressible: is Navier-Stokes turbulent incrompressible flow;

8.3.3 force_moment_data_name

A **force_moment_data_name** is an enumeration of standardized force and moment data.

Conventions for data-name identifiers for forces and moments are defined in this subclause.

Given a differential force $\vec{f}$ (i.e. a force per unit area), the force integrated over a surface is,

$$\vec{F} = F_x\hat{e}_x + F_y\hat{e}_y + F_z\hat{e}_z = \int \vec{f}\, dA,$$

where $\hat{e}_x$, $\hat{e}_y$ and $\hat{e}_z$ are the unit vectors in the x, y and z directions, respectively. The moment about a point $\vec{r}_0$ integrated over a surface is,

$$\vec{M} = M_x\hat{e}_x + M_y\hat{e}_y + M_z\hat{e}_z = \int (\vec{r} - \vec{r}_0) \times \vec{f}\, dA.$$

Lift and drag components of the integrated force are,

$$L = \vec{F} \cdot \hat{L} \qquad D = \vec{F} \cdot \hat{D}$$

where $\hat{L}$ and $\hat{D}$ are the unit vectors in the positive lift and drag directions, respectively.

Lift, drag and moment are often computed in auxiliary coordinate frames (e.g. wind axes or stability axes). We introduce the convention that lift, drag and moment are computed in the (ξ, η, ζ) coordinate system. Positive drag is assumed parallel to the ξ-direction (i.e. $\hat{D} = \hat{e}_\xi$); and positive lift is assumed parallel to the η-direction (i.e. $\hat{L} = \hat{e}_\eta$). Thus, forces and moments defined in this auxiliary coordinate system are,

$$L = \vec{F} \cdot \hat{e}_\eta \qquad D = \vec{F} \cdot \hat{e}_\xi$$

$$\vec{M} = M_\xi\hat{e}_\xi + M_\eta\hat{e}_\eta + M_\zeta\hat{e}_\zeta = \int (\vec{r} - \vec{r}_0) \times \vec{f}\, dA.$$

Lift, drag and moment coefficients in 3–D are defined as,

$$C_L = \frac{L}{\frac{1}{2}\rho_{\text{ref}} q_{\text{ref}}^2 S} \qquad C_D = \frac{D}{\frac{1}{2}\rho_{\text{ref}} q_{\text{ref}}^2 S} \qquad \vec{C}_M = \frac{\vec{M}}{\frac{1}{2}\rho_{\text{ref}} q_{\text{ref}}^2 c_{\text{ref}} S_{\text{ref}}},$$

where $\frac{1}{2}\rho_{\text{ref}} q_{\text{ref}}^2$ is a reference dynamic pressure, S_{ref} is a reference area, and c_{ref} is a reference length. For a wing, S_{ref} is typically the wing area and c_{ref} is the mean aerodynamic chord. In 2–D, the sectional force coefficients are,

$$c_l = \frac{L'}{\frac{1}{2}\rho_{\text{ref}} q_{\text{ref}}^2 c_{\text{ref}}} \qquad c_d = \frac{D'}{\frac{1}{2}\rho_{\text{ref}} q_{\text{ref}}^2 c_{\text{ref}}} \qquad \vec{c}_m = \frac{\vec{M}'}{\frac{1}{2}\rho_{\text{ref}} q_{\text{ref}}^2 c_{\text{ref}}^2},$$

where the forces are integrated along a contour (e.g. an airfoil cross-section) rather than a surface.

The following data-name identifiers and definitions are provided for forces and moments and their associated coefficients. For coefficients, the dynamic pressure and length scales used in the normalization are provided.

EXPRESS specification:

```
*)
TYPE force_moment_data_name = EXTENSIBLE ENUMERATION OF
     (force_x,
      force_y,
      force_z,
      force_r,
      force_theta,
      force_phi,
      moment_x,
      moment_y,
      moment_z,
```


Table 12 – Force and moment data name identifiers

Data name identifier	Description	Units
force_x	$F_x = \vec{F} \cdot \hat{e}_x$	$\mathbf{ML/T^2}$
force_y	$F_y = \vec{F} \cdot \hat{e}_y$	$\mathbf{ML/T^2}$
force_z	$F_z = \vec{F} \cdot \hat{e}_z$	$\mathbf{ML/T^2}$
force_r	$F_r = \vec{F} \cdot \hat{e}_r$	$\mathbf{ML/T^2}$
force_theta	$F_\theta = \vec{F} \cdot \hat{e}_\theta$	$\mathbf{ML/T^2}$
force_phi	$F_\phi = \vec{F} \cdot \hat{e}_\phi$	$\mathbf{ML/T^2}$
moment_x	$M_x = \vec{M} \cdot \hat{e}_x$	$\mathbf{ML^2/T^2}$
moment_y	$M_y = \vec{M} \cdot \hat{e}_y$	$\mathbf{ML^2/T^2}$
moment_z	$M_z = \vec{M} \cdot \hat{e}_z$	$\mathbf{ML^2/T^2}$
moment_r	$M_r = \vec{M} \cdot \hat{e}_r$	$\mathbf{ML^2/T^2}$
moment_theta	$M_\theta = \vec{M} \cdot \hat{e}_\theta$	$\mathbf{ML^2/T^2}$
moment_phi	$M_\phi = \vec{M} \cdot \hat{e}_\phi$	$\mathbf{ML^2/T^2}$
moment_xi	$M_\xi = \vec{M} \cdot \hat{e}_\xi$	$\mathbf{ML^2/T^2}$
moment_eta	$M_\eta = \vec{M} \cdot \hat{e}_\eta$	$\mathbf{ML^2/T^2}$
moment_zeta	$M_\zeta = \vec{M} \cdot \hat{e}_\zeta$	$\mathbf{ML^2/T^2}$
moment_center_x	$x_0 = \vec{r}_0 \cdot \hat{e}_x$	**L**
moment_center_y	$y_0 = \vec{r}_0 \cdot \hat{e}_y$	**L**
moment_center_z	$z_0 = \vec{r}_0 \cdot \hat{e}_z$	**L**
lift	L or L'	$\mathbf{ML/T^2}$
drag	D or D'	$\mathbf{ML/T^2}$
coef_lift	C_L or c_l	-
coef_drag	C_D or c_d	-
coef_moment_x	$\vec{C}_M \cdot \hat{e}_x$ or $\vec{c}_m \cdot \hat{e}_x$	-
coef_moment_y	$\vec{C}_M \cdot \hat{e}_y$ or $\vec{c}_m \cdot \hat{e}_y$	-
coef_moment_z	$\vec{C}_M \cdot \hat{e}_z$ or $\vec{c}_m \cdot \hat{e}_z$	-
coef_moment_r	$\vec{C}_M \cdot \hat{e}_r$ or $\vec{c}_m \cdot \hat{e}_r$	-
coef_moment_theta	$\vec{C}_M \cdot \hat{e}_\theta$ or $\vec{c}_m \cdot \hat{e}_\theta$	-
coef_moment_phi	$\vec{C}_M \cdot \hat{e}_\phi$ or $\vec{c}_m \cdot \hat{e}_\phi$	-
coef_moment_xi	$\vec{C}_M \cdot \hat{e}_\xi$ or $\vec{c}_m \cdot \hat{e}_\xi$	-
coef_moment_eta	$\vec{C}_M \cdot \hat{e}_\eta$ or $\vec{c}_m \cdot \hat{e}_\eta$	-
coef_moment_zeta	$\vec{C}_M \cdot \hat{e}_\zeta$ or $\vec{c}_m \cdot \hat{e}_\zeta$	-
coef_pressure_dynamic	$1/2\rho_{\text{ref}} q_{\text{ref}}^2$	$\mathbf{M/(LT^2)}$
coef_area	S_{ref}	$\mathbf{L^2}$
coef_length	c_{ref}	**L**

```
      moment_r,
      moment_theta,
      moment_phi,
      moment_xi,
      moment_eta,
      moment_zeta,
      moment_center_x,
      moment_center_y,
      moment_center_z,
      lift,
      drag,
      coef_lift,
      coef_drag,
      coef_moment_x,
      coef_moment_y,
      coef_moment_z,
      coef_moment_r,
      coef_moment_theta,
      coef_moment_phi,
      coef_moment_xi,
      coef_moment_eta,
      coef_moment_zeta,
      coef_moment_pressure_dynamic,
      coef_moment_area,
      coef_length);
END_TYPE;
(*
```

The meanings of the enumerated items are given in Table 12.

8.3.4 gas_model_data_name

A **gas_model_data_name** is an enumeration of the standard data names that may be associated with data corresponding to different kinds of gaseous state models.

EXPRESS specification:

```
*)
TYPE gas_model_data_name = EXTENSIBLE ENUMERATION OF
     (ideal_gas_constant,
      specific_heat_ratio,
      specific_heat_pressure,
      specific_heat_volume);
END_TYPE;
(*
```

Enumerated item definitions:

ideal_gas_constant: indicates the ideal gas constant;

specific_heat_ratio: indicates the ratio of the specific heats at constant pressure to constant volume;

specific_heat_pressure: indicates the specific heat at constant pressure;

specific_heat_volume: indicates the specific heat at constant volume.

8.3.5 gas_model_type

A **gas_model_type** is an enumeration of the gaseous state models relating pressure, temperature and density.

EXPRESS specification:

```
*)
TYPE gas_model_type = EXTENSIBLE ENUMERATION OF
     (unspecified,
      application_defined,
      ideal,
      Van_der_Waals);
END_TYPE;
(*
```

Enumerated item definitions:

unspecified: is unspecified.

application_defined: is specified via an external agreement between the data creator and the data user;

ideal: the state model is the perfect gas law. The pressure, temperature and density are related by,

$$p = \rho RT,$$

where R is the ideal gas constant. Related quantities are the specific heat at constant pressure (c_p), specific heat at constant volume (c_v) and specific heat ratio ($\gamma = c_p/c_v$). The gas constant and specific heats are related by $R = c_p - c_v$.

The **standard_data_name** identifiers associated with the perfect gas law are: **ideal_gas_constant**, **specific_heat_ratio**, **specific_heat_volume** and **specific_heat_pressure**. These are described in 8.3.4.

Van_der_Waals: the state model is Van der Waals' equation.

8.3.6 mbna_behaviour_models

An **mbna_behaviour_models** is a selection among behavioural models.

EXPRESS specification:

```
*)
TYPE mbna_behaviour_models = EXTENSIBLE SELECT
  (thermal_conductivity_model);
END_TYPE;
(*
```

8.3.7 mbna_governing_equation_type

An **mbna_governing_equation_type** is an enumeration of the classes of equations.

Table 13 – Thermal conductivity model data name identifiers

Data name identifier	Description	Units
power_law_exponent	n	-
temperature_reference	T_{ref}	Θ
thermal_conductivity_reference	k_{ref}	$\mathbf{ML}/(\mathbf{T}^2\Theta)$
Sutherland_constant_conductivity	T_s	Θ
constant_Prandtl	P_r	-

EXPRESS specification:

```
*)
TYPE mbna_governing_equation_type = EXTENSIBLE ENUMERATION OF
     (unspecified,
      application_defined);
END_TYPE;
(*
```

Enumerated item definitions:

unspecified: is unspecified;

application_defined: is specified via an external agreement between the data creator and the data user;

8.3.8 thermal_conductivity_model_data_name

A **thermal_conductivity_model_data_name** is an enumeration of the standard data names that may be associated with data corresponding to different kinds of thermal conductivity models. The identifiers and their units are given in Table 13.

EXPRESS specification:

```
*)
TYPE thermal_conductivity_model_data_name = EXTENSIBLE ENUMERATION OF
     (power_law_exponent,
      temperature_reference,
      thermal_conductivity_reference,
      Sutherland_constant_conductivity,
      constant_Prandtl);
END_TYPE;
(*
```

Enumerated item definitions:

power_law_exponent: indicates a power law exponent;

temperature_reference: indicates a reference temperature;

thermal_conductivity_reference: indicates a reference thermal conductivity;

Sutherland_constant_conductivity: indicates Sutherland's Law constant for thermal conductivity;

thermal_conductivity_molecular_reference: indicates a reference molecular thermal conductivity;

constant_Prandtl: indicates a Prandtl number.

8.3.9 thermal_conductivity_model_type

A **thermal_conductivity_model_type** is an enumeration of the relationships between the thermal-conductivity coefficient and temperature.

EXPRESS specification:

```
*)
TYPE thermal_conductivity_model_type = EXTENSIBLE ENUMERATION OF
     (unspecified,
      application_defined,
      independent,
      power_law,
      Sutherland_law,
      constant_Prandtl);
END_TYPE;
(*
```

Enumerated item definitions:

unspecified: is unspecified;

application_defined: is specified via an external agreement between the data creator and the data user;

independent: identifies that the thermal conductivity coefficient is independent of the temperature and is equal to some reference value ($k = k_{\text{ref}}$).

The standard data name identifier associated with this model is **thermal_conductivity_reference**. This is described in 8.3.8.

power_law: the thermal conductivity is related to temperature via a power-law.

$$k = k_{\text{ref}} \left(\frac{T}{T_{\text{ref}}} \right)^n .$$

The standard data name identifiers associated with this model are: **power_law_exponent**, **temperature_reference** and **thermal_conductivity_reference**. These are described in 8.3.8.

Sutherland_law: Sutherland's Law for thermal conductivity.

$$k = k_{\text{ref}} \left(\frac{T}{T_{\text{ref}}} \right)^{3/2} \frac{T_{\text{ref}} + T_s}{T + T_s},$$

where T_s is the Sutherland Law constant, and k_{ref} and T_{ref} are the reference thermal conductivity and temperature, respectively.

The standard data name identifiers associated with this model are: **Sutherland_constant_conductivity**, **temperature_reference** and **thermal_conductivity_reference**. These are described in 8.3.8.

constant_Prandtl: the Prandtl number ($Pr = \mu c_p/k$) is constant and equal to some reference value.

The standard data name identifier associated with this model is **constant_Prandtl**, and is described in 8.3.8.

NOTE

For air [4], the Prandtl number is $Pr = 0.72$, the power-law exponent is $n = 0.81$, Sutherlands Law constant (T_s) is 194.4 K, the reference temperature (T_{ref}) is 273.15 K, and the reference thermal conductivity (k_{ref}) is 2.414×10^{-2} $kg\ m/(s^3 K)$.

8.3.10 turbulence_closure_data_name

A **turbulence_closure_data_name** is an enumeration of the standard data names that may be associated with data corresponding to different kinds of turbulence enclosures.

EXPRESS specification:

```
*)
TYPE turbulence_closure_data_name = EXTENSIBLE ENUMERATION OF
     (eddy_viscosity,
      Prandtl_turbulent);
END_TYPE;
(*
```

Enumerated item definitions:

eddy_viscosity: indicates an eddy viscosity;

Prandtl_turbulent: indicates a Prandtl turbulent number.

8.3.11 turbulence_closure_type

A **turbulence_closure_type** is an enumeration of the kinds of turbulence closure for the Reynolds stress terms of the Navier-Stokes equations.

EXPRESS specification:

```
*)
TYPE turbulence_closure_type = EXTENSIBLE ENUMERATION OF
     (unspecified,
      application_defined,
      eddy_viscosity,
      Reynolds_stress,
      Reynolds_stress_algebraic);
END_TYPE;
(*
```


Table 14 – Turbulence data name identifiers

Data name identifier	Description	Units
turbulent_distance	distance to nearest wall	**L**
turbulent_energy_kinetic	$k = \frac{1}{2}(\overline{u'u'} + \overline{v'v'} + \overline{w'w'})$	$\mathbf{L}^2/\mathbf{T}^2$
turbulent_dissipation	ϵ	$\mathbf{L}^2/\mathbf{T}^3$
turbulent_dissipation_rate	ϵ/k	$\mathbf{T}^{-1}$
turbulent_BB_Reynolds	Baldwin-Barth one-equation model R_T	-
turbulent_SA_nu_tilde	Spalart-Allmaras one-equation model $\tilde{\nu}$	$\mathbf{L}^2/\mathbf{T}$
turbulent_SA_chi	S-A model $\chi = \tilde{\nu}/\nu$	-
turbulent_SA_cb1	S-A model $c_{b1} = 0.1355$	-
turbulent_SA_cb2	S-A model $c_{b2} = 0.622$	-
turbulent_SA_sigma	S-A model $\sigma = 2/3$	-
turbulent_SA_kappa	S-A model $\kappa = 0.41$ (von Karman constant)	-
turbulent_SA_cw1	S-A model $c_{w1} = 3.2391$	-
turbulent_SA_cw2	S-A model $c_{w2} = 0.3$	-
turbulent_SA_cw3	S-A model $c_{w3} = 2$	-
turbulent_SA_cv1	S-A model $c_{v1} = 7.1$	-
turbulent_SA_ct1	S-A model $c_{t1} = 1$	-
turbulent_SA_ct2	S-A model $c_{t2} = 2$	-
turbulent_SA_ct3	S-A model $c_{t3} = 1.2$	-
turbulent_SA_ct4	S-A model $c_{t4} = 0.5$	-

Enumerated item definitions:

unspecified: is unspecified;

application_defined: is specified via an external agreement between the data creator and the data user;

eddy_viscosity: Boussinesq eddy-velocity closure. The Reynolds stresses are approximated as the product of an eddy viscosity (ν_t) and the mean strain tensor. Using indicial notation, the relation is,

$$-\overline{u_i\prime u_j\prime} = \nu_t \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right)$$

where $-\overline{u_i\prime u_j\prime}$ are the Reynolds stresses.

Reynolds_stress: no approximation of the Reynolds stresses.

Reynolds_stress_algebraic: an algebraic approximation for the Reynolds stresses based on some intermediate transport quantities.

The associated **standard_data_name** name identifiers are: **eddy_viscosity** and **Prandtl_-turbulent**. These are described in 8.3.10.

8.3.12 turbulence_model_data_name

A **turbulence_model_data_name** is an enumeration of standardized Reynolds-averaged Navier-Stokes turbulence model variables.

Turbulence model solution quantities and model constants present a particularly difficult nomenclature problem — to be precise it is necessary to identify both the variable and the model (and version) that it comes from. The list (in Table 14) falls short in this respect.

EXPRESS specification:

```
*)
TYPE turbulence_model_data_name = EXTENSIBLE ENUMERATION OF
     (turbulent_distance,
      turbulent_energy_kinetic,
      turbulent_dissipation,
      turbulent_dissipation_rate,
      turbulent_BB_Reynolds,
      turbulent_SA_nu_tilde,
      turbulent_SA_chi,
      turbulent_SA_cb1,
      turbulent_SA_cb2,
      turbulent_SA_sigma,
      turbulent_SA_kappa,
      turbulent_SA_cw1,
      turbulent_SA_cw2,
      turbulent_SA_cw3,
      turbulent_SA_cv1,
      turbulent_SA_ct1,
      turbulent_SA_ct2,
      turbulent_SA_ct3,
      turbulent_SA_ct4);
END_TYPE;
(*
```

The required identifiers and their meanings are given in Table 14.

8.3.13 turbulence_model_type

A **turbulence_model_type** is an enumeration of the equation sets for modeling the turbulence quantities.

EXPRESS specification:

```
*)
TYPE turbulence_model_type = EXTENSIBLE ENUMERATION OF
     (unspecified,
      application_defined,
      algebraic_Baldwin_Lomax,
      algebraic_Cebeci_Smith,
      half_equation_Johnson_King,
      one_equation_Baldwin_Barth,
      one_equation_Spalart_Allmaras,
      two_equation_Jones_Launder,
      two_equation_Menter_SST,
      two_equation_Wilcox);
END_TYPE;
(*
```

Enumerated item definitions:

unspecified: is unspecified;

application_defined: is specified via an external agreement between the data creator and the data user;

algebraic_Baldwin_Lomax: is Baldwin-Lomax;

algebraic_Cebeci_Smith: is Cebeci-Smith;

half_equation_Johnson_King: is Johnson-King;

one_equation_Baldwin_Barth: is Baldwin-Barth;

one_equation_Spalart_Allmaras: is Spalart-Allmaras;

two_equation_Jones_Launder: is Jones-Launder;

two_equation_Menter_SST: is Menter;

two_equation_Wilcox: is Wilcox.

The associated **standard_data_name** name identifiers for the Spalart-Allmaras turbulence model (version Ia) are: **turbulent_SA_cb1**, **turbulent_SA_cb2**, **turbulent_SA_sigma**, **turbulent_SA_kappa**, **turbulent_SA_cw1**, **turbulent_SA_cw2**, **turbulent_SA_cw3**, **turbulent_SA_cv1**, **turbulent_SA_ct1**, **turbulent_SA_ct2**, **turbulent_SA_ct3**, and **turbulent_SA_ct4**. These are described in 8.3.12.

8.3.14 viscosity_model_data_name

A **viscosity_model_data_name** is an enumeration of the standard data names that may be associated with data corresponding to different kinds of viscosity models.

EXPRESS specification:

```
*)
TYPE viscosity_model_data_name = EXTENSIBLE ENUMERATION OF
     (viscosity_molecular_reference,
      Sutherland_constant_viscosity);
END_TYPE;
(*
```

Enumerated item definitions:

viscosity_molecular_reference: indicates a molecular viscosity reference;

Sutherland_constant_viscosity: indicates Sutherland's constant for Sutherland's Law for molecular viscosity,

8.3.15 viscosity_model_type

A **viscosity_model_type** is an enumeration of the relationships between molecular viscosity and temperature.

EXPRESS specification:

```
*)
```



```
TYPE viscosity_model_type = EXTENSIBLE ENUMERATION OF
     (unspecified,
      application_defined,
      constant_viscosity,
      power_law,
      Sutherland_law);
END_TYPE;
(*
```

Enumerated item definitions:

unspecified: is unspecified;

application_defined: is specified via an external agreement between the data creator and the data user;

constant_viscosity: the molecular viscosity is constant throughout the field and is equal to some reference value ($\mu = \mu_{\text{ref}}$).

The standard data name identifier associated with this model is **viscosity_molecular_reference** and is described in 8.3.14.

power_law: the molecular viscosity follows a power-law relation,

$$\mu = \mu_{\text{ref}} \left(\frac{T}{T_{\text{ref}}} \right)^n .$$

The standard data name identifiers associated with this model are: **power_law_exponent**, **temperature_reference** and **viscosity_molecular_reference**. These are described in 8.3.14 and 8.3.8.

Sutherland_law: Sutherland's Law for molecular viscosity,

$$\mu = \mu_{\text{ref}} \left(\frac{T}{T_{\text{ref}}} \right)^{3/2} \frac{T_{\text{ref}} + T_s}{T + T_s},$$

where T_s is the Sutherland Law constant, and μ_{ref} and T_{ref} are the reference viscosity and temperature, respectively.

The standard data name identifiers associated with this model are: **Sutherland_constant_viscosity**, **temperature_reference** and **viscosity_molecular_reference**. These are described in 8.3.14 and 8.3.8.

NOTE 1

For air [4], the power-law exponent is $n = 0.666$, Sutherlands Law constant (T_s) is 110.6 K, the reference temperature (T_{ref}) is 273.15 K, and the reference viscosity (μ_{ref}) is 1.716×10^{-5} kg/(m s).

8.4 equations_schema entity definitions

8.4.1 fd_diffusion_equation

An **fd_diffusion_equation** is an **fd_governing_equation** which includes diffusion.

Table 15 – Encoding of the 3-D fd_diffusion_model terms

Element	Modelled terms
$n = 1$	diffusion terms in i ($\partial^2/\partial\xi^2$)
$n = 2$	diffusion terms in j ($\partial^2/\partial\eta^2$)
$n = 3$	diffusion terms in k ($\partial^2/\partial\zeta^2$)
$n = 4$	cross-diffusion terms in i-j ($\partial^2/\partial\xi\partial\eta$ and $\partial^2/\partial\eta\partial\xi$)
$n = 5$	cross-diffusion terms in j-k ($\partial^2/\partial\eta\partial\zeta$ and $\partial^2/\partial\zeta\partial\eta$)
$n = 6$	cross-diffusion terms in k-i ($\partial^2/\partial\zeta\partial\xi$ and $\partial^2/\partial\xi\partial\zeta$)

EXPRESS specification:

```
*)
ENTITY fd_diffusion_equation
  SUBTYPE OF (fd_governing_equation);
  diffusion_model : fd_diffusion_model;
END_ENTITY;
(*
```

Attribute definitions:

diffusion_model: describes the viscous diffusion terms modelled in the flow equations, and is applicable only to Navier-Stokes equations.

8.4.2 fd_diffusion_model

An **fd_diffusion_model** is the viscous diffusion terms modelled in the flow equations, and is applicable only to Navier-Stokes equations.

Typically, thin-layer approximations include only the diffusion terms in one or two computational-coordinate directions. **fd_diffusion_model** encodes the coordinate directions that include second-derivative and cross-derivative diffusion terms. The first N elements, where N is the computational dimension, are second-derivative terms and the remainder elements are cross-derivative terms. A value of TRUE indicates the diffusion term is modelled, and FALSE indicates that it is not modelled. In 3–D, the encoding of the **fd_diffusion_model** terms is given in Table 15, where derivatives in the i, j and k computational-coordinates are ξ, η and ζ, respectively.

EXAMPLE The full Navier-Stokes equations in 3–D are indicated by:
terms = `[TRUE,TRUE,TRUE,TRUE,TRUE,TRUE]`
while the thin-layer equations including only diffusion in the j-direction are indicated by:
terms = `[FALSE,TRUE,FALSE,FALSE,FALSE,FALSE]`.

EXPRESS specification:

```
*)
ENTITY fd_diffusion_model;
  terms : ARRAY [1:diff] OF BOOLEAN;
  diff  : INTEGER;
END_ENTITY;
```



```
(*
```

Attribute definitions:

terms: the diffusion terms;

diff: the number of elements in the **terms** array. For 1-D this is one, for 2-D it is three, and for 3-D it is six.

8.4.3 fd_governing_equation

An **fd_governing_equation** is an **mbna_governing_equation** describing the class of governing flow equations associated with the solution.

EXPRESS specification:

```
*)
ENTITY fd_governing_equation
  SUBTYPE OF (mbna_governing_equation);
  SELF\mbna_governing_equation.equation_type
                  : fd_governing_equation_type;
END_ENTITY;
(*
```

Attribute definitions:

equation_type: the kind of equation.

8.4.4 flow_equation_set

A **flow_equation_set** is a **mbna_equation_set** providing a general description of governing flow equations. It includes the dimensionality of the governing equations.

EXPRESS specification:

```
*)
ENTITY flow_equation_set
  SUBTYPE OF (mbna_equation_set);
  SELF\mbna_equation_set.equations : fd_governing_equation;
  fd_models                        : SET OF fd_behaviour_models;
END_ENTITY;
(*
```

Attribute definitions:

equations: describes the general class of CFD equations;

fd_models: describes zero or more of:

— the gaseous equation of state;

— the auxiliary relations for molecular viscosity;

— the turbulent closure for Reynolds-averaged Navier-Stokes equations;

— the turbulence model for Reynolds-averaged Navier-Stokes equations.

8.4.5 gas_model

A **gas_model** is the equation of state model used in the governing equations to relate pressure, temperature and density.

EXPRESS specification:

```
*)
ENTITY gas_model
  SUBTYPE OF (mbna_behaviour_model);
  model_type : gas_model_type;
END_ENTITY;
(*
```

Attribute definitions:

model_type: the particular gaseous equation of state model.

8.4.6 mbna_behaviour_model

An **mbna_behaviour_model** is a model for relating physical or mathematical quantities.

EXPRESS specification:

```
*)
ENTITY mbna_behaviour_model
  SUBTYPE OF (mbna_equation);
INVERSE
  data : SET OF model_property_distribution FOR domain;
END_ENTITY;

SUBTYPE_CONSTRAINT sc1_mbna_behaviour_model FOR mbna_behaviour_model;
  ABSTRACT SUPERTYPE;
  ONEOF(thermal_conductivity_model,
        gas_model,
        turbulence_closure,
        turbulence_model,
        viscosity_model);
END_SUBTYPE_CONSTRAINT;
(*
```

Attribute definitions:

data: the data.

8.4.7 mbna_equation

A **mbna_equation** represents the concept of a mathematical formulation of a physics phenonema.

EXPRESS specification:

```
*)
ENTITY mbna_equation
  SUBTYPE OF (model_product_domain);
END_ENTITY;

SUBTYPE_CONSTRAINT sc1_mbna_equation FOR mbna_equation;
  ONEOF(mbna_equation_set,
        mbna_governing_equation,
        mbna_behaviour_model);
END_SUBTYPE_CONSTRAINT;
(*
```

Attribute definitions:

description: (inherited) annotation;

name: (inherited) user-specified instance identifier;

id: (inherited) an identifier.

8.4.8 mbna_equation_set

An **mbna_equation_set** is a **mbna_equation** that gives a general description of governing equations. It includes the dimensionality of the governing equations.

EXPRESS specification:

```
*)
ENTITY mbna_equation_set
  SUBTYPE OF (mbna_equation);
  dimension   : INTEGER;
  equations   : mbna_governing_equation;
  mbna_models : SET OF mbna_behaviour_models;
END_ENTITY;
(*
```

Attribute definitions:

dimension: the dimensionality of the governing equations; it is the number of spatial variables describing the equation;

equations: describes the general class of equations.

mbna_models: zero or more behavioural models;

8.4.9 mbna_governing_equation

An **mbna_governing_equation** is an **mbna_equation** describing the class of governing equations associated with the analysis solution.

EXPRESS specification:

```
*)
ENTITY mbna_governing_equation
  SUBTYPE OF (mbna_equation);
  equation_type : mbna_governing_equation_type;
END_ENTITY;
(*
```

Attribute definitions:

equation_type: the kind of equation.

8.4.10 thermal_conductivity_model

A **thermal_conductivity_model** is the model for relating the thermal-conductivity coefficient (k) to temperature.

EXPRESS specification:

```
*)
ENTITY thermal_conductivity_model
  SUBTYPE OF (mbna_behaviour_model);
  model_type : thermal_conductivity_model_type;
END_ENTITY;
(*
```

Attribute definitions:

model_type: the particular thermal conductivity model type.

8.4.11 turbulence_closure

A **turbulence_closure** is the turbulence closure for the Reynolds stress terms of the Navier-Stokes equations.

EXPRESS specification:

```
*)
ENTITY turbulence_closure
  SUBTYPE OF (mbna_behaviour_model);
  closure_type : turbulence_closure_type;
END_ENTITY;
```



```
(*
```

Attribute definitions:

closure_type: the particular turbulence closure type.

8.4.12 turbulence_model

A **turbulence_model** is the equation set used to model the turbulence quantities.

EXPRESS specification:

```
*)
ENTITY turbulence_model
  SUBTYPE OF (mbna_behaviour_model);
  model_type      : turbulence_model_type;
  diffusion_model : OPTIONAL fd_diffusion_model;
END_ENTITY;
(*
```

Attribute definitions:

model_type: the particular turbulence model type;

diffusion_model: the description of the viscous diffusion terms included in the turbulent transport model equations.

8.4.13 viscosity_model

A **viscosity_model** is the model for relating molecular viscosity (μ) to temperature.

EXPRESS specification:

```
*)
ENTITY viscosity_model
  SUBTYPE OF (mbna_behaviour_model);
  model_type : viscosity_model_type;
END_ENTITY;
(*
```

Attribute definitions:

model_type: the particular viscosity model type.

EXPRESS specification:

```
*)
```



```
END_SCHEMA; -- end of equations_schema
(*
```

9 Results

The following EXPRESS declaration begins the **results_schema** and identifies the necessary external references.

EXPRESS specification:

```
*)
SCHEMA results_schema;
  REFERENCE FROM analysis_schema           -- ISO 10303-53
                (model_property_distribution);
  REFERENCE FROM mathematical_description_of_distribution_schema -- ISO 10303-51
                (property_distribution_description);
  REFERENCE FROM mesh_topology_schema      -- ISO 10303-52
                (mesh_location,
                 rind);
  REFERENCE FROM support_resource_schema   -- ISO 10303-41
                (text);
(*
```

NOTE The schemas referenced above can be found in the following parts of ISO 10303:

analysis_schema	part 53
mathematical_description_of_distribution_schema	part 51
mesh_topology_schema	part 52
support_resource_schema	part 41

Abbreviated names are used in identifiers of elements declared in this schema. Prefixes used in these identifiers have the following meanings:

fd fluid_dynamics;

mbna mesh_based_numerical_analysis.

9.1 Introduction

This schema defines and describes solution data and other data resulting from a mesh-based numerical analysis.

9.2 Fundamental concepts and assumptions

The principal result of an analysis is the solution data over the computational grid.

Other data, such as equation residuals, can also form part of the results.

9.3 results_schema type definitions

9.3.1 flow_solution_data_name

A **flow_solution_data_name** is an enumeration of standardized flow solution data.

This clause describes data-name identifiers for typical Navier-Stokes solution variables. The list is obviously incomplete, but should suffice for initial implementation of the CFD system. The variables listed in this section are dimensional or raw quantities; nondimensional parameters and coefficients based on these variables are discussed in 4.3.7.

A reasonably universal notation is used for state variables. Static quantities are measured with the fluid at speed: static density (ρ), static pressure (p), static temperature (T), static internal energy per unit mass (e), static enthalpy per unit mass (h), entropy (s), and static speed of sound (c). The true entropy isaa approximated by the function $\tilde{s} = p/\rho^{\gamma}$ (this assumes an ideal gas). The velocity is $\vec{q} = u\hat{e}_x + v\hat{e}_y + w\hat{e}_z$, with magnitude $q = \sqrt{\vec{q}\cdot\vec{q}}$. Stagnation quantities are obtained by bringing the fluid isentropically to rest; these are identified by a subscript '$_0$'. The term 'total' is also used to refer to stagnation quantities.

Conservation variables are density, momentum ($\rho\vec{q} = \rho u\hat{e}_x + \rho v\hat{e}_y + \rho w\hat{e}_z$), and stagnation energy per unit volume (ρe_0).

Molecular diffusion and heat transfer introduce the molecular viscosity (μ), kinematic viscosity (ν) and thermal conductivity coefficient (k). These are obtained from the state variables through auxiliary correlations. For a perfect gas, μ and k are functions of static temperature only.

The Navier-Stokes equations involve the strain tensor ($\bar{\bar{S}}$) and the shear-stress tensor ($\bar{\bar{\tau}}$). Using indicial notation, the 3–D cartesian components of the strain tensor are,

$$\bar{\bar{S}}_{i,j} = \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right),$$

and the stress tensor is,

$$\bar{\bar{\tau}}_{i,j} = \mu \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) + \lambda \frac{\partial u_k}{\partial x_k},$$

where $(x_1, x_2, x_3) = (x, y, z)$ and $(u_1, u_2, u_3) = (u, v, w)$. The bulk viscosity is usually approximated as $\lambda = -2/3\mu$.

Reynolds averaging of the Navier-Stokes equations introduce Reynolds stresses ($-\rho\overline{u'v'}$, etc.) and turbulent heat flux terms ($-\rho\overline{u'e'}$, etc.), where primed quantities are instantaneous fluctuations and the bar is an averaging operator. These quantities are obtained from auxiliary turbulence closure models. Reynolds-stress models formulate transport equations for the Reynolds stresses directly; whereas, eddy-viscosity models correlate the Reynolds stresses with the mean strain rate,

$$-\overline{u'v'} = \nu_t \left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} \right),$$

where ν_t is the eddy viscosity. The eddy viscosity is either correlated to mean flow quantities by algebraic models or by auxiliary transport models. An example two-equation turbulence transport model is the k-ϵ model, where transport equations are formulated for the turbulent kinetic energy ($k = \frac{1}{2}(\overline{u'u'} + \overline{v'v'} + \overline{w'w'})$) and turbulent dissipation (ϵ).

Skin friction evaluated at a surface is the dot product of the shear stress tensor with the surface normal:

$$\vec{\tau} = \bar{\bar{\tau}}\cdot\hat{n},$$

Note that skin friction is a vector.

The following data-name identifiers are defined for flow-solution quantities.

Table 16 – Flow solution data name identifiers

Data name identifier	Description	Units
potential	potential: $\nabla\phi = \vec{q}$	$\mathbf{L}^2/\mathbf{T}$
stream_function	stream function (2–D): $\nabla\times\psi = \vec{q}$	$\mathbf{L}^2/\mathbf{T}$
density	static density (ρ)	$\mathbf{M}/\mathbf{L}^3$
pressure	static pressure (p)	$\mathbf{M}/(\mathbf{L}\mathbf{T}^2)$
temperature	static temperature (T)	Θ
energy_internal	static internal energy per unit mass (e)	$\mathbf{L}^2/\mathbf{T}^2$
enthalpy	static enthalpy per unit mass (h)	$\mathbf{L}^2/\mathbf{T}^2$
entropy	entropy (s)	$\mathbf{M}\mathbf{L}^2/(\mathbf{T}^2\Theta)$
entropy_approx	approximate entropy ($\tilde{s} = p/\rho^\gamma$)	$\mathbf{L}^{3\gamma-1}/(\mathbf{M}^{\gamma-1}\mathbf{T}^2)$
density_stagnation	stagnation density (ρ_0)	$\mathbf{M}/\mathbf{L}^3$
pressure_stagnation	stagnation pressure (p_0)	$\mathbf{M}/(\mathbf{L}\mathbf{T}^2)$
temperature_stagnation	stagnation temperature (T_0)	Θ
energy_stagnation	stagnation energy per unit mass (e_0)	$\mathbf{L}^2/\mathbf{T}^2$
enthalpy_stagnation	stagnation enthalpy per unit mass (h_0)	$\mathbf{L}^2/\mathbf{T}^2$
energy_stagnation_density	stagnation energy per unit volume (ρe_0)	$\mathbf{M}/(\mathbf{L}\mathbf{T}^2)$
velocity_x	x-component of velocity ($u = \vec{q}\cdot\hat{e}_x$)	$\mathbf{L}/\mathbf{T}$
velocity_y	y-component of velocity ($v = \vec{q}\cdot\hat{e}_y$)	$\mathbf{L}/\mathbf{T}$
velocity_z	z-component of velocity ($w = \vec{q}\cdot\hat{e}_z$)	$\mathbf{L}/\mathbf{T}$
velocity_r	radial velocity component ($\vec{q}\cdot\hat{e}_r$)	$\mathbf{L}/\mathbf{T}$
velocity_theta	velocity component in θ direction ($\vec{q}\cdot\hat{e}_\theta$)	$\mathbf{L}/\mathbf{T}$
velocity_phi	velocity component in ϕ direction ($\vec{q}\cdot\hat{e}_\phi$)	$\mathbf{L}/\mathbf{T}$
velocity_magnitude	velocity magnitude ($q = \sqrt{\vec{q}\cdot\vec{q}}$)	$\mathbf{L}/\mathbf{T}$
velocity_normal	normal velocity component ($\vec{q}\cdot\hat{n}$)	$\mathbf{L}/\mathbf{T}$
velocity_tangential	tangential velocity component (2–D)	$\mathbf{L}/\mathbf{T}$
velocity_sound	static speed of sound	$\mathbf{L}/\mathbf{T}$
velocity_sound_stagnation	stagnation speed of sound	$\mathbf{L}/\mathbf{T}$
momentum_x	x-component of momentum (ρu)	$\mathbf{M}/(\mathbf{L}^2\mathbf{T})$
momentum_y	y-component of momentum (ρv)	$\mathbf{M}/(\mathbf{L}^2\mathbf{T})$
momentum_z	z-component of momentum (ρw)	$\mathbf{M}/(\mathbf{L}^2\mathbf{T})$
momentum_magnitude	magnitude of momentum (ρq)	$\mathbf{M}/(\mathbf{L}^2\mathbf{T})$
energy_kinetic	$\frac{1}{2}(u^2+v^2+w^2) = \frac{1}{2}q^2$	$\mathbf{L}^2/\mathbf{T}^2$
pressure_dynamic	$\frac{1}{2}\rho q^2$	$\mathbf{M}/(\mathbf{L}\mathbf{T}^2)$
vorticity_x	$\omega_x = \partial w/\partial y - \partial v/\partial z = \vec{\omega}\cdot\hat{e}_x$	$\mathbf{T}^{-1}$
vorticity_y	$\omega_y = \partial u/\partial z - \partial w/\partial x = \vec{\omega}\cdot\hat{e}_y$	$\mathbf{T}^{-1}$
vorticity_z	$\omega_z = \partial v/\partial x - \partial u/\partial y = \vec{\omega}\cdot\hat{e}_z$	$\mathbf{T}^{-1}$

Continued on next page

Table 16 — concluded from previous page

Data name identifier	Description	Units
vorticity_magnitude	$\omega = \sqrt{\vec{\omega} \cdot \vec{\omega}}$	$\mathbf{T}^{-1}$
skin_friction_x	x-component of skin friction ($\vec{\tau} \cdot \hat{e}_x$)	$\mathbf{M}/(\mathbf{LT}^2)$
skin_friction_y	y-component of skin friction ($\vec{\tau} \cdot \hat{e}_y$)	$\mathbf{M}/(\mathbf{LT}^2)$
skin_friction_z	z-component of skin friction ($\vec{\tau} \cdot \hat{e}_z$)	$\mathbf{M}/(\mathbf{LT}^2)$
skin_friction_magnitude	skin friction magnitude ($\sqrt{\vec{\tau} \cdot \vec{\tau}}$)	$\mathbf{M}/(\mathbf{LT}^2)$
velocity_angle_x	velocity angle ($\arccos(u/q) \in [0, 180°)$)	α
velocity_angle_y	$\arccos(v/q)$	α
velocity_angle_z	$\arccos(w/q)$	α
velocity_unit_vector_x	x-component of velocity unit vector ($(\vec{q} \cdot \hat{e}_x)/q$)	-
velocity_unit_vector_y	y-component of velocity unit vector ($(\vec{q} \cdot \hat{e}_y)/q$)	-
velocity_unit_vector_z	z-component of velocity unit vector ($(\vec{q} \cdot \hat{e}_z)/q$)	-
mass_flow	mass flow normal to a plane ($\rho \vec{q} \cdot \hat{n}$)	$\mathbf{M}/(\mathbf{L}^2\mathbf{T})$
viscosity_kinematic	kinematic viscosity ($\nu = \mu/\rho$)	$\mathbf{L}^2/\mathbf{T}$
viscosity_molecular	molecular viscosity (μ)	$\mathbf{M}/(\mathbf{LT})$
viscosity_eddy	eddy viscosity (ν_t)	$\mathbf{L}^2/\mathbf{T}$
thermal_conductivity	thermal conductivity coefficient (k)	$\mathbf{ML}/(\mathbf{T}^3\Theta)$
ideal_gas_constant	ideal gas constant ($R = c_p - c_v$)	$\mathbf{L}/(\mathbf{T}^2\Theta)$
specific_heat_pressure	specific heat at constant pressure (c_p)	$\mathbf{L}^2/(\mathbf{T}^2\Theta)$
specific_heat_volume	specific heat at constant volume (c_v)	$\mathbf{L}^2/(\mathbf{T}^2\Theta)$
Reynolds_stress_xx	Reynolds stress $-\rho\overline{u'u'}$	$\mathbf{M}/(\mathbf{LT}^2)$
Reynolds_stress_xy	Reynolds stress $-\rho\overline{u'v'}$	$\mathbf{M}/(\mathbf{LT}^2)$
Reynolds_stress_xz	Reynolds stress $-\rho\overline{u'w'}$	$\mathbf{M}/(\mathbf{LT}^2)$
Reynolds_stress_yy	Reynolds stress $-\rho\overline{v'v'}$	$\mathbf{M}/(\mathbf{LT}^2)$
Reynolds_stress_yz	Reynolds stress $-\rho\overline{v'w'}$	$\mathbf{M}/(\mathbf{LT}^2)$
Reynolds_stress_zz	Reynolds stress $-\rho\overline{w'w'}$	$\mathbf{M}/(\mathbf{LT}^2)$

EXPRESS specification:

```
*)
TYPE flow_solution_data_name = EXTENSIBLE ENUMERATION OF
     (potential,
      stream_function,
      density,
      pressure,
      temperature,
      energy_internal,
      enthalpy,
      entropy,
      entropy_approx,
      density_stagnation,
      pressure_stagnation,
      temperature_stagnation,
      energy_stagnation,
      enthalpy_stagnation,
      energy_stagnation_density,
      velocity_x,
      velocity_y,
      velocity_z,
      velocity_r,
      velocity_theta,
      velocity_phi,
      velocity_magnitude,
```



```
      velocity_normal,
      velocity_tangential,
      velocity_sound,
      velocity_sound_stagnation,
      momentum_x,
      momentum_y,
      momentum_z,
      momentum_magnitude,
      energy_kinetic,
      pressure_dynamic,
      vorticity_x,
      vorticity_y,
      vorticity_z,
      vorticity_magnitude,
      skin_friction_x,
      skin_friction_y,
      skin_friction_z,
      skin_friction_magnitude,
      velocity_angle_x,
      velocity_angle_y,
      velocity_angle_z,
      velocity_unit_vector_x,
      velocity_unit_vector_y,
      velocity_unit_vector_z,
      mass_flow,
      viscosity_kinematic,
      viscosity_molecular,
      viscosity_eddy,
      thermal_conductivity,
      ideal_gas_constant,
      specific_heat_pressure,
      specific_heat_volume,
      Reynolds_stress_xx,
      Reynolds_stress_xy,
      Reynolds_stress_xz,
      Reynolds_stress_yy,
      Reynolds_stress_yz,
      Reynolds_stress_zz);
END_TYPE;
(*
```

The meanings of the identifiers are given in Table 16.

9.4 results_schema entity definitions

9.4.1 mbna_discrete_data

An **mbna_discrete_data** is generic discrete data (i.e., data defined on a computational grid). This structure can be used for field data, such as fluxes or equation residuals, that is not typically considered part of the solution.

EXPRESS specification:

```
*)
ENTITY mbna_discrete_data
  SUBTYPE OF (mbna_result);
```



```
  gridloc : mesh_location;
INVERSE
  data : SET OF property_distribution_description FOR physical_function;
END_ENTITY;
(*
```

Attribute definitions:

gridloc: the location of the data with respect to the grid. All data within a given instance of **mbna_discrete_data** resides at the same kind of grid location;

data: the data.

9.4.2 mbna_discrete_data_with_rind

An **mbna_discrete_data_with_rind** is an **mbna_discrete_data** with rind point data.

EXPRESS specification:

```
*)
ENTITY mbna_discrete_data_with_rind
  SUBTYPE OF (mbna_discrete_data);
  rind_planes : rind;
END_ENTITY;
(*
```

Attribute definitions:

rind_planes: the rind planes included in the data.

9.4.3 mbna_history

An **mbna_history** is solver convergence history information.

Measures used to record convergence vary greatly among current solver implementations. Convergence information typically includes global forces, norms of equation residuals, and norms of solution changes.

NOTE Attempts to systematically define a set of convergence measures have been futile. For global parameters, such as forces and moments (Table 12), a set of standarized data-array identifiers can be agreed. For equation residuals and solution changes, no such standard list exists. Therefore, either the **id** attribute of the **general_property** associated with the **data_array**s, or an **externally_defined_item** value for the **specified_name** of an associated **specified_general_property** has to be used as the data-array identifier. It is suggested that identifiers for norms of equation residuals begin with RSD, and those for solution changes begin with CHG. For example, **'RSD Mass RMS'** could be used for the L_2-norm (RMS) of mass conservation residuals.

EXPRESS specification:

```
*)
```



```
ENTITY mbna_history
  SUBTYPE OF (mbna_result);
  notes : LIST OF text;
INVERSE
  data : SET OF property_distribution_description FOR physical_function;
END_ENTITY;
(*
```

Attribute definitions:

notes: description of the information recorded as **data**;

data: history data.

9.4.4 mbna_result

An **mbna_result** represents the concept of a solution to an analysis problem and/or other data resulting from an analysis.

EXPRESS specification:

```
*)
ENTITY mbna_result
  SUBTYPE OF (model_property_distribution);
END_ENTITY;

SUBTYPE_CONSTRAINT sc1_mbna_result FOR mbna_result;
  ONEOF(mbna_solution,
        mbna_history,
        mbna_discrete_data);
END_SUBTYPE_CONSTRAINT;
(*
```

9.4.5 mbna_solution

An **mbna_solution** is the solution within a zone.

EXPRESS specification:

```
*)
ENTITY mbna_solution
  SUBTYPE OF (mbna_result);
  gridloc  : mesh_location;
INVERSE
  solution : SET OF property_distribution_description FOR physical_function;
END_ENTITY;
(*
```


Attribute definitions:

gridloc: specifies the location of the solution data with respect to the grid. All data within a given instance of **mbna_solution** resides at the same kind of grid location;

solution: the solution data.

9.4.6 mbna_solution_with_rind

An **mbna_solution_with_rind** is an **mbna_solution** with rind point data.

EXPRESS specification:

```
*)
ENTITY mbna_solution_with_rind
  SUBTYPE OF (mbna_solution);
  rind_planes : rind;
END_ENTITY;
(*
```

Attribute definitions:

rind_planes: the rind planes included in the data.

EXPRESS specification:

```
*)
END_SCHEMA; -- end of results_schema
(*
```


Annex A
(normative)
Short names of entities

Table A.1 provides the short names of entities specified in this part of ISO 10303. Requirements on the use of short names are found in the implementation methods included in ISO 10303.

NOTE The short names are available from the Internet — see annex C.

Table A.1 – Short names of entities

Entity data types names	Short names
elements_bc	ELMBC
fd_bc	FDBC
fd_bc_dataset	FDBCDT
fd_diffusion_equation	FDDFEQ
fd_diffusion_model	FDDFMD
fd_governing_equation	FDGVEQ
fd_step	FDSTP
fd_zone	FDZN
fd_zone_bc	FDZNBC
flow_equation_set	FLEQST
gas_model	GSMDL
geometry_reference	GMTRFR
grid_coordinates	GRDCRD
grid_coordinates_with_rind	GCWR
indexed_elements_bc	INELBC
indexed_points_bc	INPNBC
mbna_bc	MBNBC
mbna_bc_data	MBBCDT
mbna_bc_data_global	MBDG
mbna_bc_data_local	MBDL
mbna_bc_dataset	MBB0
mbna_behaviour_model	MBBHMD
mbna_condition	MBNCND
mbna_Dirichlet_bc_dataset	MDBD
mbna_discrete_data	MBDSDT
mbna_discrete_data_with_rind	MDDWR
mbna_equation	MBNEQT
mbna_equation_set	MBEQST
mbna_family	MBNFML
mbna_governing_equation	MBGVEQT
mbna_history	MBNHST
mbna_integral_data	MBINDT
mbna_model	MBNMDL
mbna_Neumann_bc_dataset	MNBD
mbna_reference_state	MBRFST
mbna_result	MBNRSL
mbna_solution	MBNSLT
mbna_solution_with_rind	MSWR
mbna_step	MBNSTP

Table A.1 – (concluded)

Entity data types names	Short names
mbna_structured_zone	MBSTZN
mbna_unstructured_zone	MBUNZN
mbna_zone	MBNZN
mbna_zone_bc	MBZNBC
specified_general_property	SGPNPR
specified_representation_context	SPRPCN
thermal_conductivity_model	THCNMD
turbulence_closure	TRBCLS
turbulence_model	TRBMDL
viscosity_model	VSCMDL

Annex B
(normative)
Information object registration

B.1 Document identification

To provide for unambiguous identification of an information object in an open system, the object identifier

{ iso standard 10303 part(110) version(-1) }

is assigned to this part of ISO 10303. The meaning of this value is defined in ISO/IEC 8824-1, and is described in ISO 10303-1.

B.2 Schema identification

To provide for unambiguous identification of the schema-name in an open information system, the object identifier

{ iso standard 10303 part(110) version(1) schema(1) basis-schema(1) }

is assigned to the **basis_schema** schema (see 4). The meaning of this value is defined in ISO/IEC 8824-1, and is described in ISO 10303-1.

To provide for unambiguous identification of the schema-name in an open information system, the object identifier

{ iso standard 10303 part(110) version(1) schema(1) hierarchy-schema(1) }

is assigned to the **hierarchy_schema** schema (see 5). The meaning of this value is defined in ISO/IEC 8824-1, and is described in ISO 10303-1.

To provide for unambiguous identification of the schema-name in an open information system, the object identifier

{ iso standard 10303 part(110) version(1) schema(1) domain-schema(1) }

is assigned to the **domain_schema** schema (see 6). The meaning of this value is defined in ISO/IEC 8824-1, and is described in ISO 10303-1.

To provide for unambiguous identification of the schema-name in an open information system, the object identifier

{ iso standard 10303 part(110) version(1) schema(1) conditions-schema(1) }

is assigned to the **conditions_schema** schema (see 7). The meaning of this value is defined in ISO/IEC 8824-1, and is described in ISO 10303-1.

To provide for unambiguous identification of the schema-name in an open information system, the object identifier

{ iso standard 10303 part(110) version(1) schema(1) equations-schema(1) }

is assigned to the **equations_schema** schema (see 8). The meaning of this value is defined in ISO/IEC 8824-1, and is described in ISO 10303-1.

To provide for unambiguous identification of the schema-name in an open information system, the object identifier

{ iso standard 10303 part(110) version(1) schema(1) results-schema(1) }

is assigned to the **results_schema** schema (see 9). The meaning of this value is defined in ISO/IEC 8824-1, and is described in ISO 10303-1.

Annex C
(informative)
EXPRESS listing

This annex references a listing of the EXPRESS entity data type names and corresponding short names as specified in this part of ISO 10303. It also references a listing of each EXPRESS schema specified in this part of ISO 10303, without comments or other explanatory text. These listings are available in computer-interpretable form and can be found at the following URLs:

Short names: `<http://www.mel.nist.gov/div826/subject/apde/snr/>`
EXPRESS: `<http://www.mel.nist.gov/step/parts/part110/cd/>`

If there is difficulty accessing these sites contact ISO Central Secretariat or contact the ISO TC 184/SC4 Secretariat directly at: `sc4sec@cme.nist.gov`.

NOTE The information provided in computer-interpretable form at the above URLs is informative. The information that is contained in the body of this part of ISO 10303 is normative.

Annex D
(informative)
EXPRESS-G diagrams

The diagrams in this annex correspond to the EXPRESS schemas specified in this part of ISO 10303. The diagrams use the EXPRESS-G graphical notation for the EXPRESS language. EXPRESS-G is defined in annex D of ISO 10303-11.

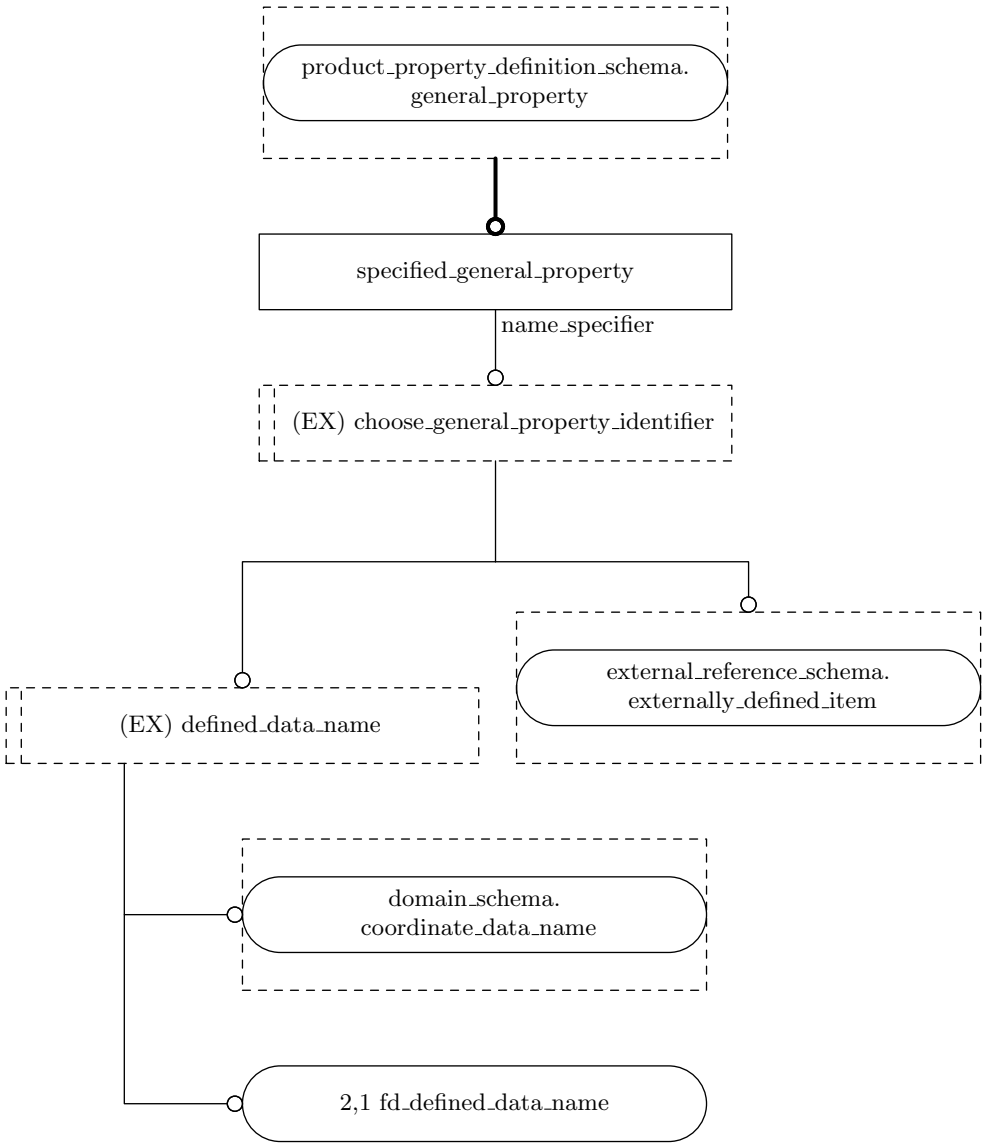

Figure D.1 – Entity level diagram of basis schema (page 1 of 5)

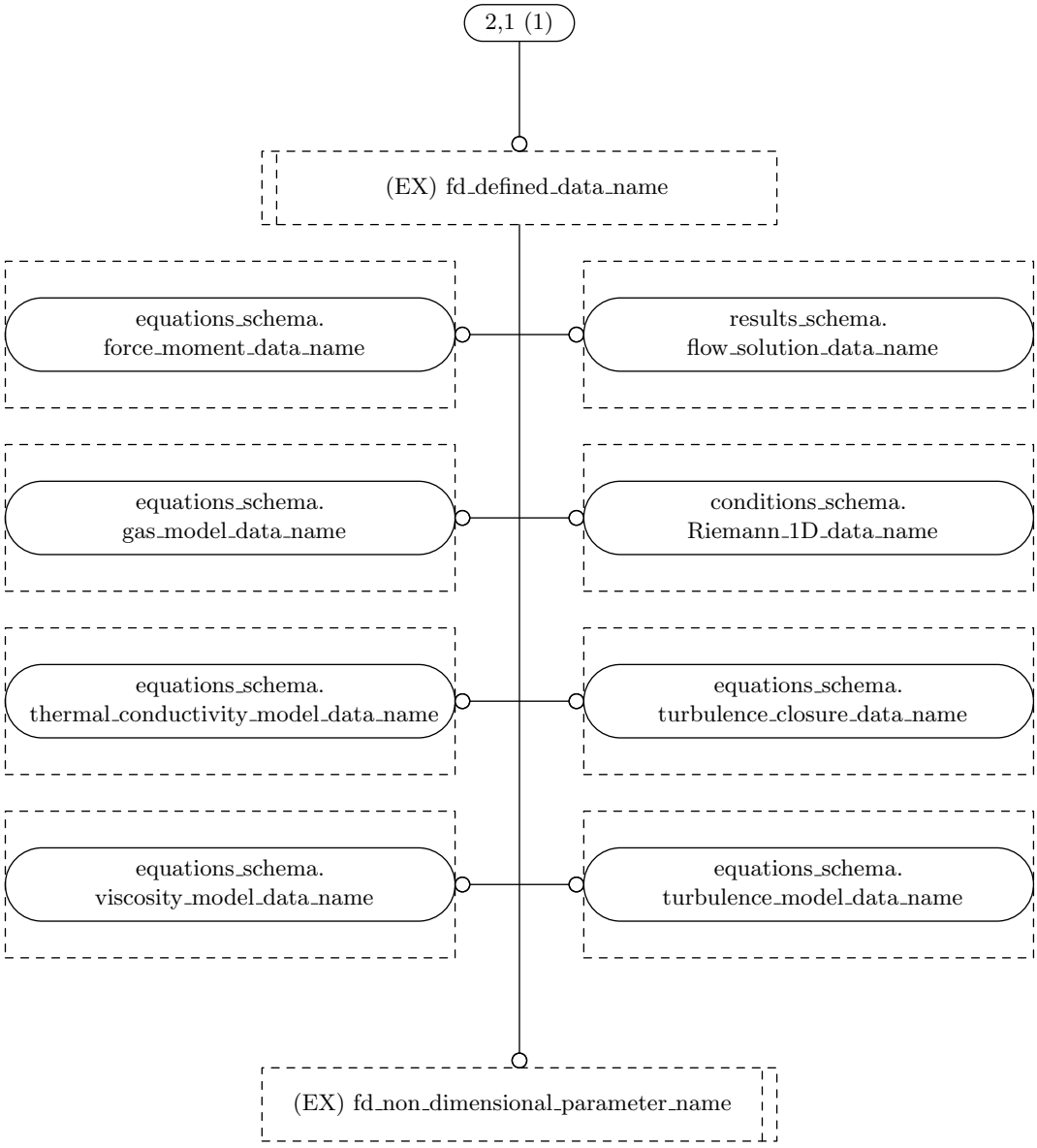

Figure D.2 – Entity level diagram of basis schema (page 2 of 5)

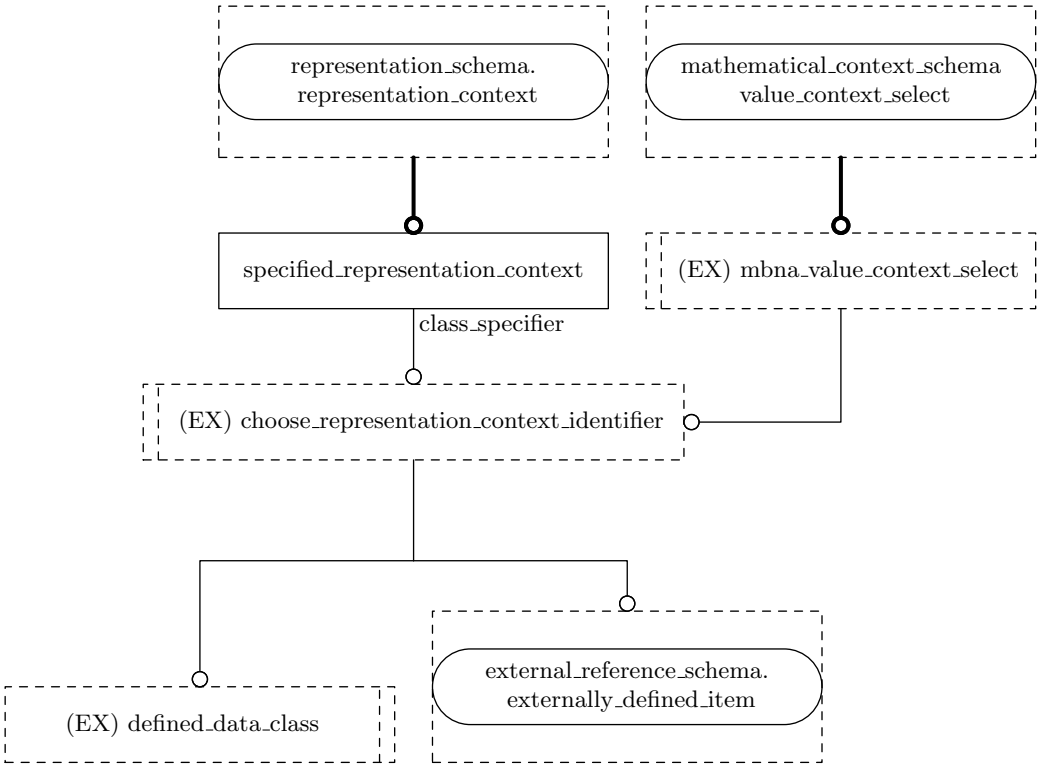

Figure D.3 – Entity level diagram of basis schema (page 3 of 5)

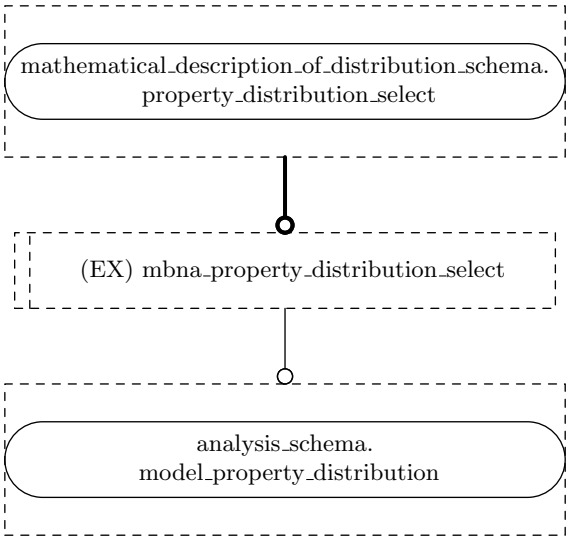

Figure D.4 – Entity level diagram of basis schema (page 4 of 5)

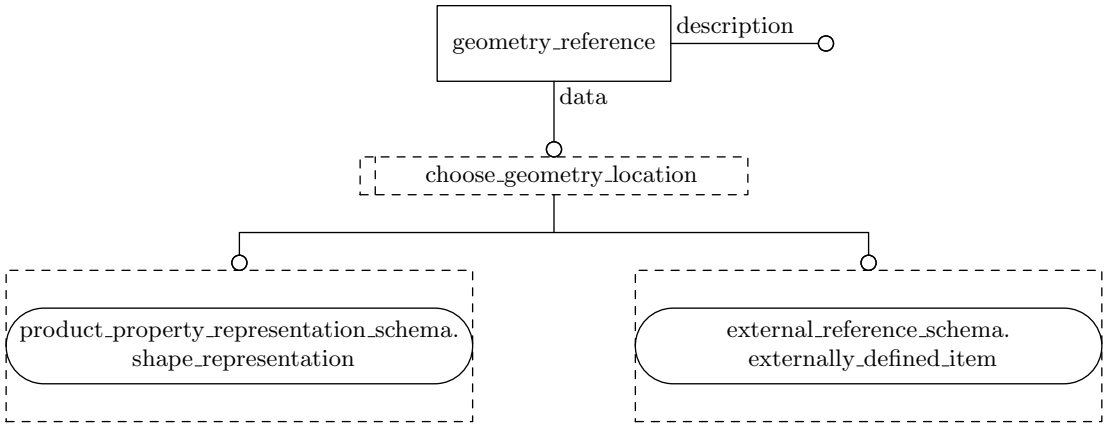

Figure D.5 – Entity level diagram of basis schema (page 5 of 5)

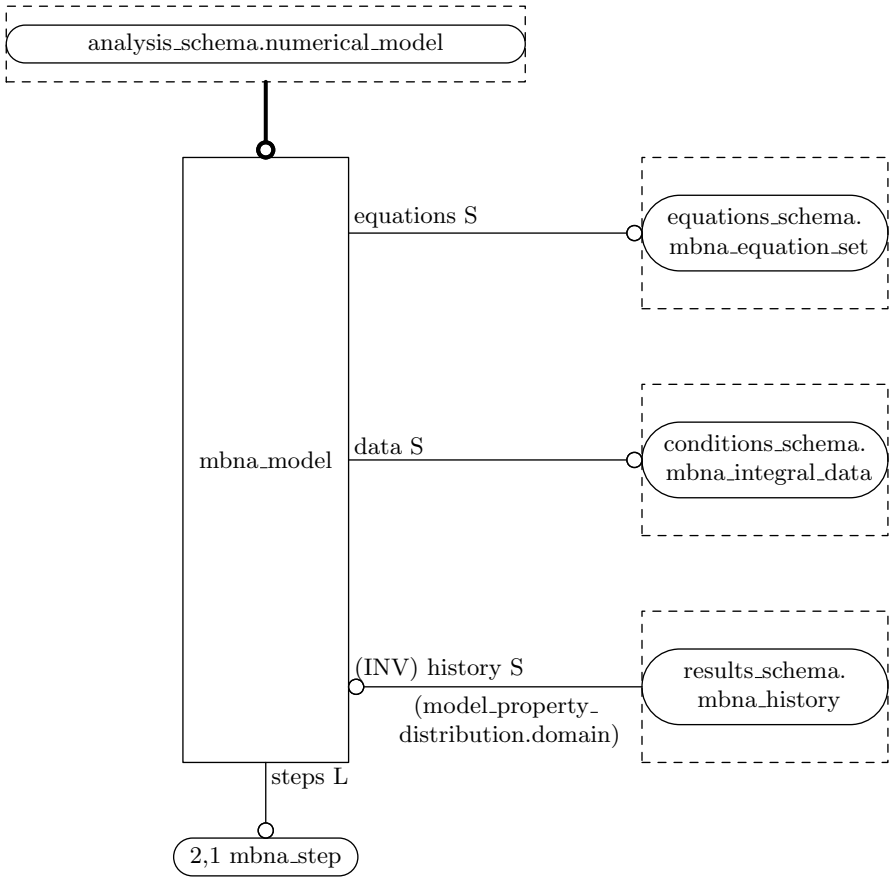

Figure D.6 – Entity level diagram of hierarchy schema (page 1 of 6)

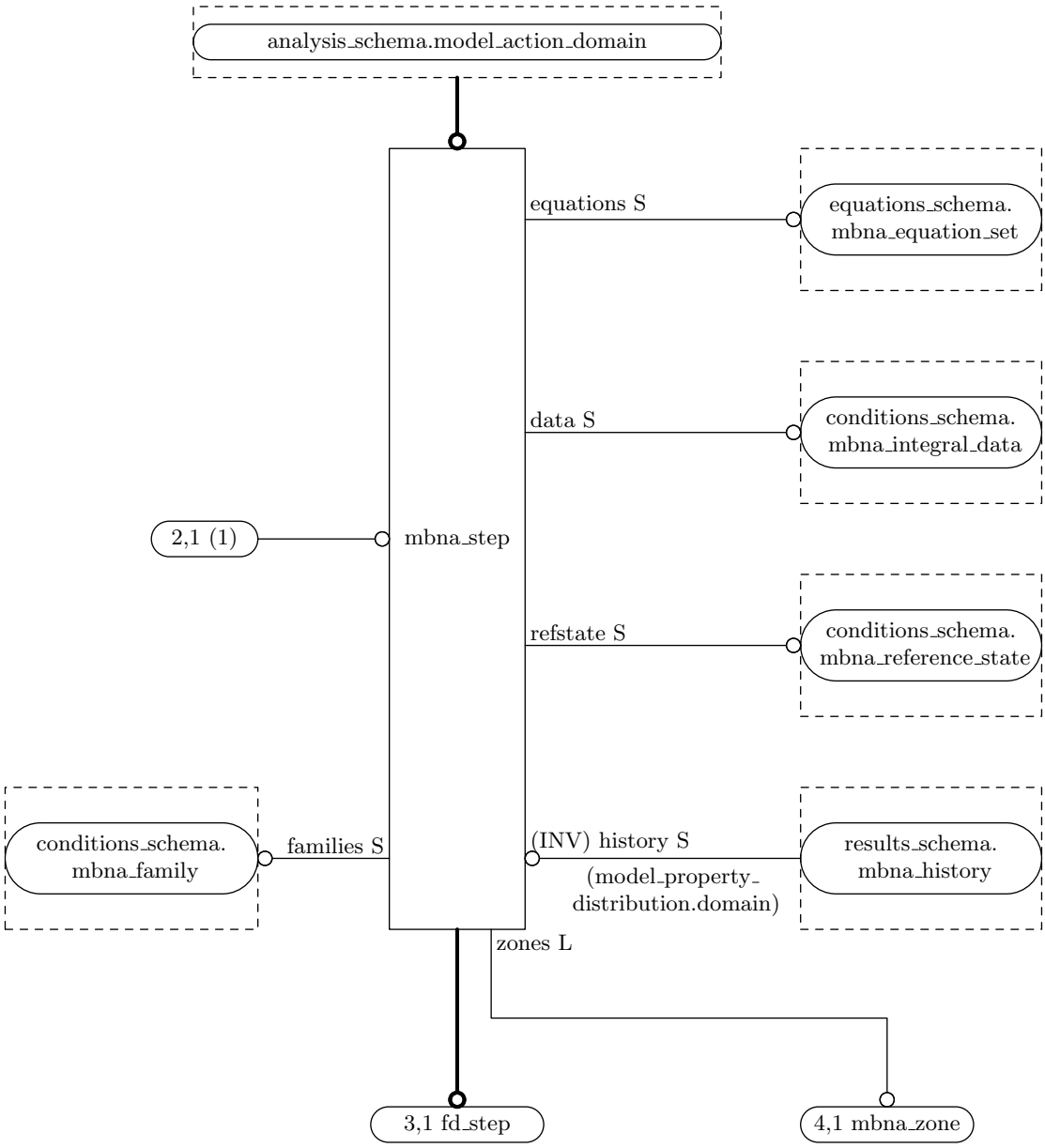

Figure D.7 – Entity level diagram of hierarchy schema (page 2 of 6)

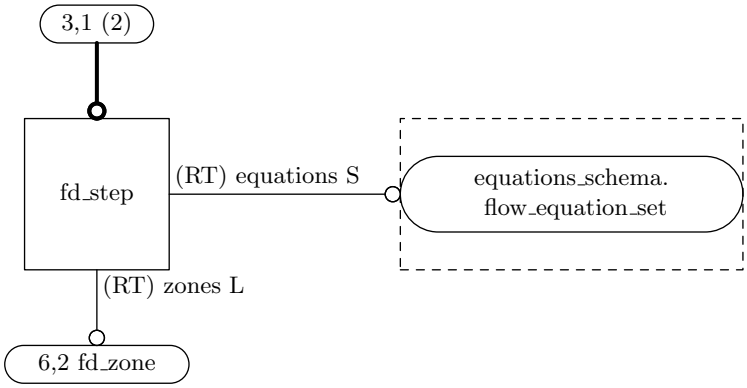

Figure D.8 – Entity level diagram of hierarchy schema (page 3 of 6)

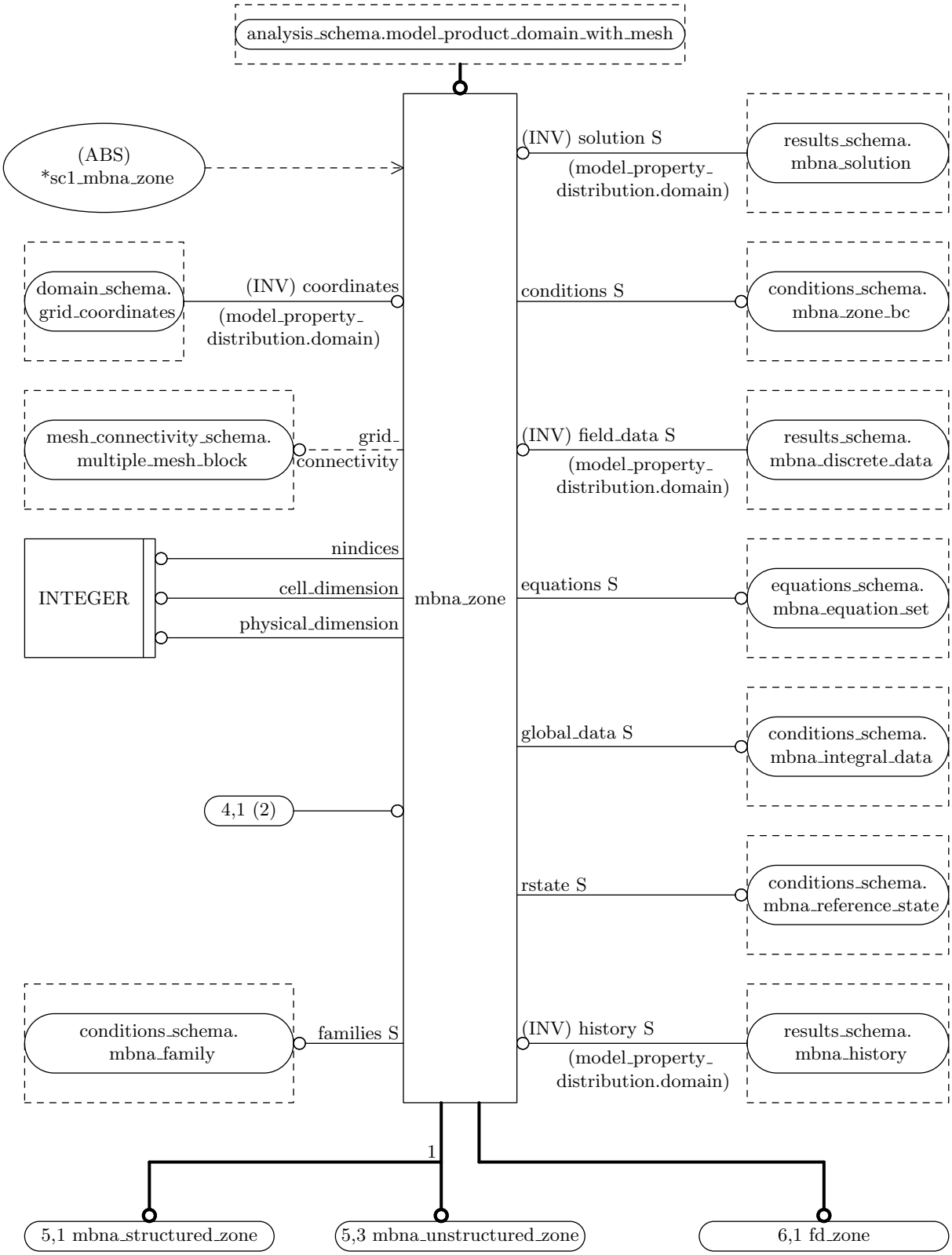

Figure D.9 – Entity level diagram of hierarchy schema (page 4 of 6)

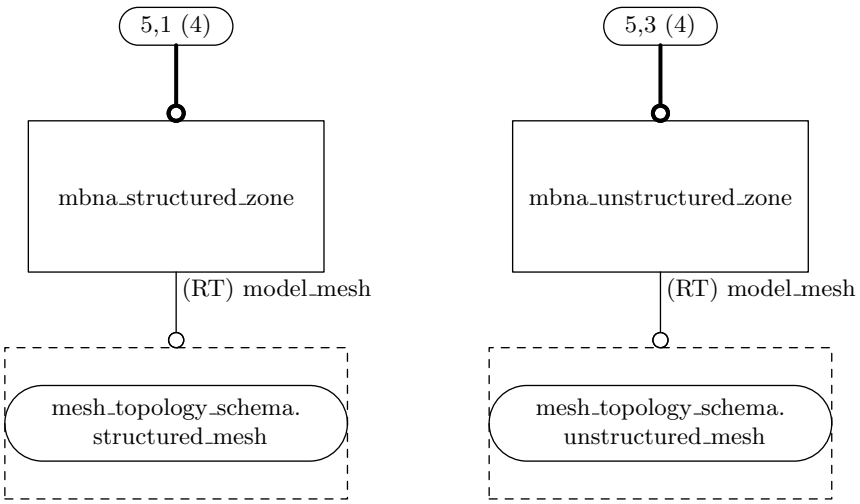

Figure D.10 – Entity level diagram of hierarchy schema (page 5 of 6)

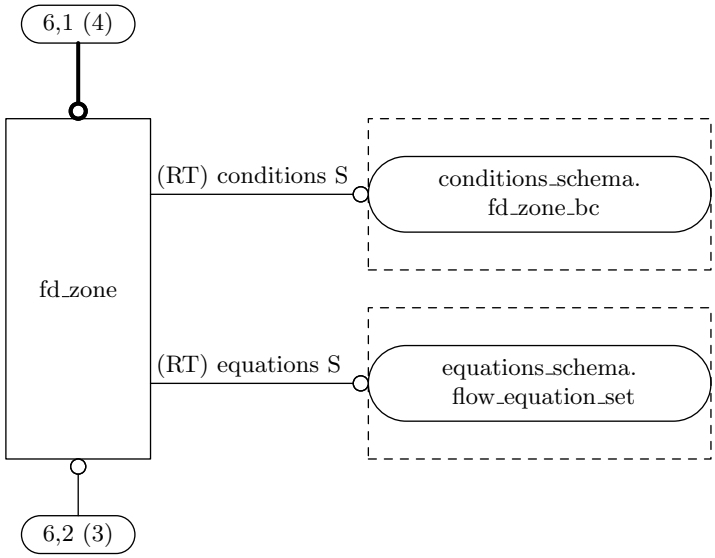

Figure D.11 – Entity level diagram of hierarchy schema (page 6 of 6)

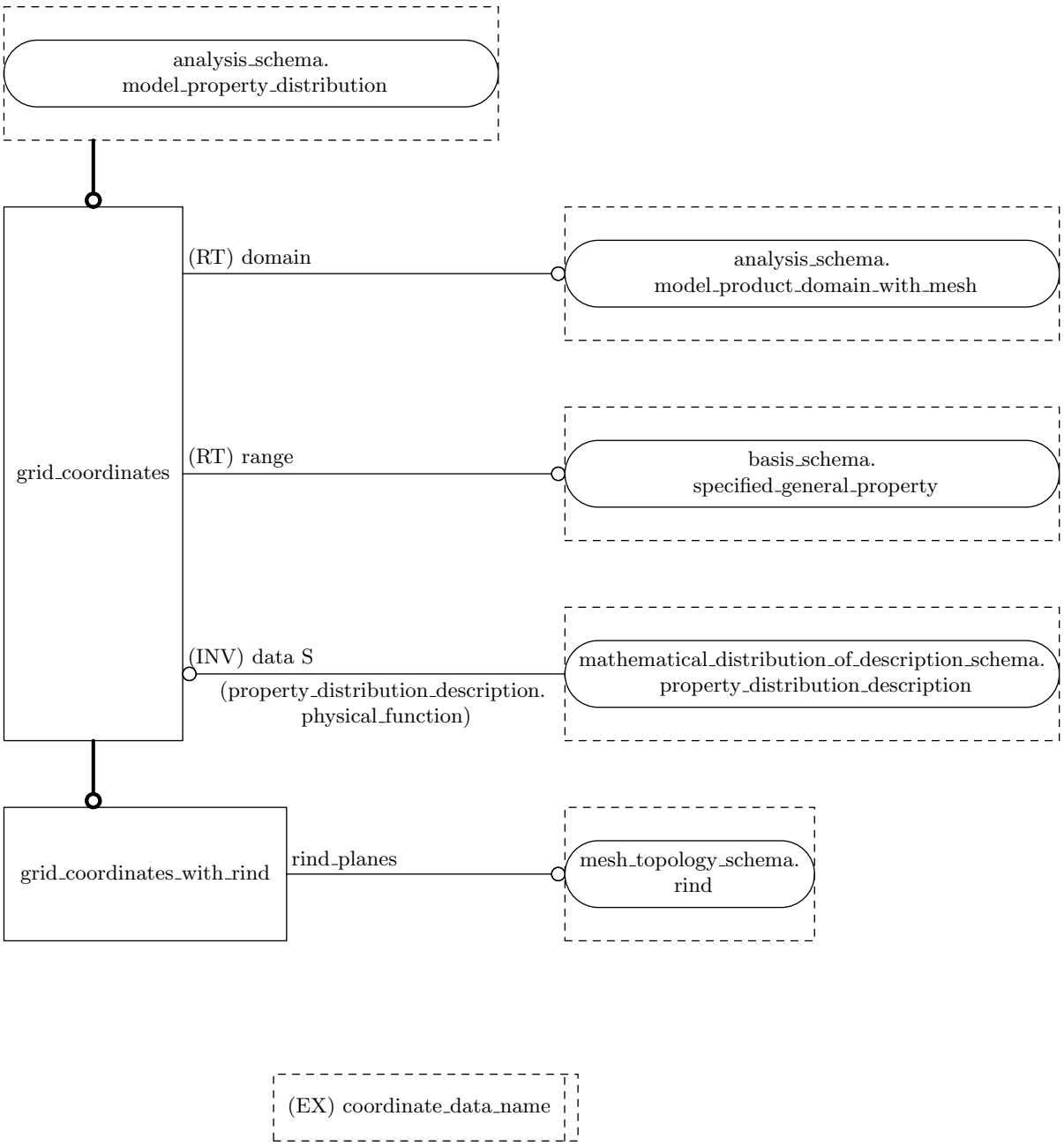

Figure D.12 – Entity level diagram of domain schema (page 1 of 1)

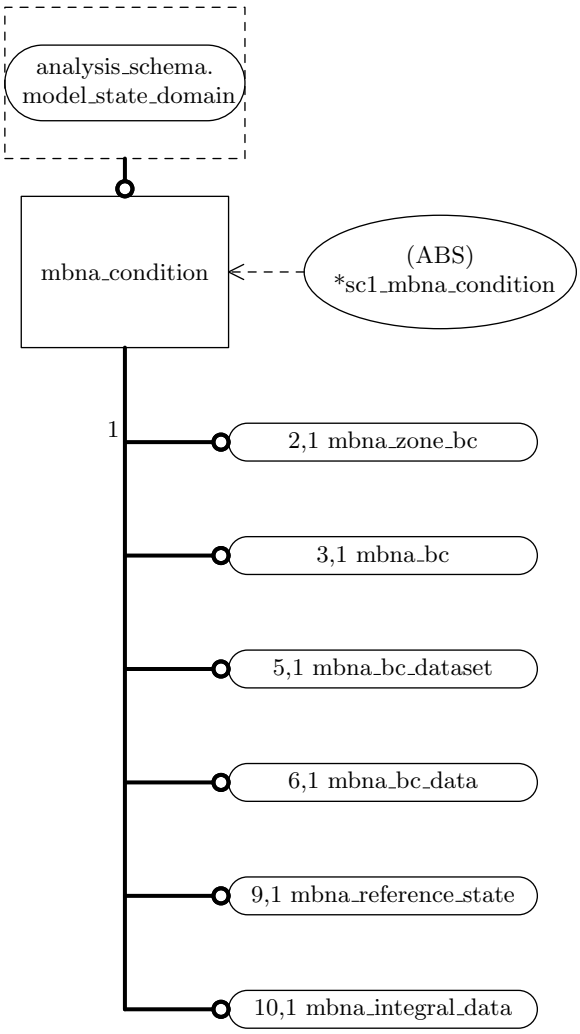

Figure D.13 – Entity level diagram of conditions schema (page 1 of 11)

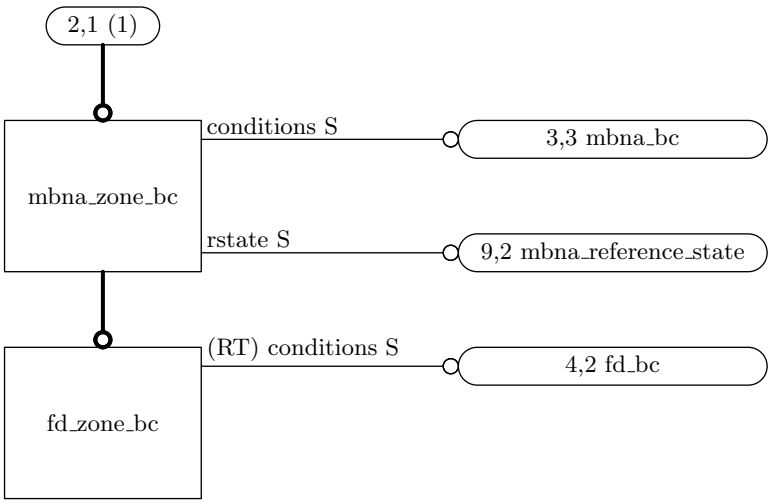

Figure D.14 – Entity level diagram of conditions schema (page 2 of 11)

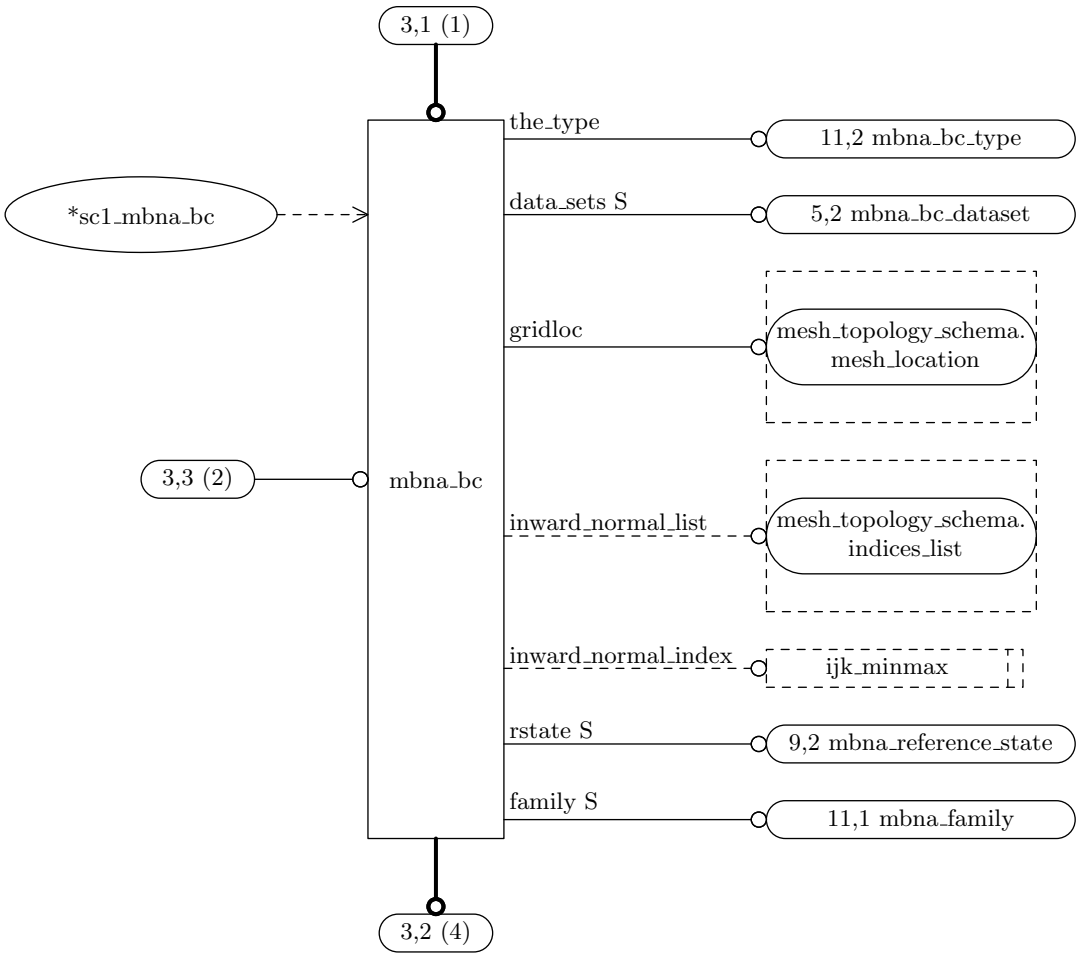

Figure D.15 – Entity level diagram of conditions schema (page 3 of 11)

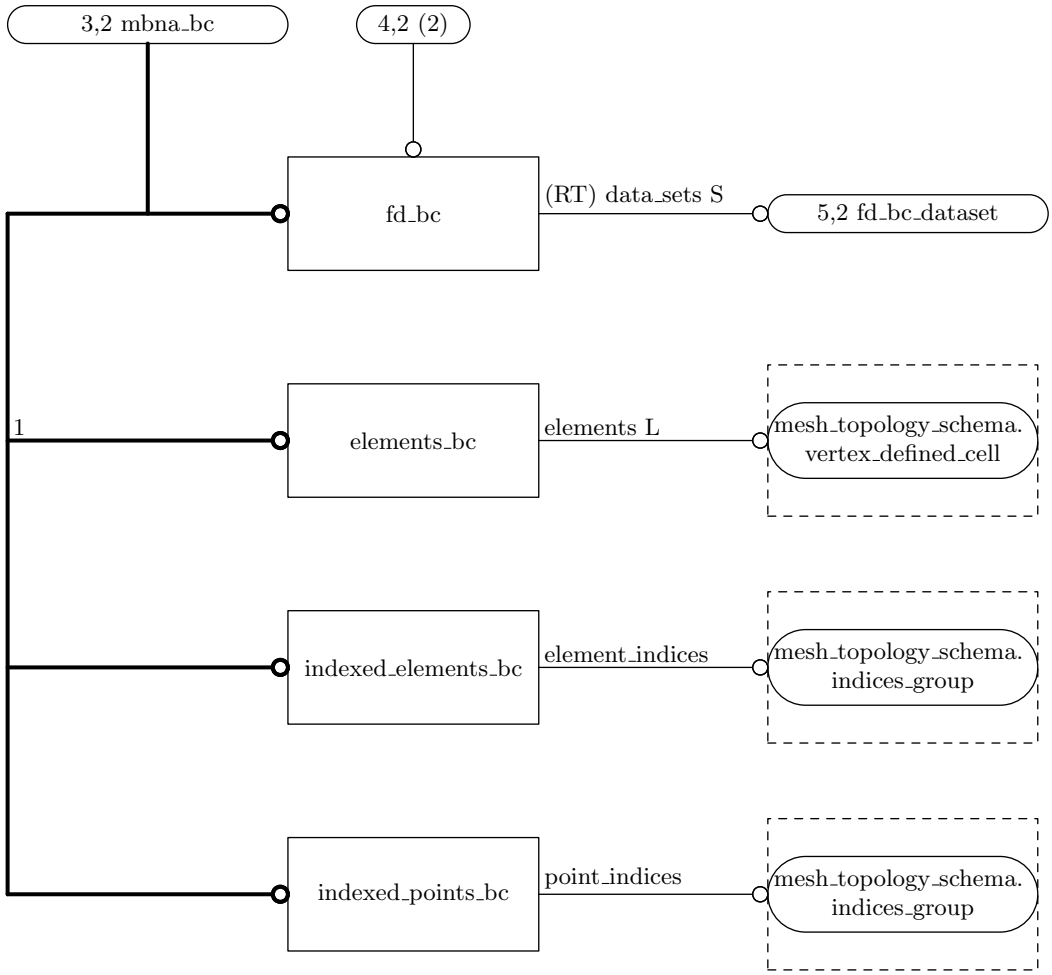

Figure D.16 – Entity level diagram of conditions schema (page 4 of 11)

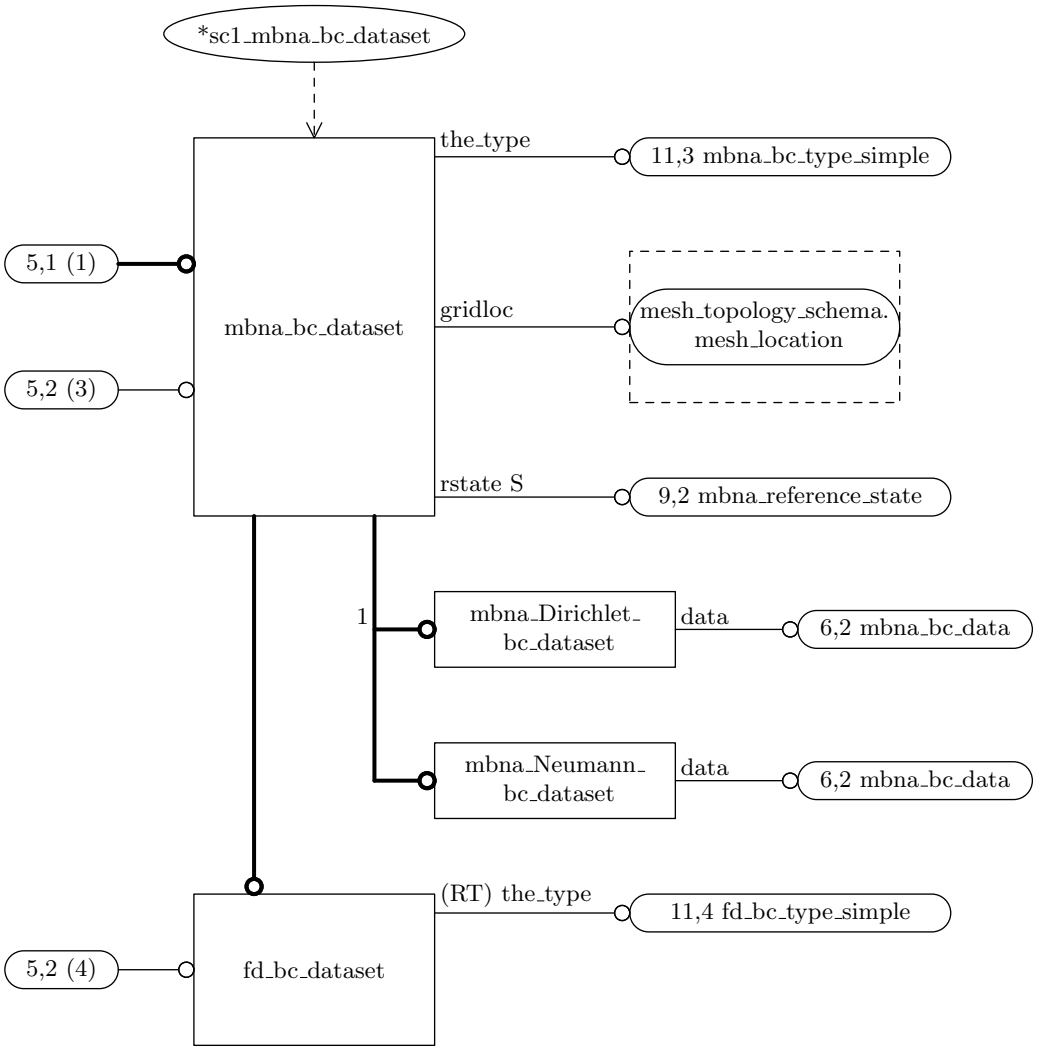

Figure D.17 – Entity level diagram of conditions schema (page 5 of 11)

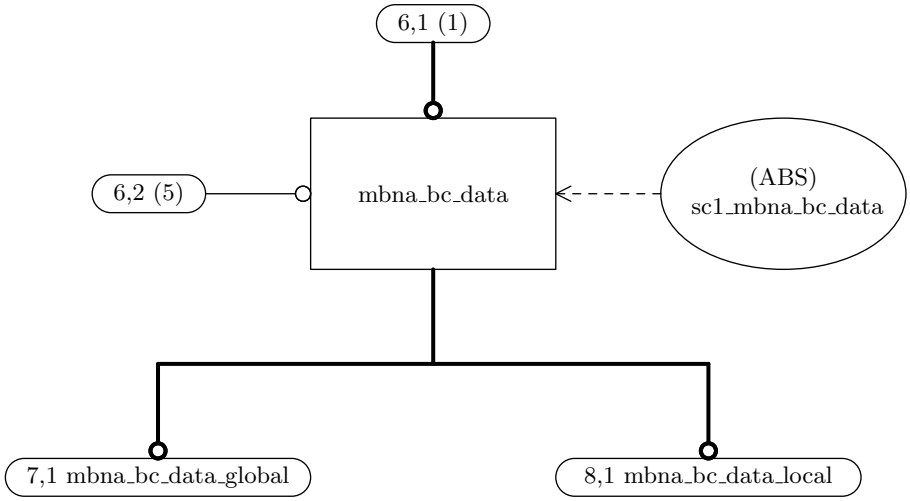

Figure D.18 – Entity level diagram of conditions schema (page 6 of 11)

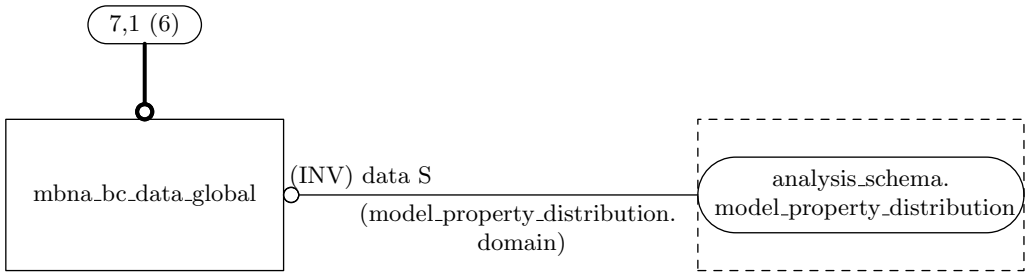

Figure D.19 – Entity level diagram of conditions schema (page 7 of 11)

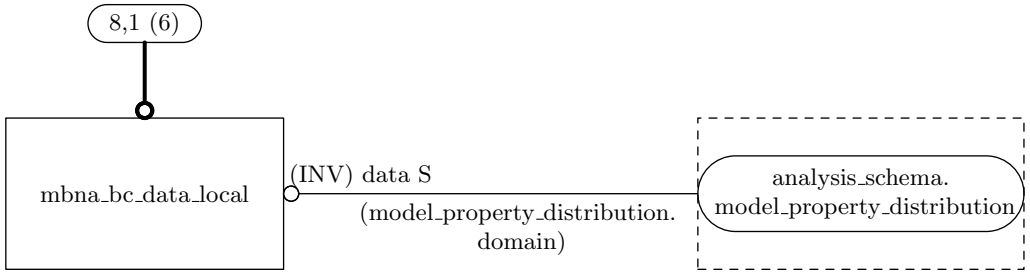

Figure D.20 – Entity level diagram of conditions schema (page 8 of 11)

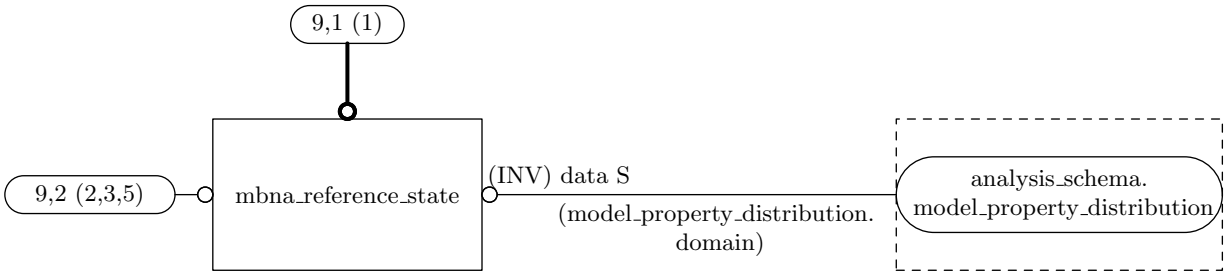

Figure D.21 – Entity level diagram of conditions schema (page 9 of 11)

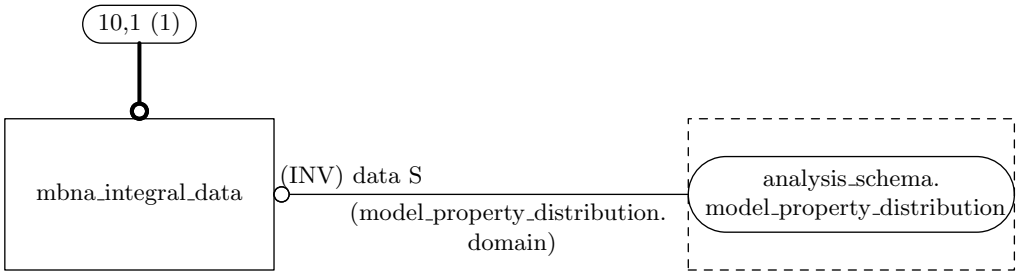

Figure D.22 – Entity level diagram of conditions schema (page 10 of 11)

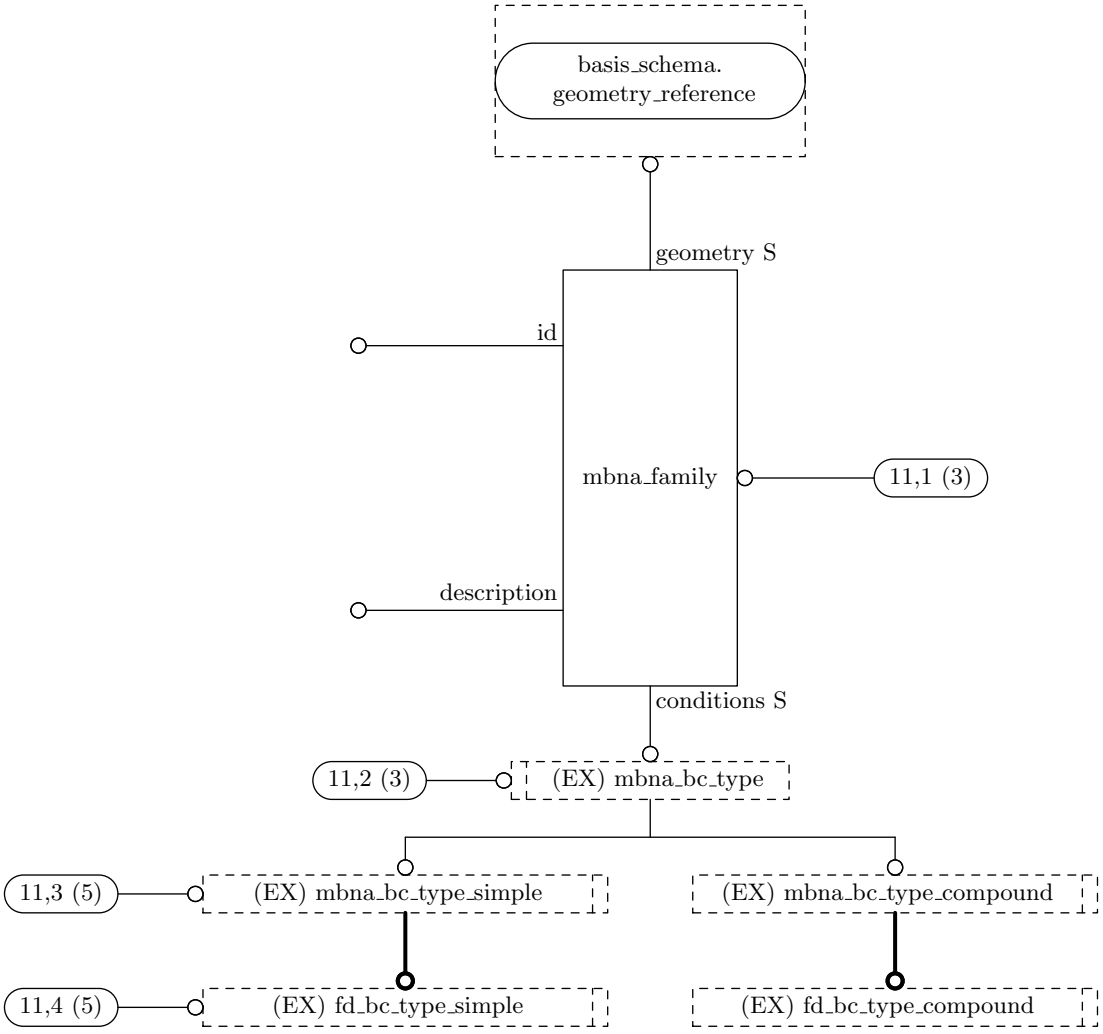

Figure D.23 – Entity level diagram of conditions schema (page 11 of 11)

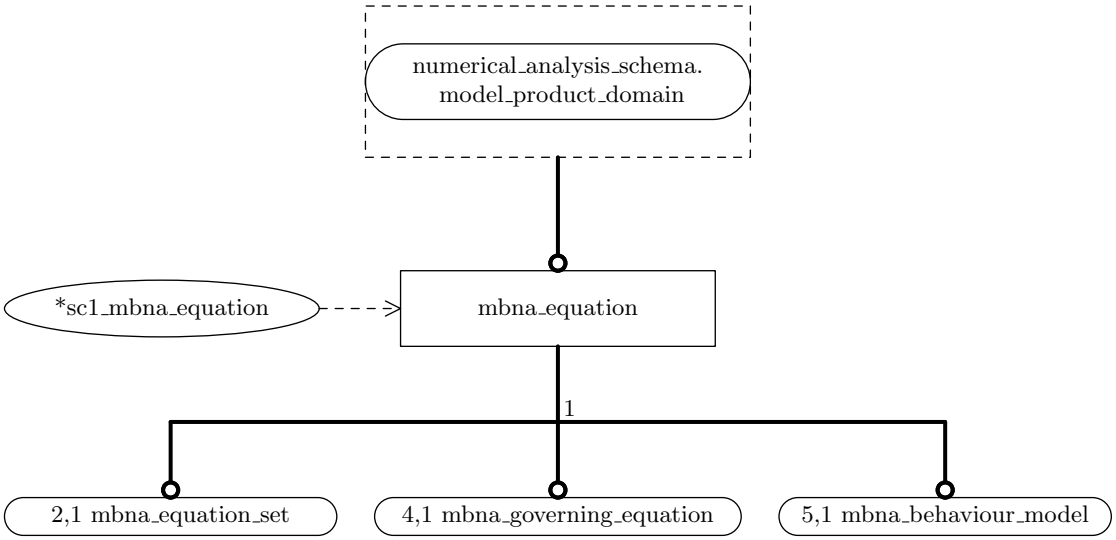

Figure D.24 – Entity level diagram of equations schema (page 1 of 6)

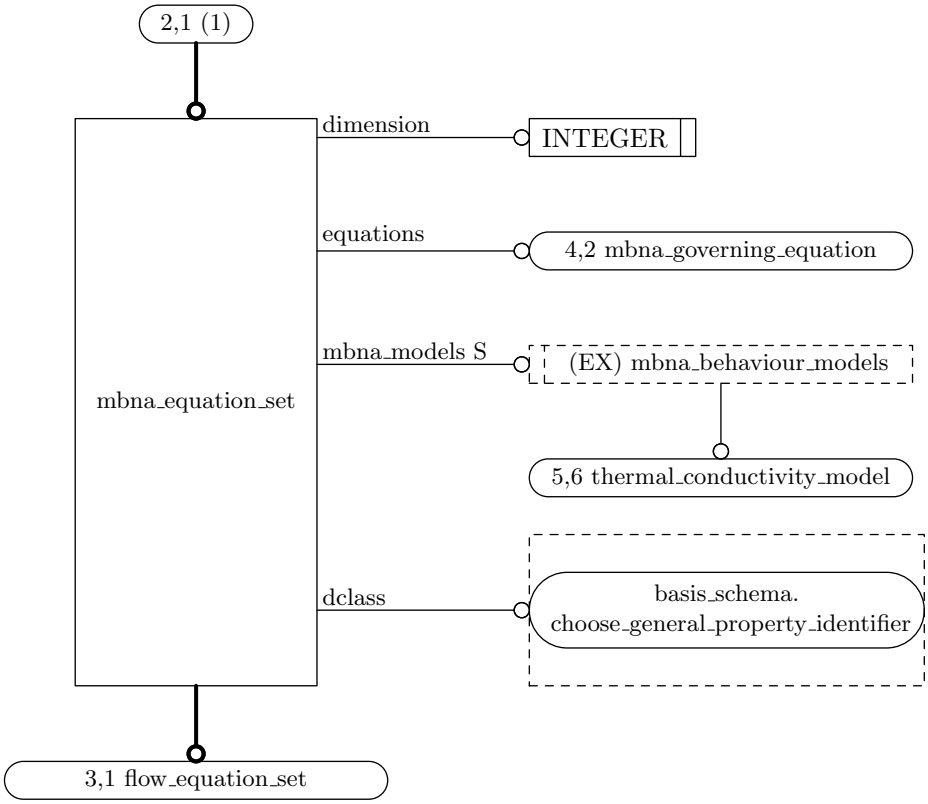

Figure D.25 – Entity level diagram of equations schema (page 2 of 6)

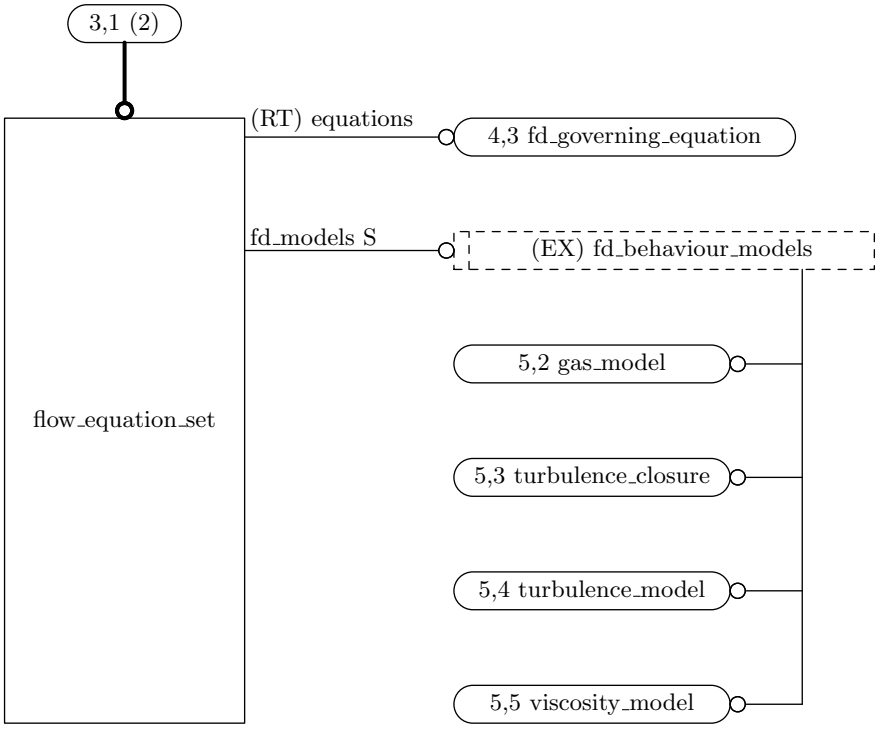

Figure D.26 – Entity level diagram of equations schema (page 3 of 6)

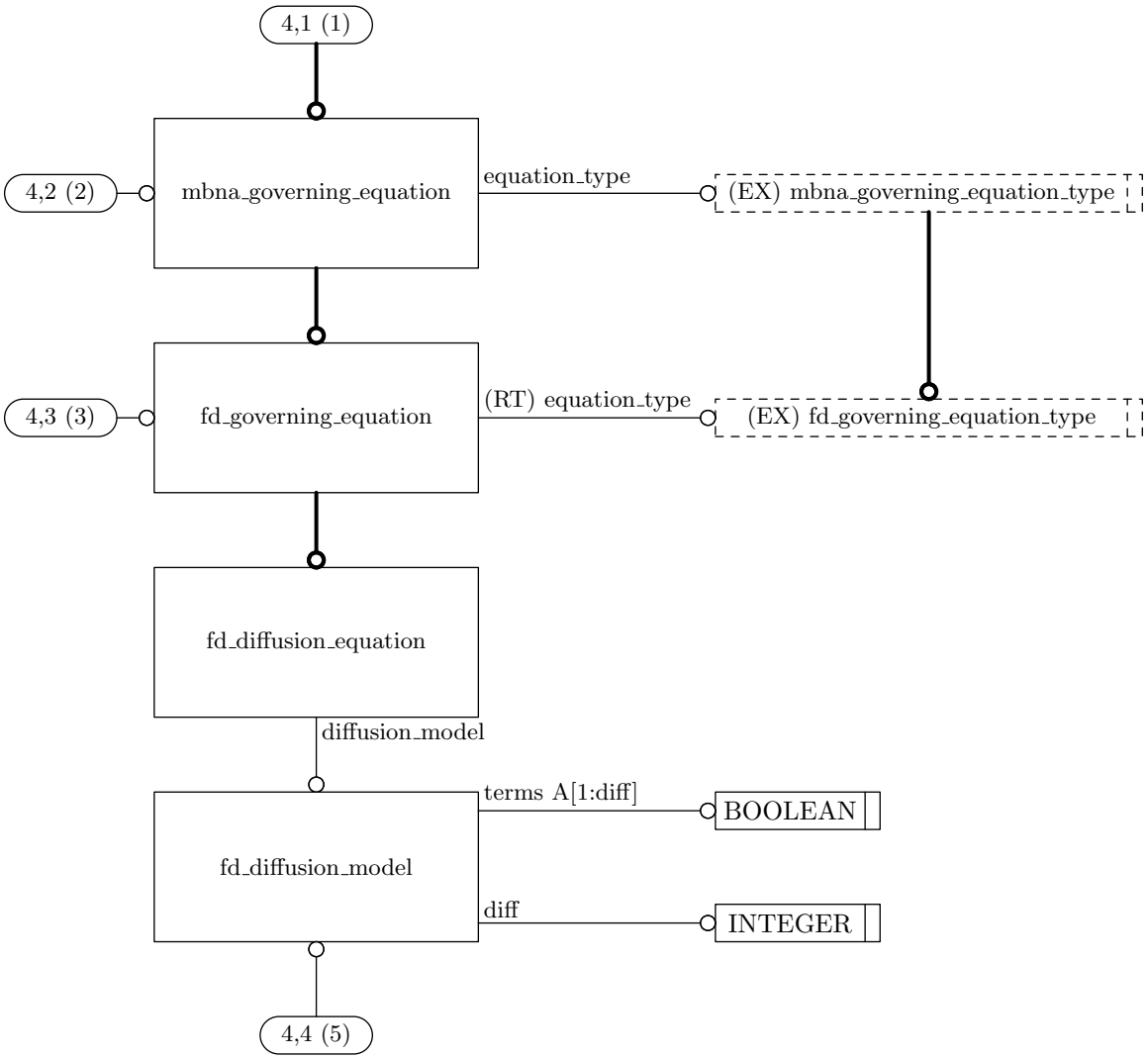

Figure D.27 – Entity level diagram of equations schema (page 4 of 6)

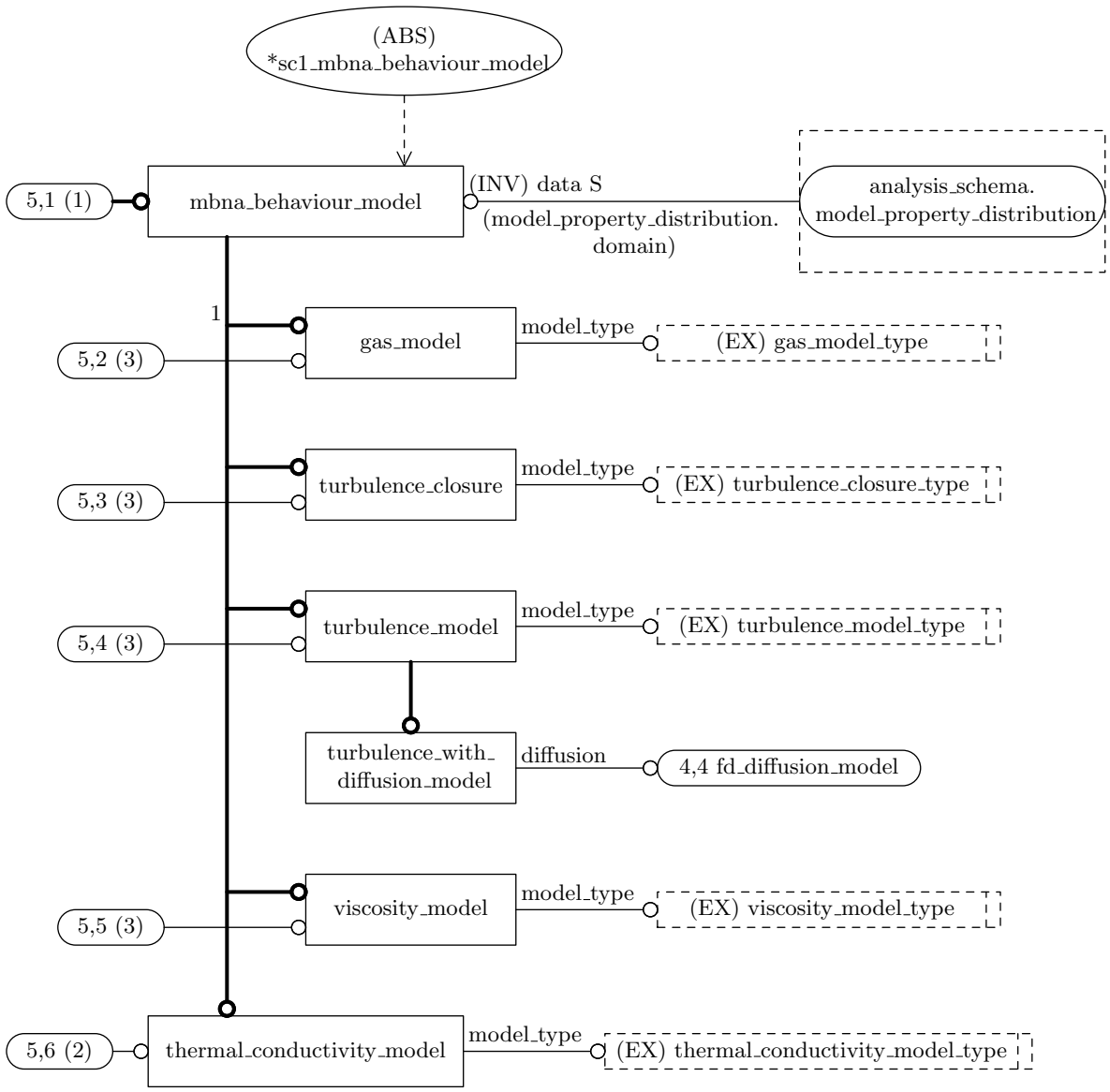

Figure D.28 – Entity level diagram of equations schema (page 5 of 6)

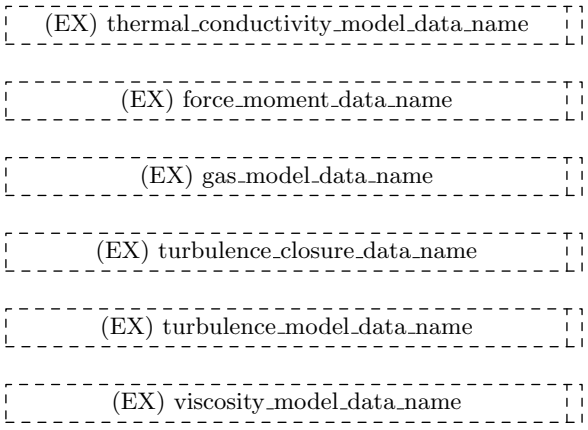

Figure D.29 – Entity level diagram of equations schema (page 6 of 6)

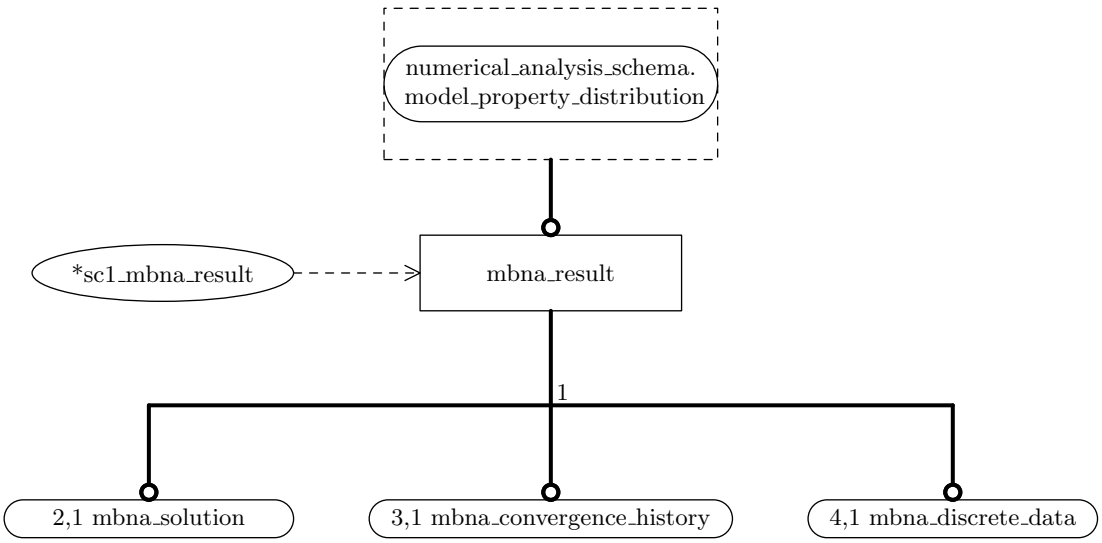

Figure D.30 – Entity level diagram of results schema (page 1 of 4)

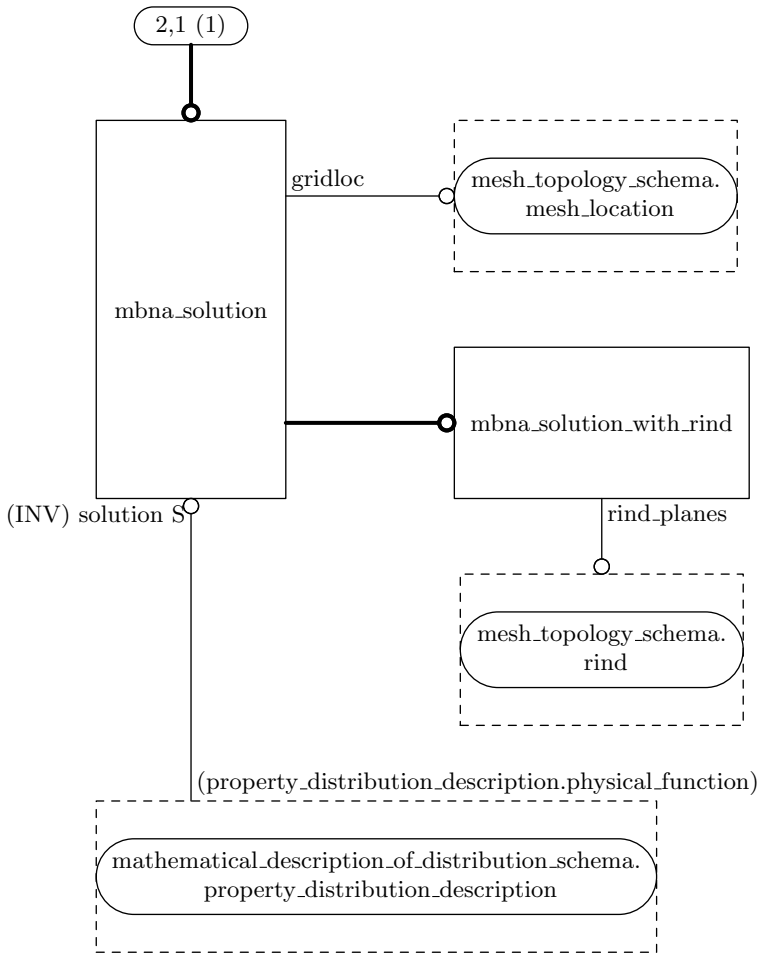

Figure D.31 – Entity level diagram of results schema (page 2 of 4)

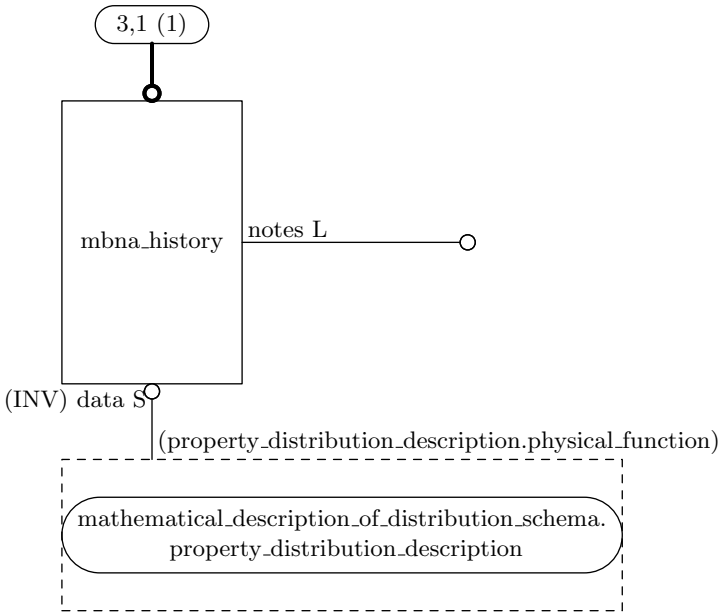

Figure D.32 – Entity level diagram of results schema (page 3 of 4)

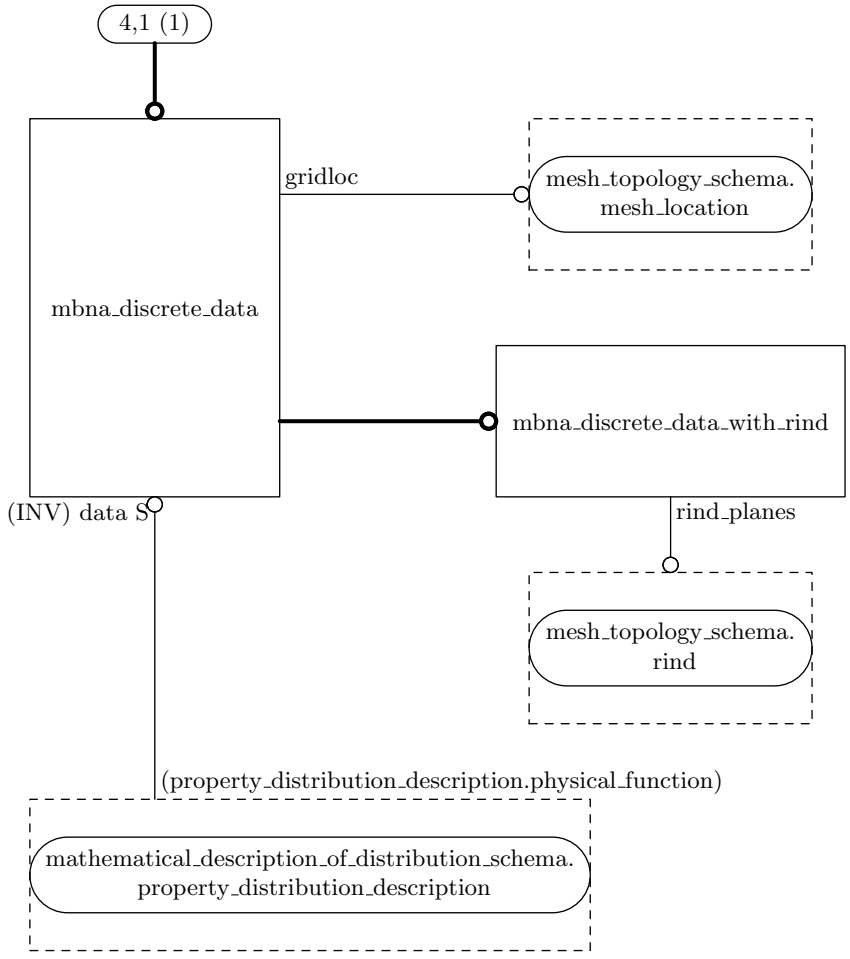

Figure D.33 – Entity level diagram of results schema (page 4 of 4)

Annex E
(informative)
Description by maths functions and founding

This annex discusses:

— the way in which a variation of property with respect to space, time or state is described by a maths function; and

— the way in which a representation context is specified for such a description.

Figure E.1 shows the main entities involved, and their relationships. The diagram is EXPRESS-G and has been simplified by removing some attributes and SELECT types. The numbers in parentheses are the ISO 10303 parts in which complete definitions of the entities can be found.

The representation context defines:

— units of measure; and

— the founding for representations of geometric data such as coordinates, and vector and tensor quantities.

The entity **numerical_model** is a model space, which can be:

— a set (finite or infinite) of different positions within a product;

— a set (finite or infinite) of different states of a product.

The entity **model_property_distribution** is the variation of a property over the model space.

EXAMPLE 1 The numerical model can be the discrete set of cells within a CFD model at a particular state. The model property distribution can be the pressure and a velocity for each cell.

EXAMPLE 2 The numerical model can be the continuous set of states of a material product at different temperatures. The model property distribution can be the variation of tensile yield strength with temperature.

The entity **general_property** is the physical quantity, such as position or pressure.

The entity **property_distribution_description** specifies how **model_property_distribution** is described by a **maths_function**. This entity assigns two contexts for the description as follows:

— **domain_context** is a parameterisation relationship that relates the domain of the maths function with the numerical model. Usually, each property distribution description for a numerical model uses the same parameterisation. This can be indicated by creating an instance of **maths_space_context** that is referenced as the **domain_context** by each description.

— **range_context** indicates the way in which the range of the maths function describes the physical quantity in the distribution. The use of **representation_context_defined_maths_space_context** ensures that the units of measure and the coordinate system are assigned exactly as in ISO 10303-104 and ISO 10303-209.

NOTE It is strongly recommended that the same **representation_context** be used for each **property_distribution_description** that references a **numerical_model**. As shown in Figure E.1, an **fea_model** has such a single **representation_context**.

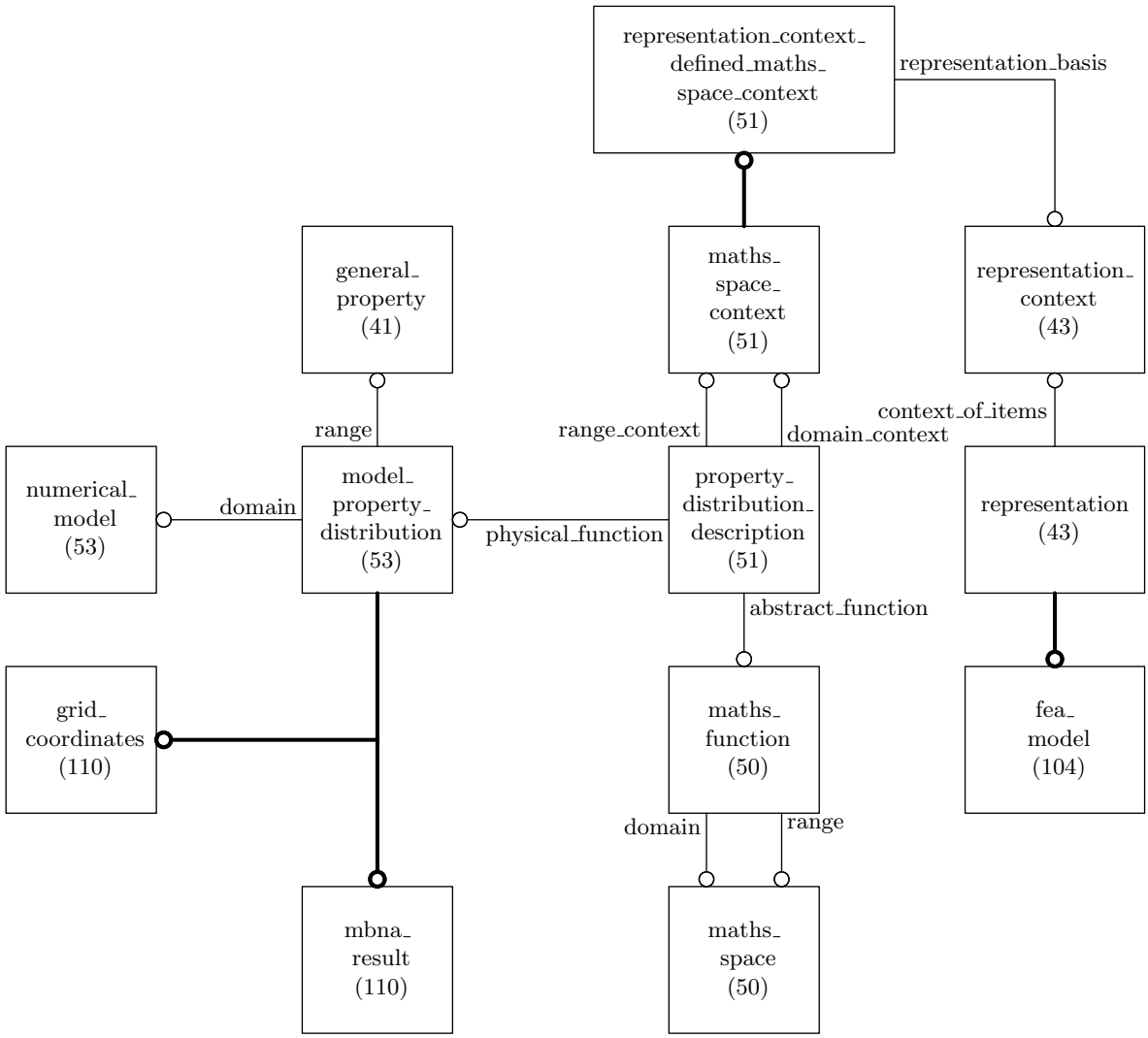

Figure E.1 – EXPRESS-G **partial model illustrating founding**

The entity **maths_function** provides a description for the **model_property_distribution**. If the **numerical_model** is a discrete space, then the maths function is a table of values. If the **numerical_model** is a continuous space, then the maths function is defined over a continuous domain.

Annex F
(informative)
Additional information

Table F.1 lists the elements of the **basis_schema** that are used by other schemas.

Table F.1 – Elements of basis_schema used by other schemas

Element	Used in schema	By element
geometry_reference	**conditions_schema**	**mbna_family**

Table F.2 lists the elements of the **conditions_schema** that are used by other schemas.

Table F.2 – Elements of conditions_schema used by other schemas

Element	Used in schema	By element
Riemann_1D_data_name	**basis_schema**	**fd_defined_data_name**
fd_zone_bc	**hierarchy_schema**	**fd_zone**
mbna_family		**mbna_zone**
mbna_integral_data		**mbna_model**
		mbna_step
		mbna_zone
mbna_reference_state		**mbna_step**
		mbna_zone
mbna_zone_bc		**mbna_zone**

Table F.3 lists the elements of the **domain_schema** that are used by other schemas. An entity that uses the **domain_schema** element as an inverse attribute is indicated by (INV).

Table F.3 – Elements of domain_schema used by other schemas

Element	Used in schema	By element
coordinate_data_name	**basis_schema**	**defined_data_name**
grid_coordinates	**hierarchy_schema**	**mbna_zone** (INV)

Table F.4 lists the elements of the **equations_schema** that are used by other schemas.

Table F.5 lists the elements of the **results_schema** that are used by other schemas. An entity that uses the **results_schema** element as an inverse attribute is indicated by (INV).

Table F.4 – Elements of equations_schema used by other schemas

Element	Used in schema	By element
force_moment_data_-name	basis_schema	fd_defined_data_name
gas_model_data_name		fd_defined_data_name
thermal_conductivity_-model_data_name		fd_defined_data_name
turbulence_closure_-data_name		fd_defined_data_name
turbulence_model_data_-name		fd_defined_data_name
viscosity_model_data_-name		fd_defined_data_name
flow_equation_set	hierarchy_schema	fd_step
mbna_equation_set		mbna_model mbna_step mbna_zone

Table F.5 – Elements of results_schema used by other schemas

Element	Used in schema	By element
flow_solution_data_name	basis_schema	fd_defined_data_name
mbna_discrete_data	hierarchy_schema	mbna_zone (INV)
mbna_history		mbna_model (INV) mbna_step (INV) mbna_zone (INV)
mbna_solution		mbna_zone (INV)

Bibliography

[1] ALLMARAS, S. and McCARTHY, D., *CGNS Standard Interface Structures,* 11 August 1999

[2] POIRIER, D., *SIDS additions/modifications to support unstructured meshes and geometry links,* June 1999

[3] CGNS PROJECT GROUP, *The CFD General Notation System: Standard Interface Data Structures,* August 2000

[4] WHITE, F. M., *Viscous Fluid Flow,* McGraw-Hill, 1974

Index

