



TECHNICAL REPORT	EIA-IS-116™	
	Issued 1996-12 Stabilized 2015-07	
CDIF Integrated Meta-model - State/Event Model Subject Area		

NOTICE

This document has been taken directly from the original TechAmerica document and contains only minor editorial and format changes required to bring it into conformance with the publishing requirements of SAE Technical Standards. The release of this document is intended to replace the original with the SAE International document. Any numbers established by the original document remain unchanged.

The original document was adopted as an SAE publication under the provisions of the SAE Technical Standards Board (TSB) Rules and Regulations (TSB 001) pertaining to accelerated adoption of specifications and standards. TSB rules provide for (a) the publication of portions of unrevised specifications and standards without consensus voting at the SAE committee level, and (b) the use of the existing specification or standard format.

SAE Technical Standards Board Rules provide that: "This report is published by SAE to advance the state of technical and engineering sciences. The use of this report is entirely voluntary, and its applicability and suitability for any particular use, including any patent infringement arising therefrom, is the sole responsibility of the user."

SAE reviews each technical report at least every five years at which time it may be revised, reaffirmed, stabilized, or cancelled. SAE invites your written comments and suggestions.

Copyright © 2016 SAE International

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior written permission of SAE.

TO PLACE A DOCUMENT ORDER: Tel: 877-606-7323 (inside USA and Canada)
Tel: +1 724-776-4970 (outside USA)
Fax: 724-776-0790
Email: CustomerService@sae.org
<http://www.sae.org>

SAE WEB ADDRESS:

Full document details: SAE EIAIS116

SAE values your input. To provide feedback on this Technical Report, please visit <http://www.sae.org/technical/standards/EIAIS116>

Documents
Document Sharing Hub

Interim Standard

EIA/IS-116

CDIF

CASE Data Interchange Format
A Division of Electronic Industries Association

CDIF
Integrated Meta-model -
State/Event Model Subject Area

December 1996
Electronic Industries Association
CDIF Technical Committee

NOTICE

EIA Engineering standards and publications are designed to serve the public interest through eliminating misunderstandings between manufacturers and purchasers, facilitating interchangeability and improvement of products, and assisting the purchaser in selecting and obtaining with minimum delay the proper product for his particular need. Existence of such standards and publications shall not in any respect preclude any member or nonmember of EIA from manufacturing or selling products not conforming to such standards and publications, nor shall the existence of such standards and publications preclude their voluntary use by those other than EIA members, whether the standard is to be used either domestically or internationally.

Recommended standards and publications are adopted by EIA without regard to whether or not their adoption may involve patents on articles, materials or processes. By such action, EIA does not assume any liability to any patent owner, nor does it assume any obligation whatever to parties adopting the standard or publication.

EIA INTERIM STANDARDS

EIA Interim Standards contain information deemed to be of technical value to the industry, and are published at the request of the originating Committee without necessarily following the rigorous public review and resolution of comments which is a procedural part of the development of an EIA Recommended Standard.

EIA Interim Standards should be reviewed on an annual basis by the formulating Committee and a decision made on whether to proceed to develop an EIA Recommended Standard on this subject. EIA Interim Standards must be canceled by the Committee and removed from the EIA Standards Catalog before the end of their third year of existence.

This EIA standard is considered to have international standardization implication, but the International Standardization Organization activity has not progressed to the point where a valid comparison between the EIA standard and the ISO work can be made.

COPYRIGHT 1996

Published by
ELECTRONIC INDUSTRIES ASSOCIATION
Engineering Department
2500 Wilson Blvd.
Arlington, VA 22201

PRICE: Please refer to the current
Catalog of EIA & JEDEC STANDARDS & ENGINEERING PUBLICATIONS
or contact EIA Electronic Information Group
Phone: (703) 907-2545, FAX: (703) 907-7501

All rights reserved
Printed in U.S.A.

ISBN 0-7908-0022-5

PREFACE

This standard will assist the vendors and users of CASE tools in developing mechanisms for interchanging information between CASE tools. This standard specifies an element of a family of related standards. When used together, these standards specify a mechanism for transferring information between CASE tools.

EIA/IS-106 CDIF - CASE Data Interchange Format - Overview and EIA/IS-107 CDIF - Framework for Modeling and Extensibility should be read first when initially exploring CDIF. The first explains the overall CDIF Architecture and how the family of standards fits together. The second explains the scope, and modeling approach in CDIF. The CDIF Meta-meta-model and extensibility mechanism are also defined in that document.

This standard explains the State/Event Model Subject Area. This is one of the subject areas of the CDIF Integrated Meta-model, which is used to ensure that the information held by tools communicating using CDIF is transferred with an agreed meaning. It addresses the semantic information of event-driven finite state machine modeling and covers component states, actions, transitions between states, and their enabling events and conditions. State/event models may be decomposed into sub-models. These sub-models can execute concurrently, sequentially, or as mutually exclusive alternates. These semantics can apply to Object-oriented Behavior models and Entity-Life-History models as well as State Machines.

This standard has been developed with the wide support and participation of vendors, users, academia and government involved in or familiar with the CASE industry, its products and the general requirements associated with interchanging information between these products.

The material contained in this publication was formulated under the cognizance of the CDIF Division of the Electronic Industries Association.

CONTENTS

1. SCOPE	1
2. INTRODUCTION.....	2
2.1 Audience	2
2.2 Structure of this Document	2
2.3 Conventions	3
3. STRUCTURE OF SUBJECT AREA DEFINITION	5
3.1 Introduction.....	5
3.2 Structure of Subject Area Overview.....	5
3.2.1 Summary.....	5
3.2.2 Description of Major Concepts.....	5
3.2.3 Diagram(s).....	5
3.2.4 AttributableMetaObject Hierarchy	7
3.2.5 MetaEntity and MetaRelationship Summaries.....	8
3.3 Structure of Subject Area Detailed Definitions.....	8
3.3.1 Summary.....	8
3.3.2 Subject Area Definition	9
3.3.3 MetaEntity Definitions	9
3.3.4 MetaRelationship Definitions.....	11
3.4 Structure of Referenced Meta-object Definitions.....	12
4. SUBJECT AREA OVERVIEW	14
4.1 Description.....	14
4.1.1 Introduction	14
4.1.2 State/Event Model	15
4.1.3 States.....	15
4.1.4 Transitions	16
4.1.5 Action.....	16
4.1.6 Events and Conditions.....	16
4.1.7 The General Structuring Mechanism.....	17
4.1.7.1 Introduction.....	17
4.1.7.2 Meta-entities and Meta-relationships.....	17
4.1.7.3 DefinitionObject and ComponentObject	18
4.1.8 State Decomposition and Definition	20
4.1.9 Action Decomposition and Definition.....	21
4.1.10 Accessing a Specific Component in a Shared Definition.....	22
4.1.11 Identical Instances in Shared Definitions.....	24
4.1.12 Alternate Decompositions	25
4.1.13 Transition Splits and Merges	26
4.1.14 Interface of States	28

4.1.15 Subtypes of TransitionPort	31
4.1.16 Actions.....	33
4.1.17 Conflicts and Priorities	33
4.1.18 Timing Behavior.....	34
4.1.19 Computable Languages	34
4.1.20 Formats for Computable Values	34
4.2 Diagrams.....	35
4.3 AttributableMetaObject Hierarchy.....	39
4.4 MetaEntity Summary.....	43
4.5 MetaRelationship Summary	53
5. SUBJECT AREA DETAILED DEFINITIONS.....	55
5.1 Introduction.....	55
5.2 SubjectArea Definition.....	55
5.3 MetaEntity Definitions.....	56
5.3.1 ComponentObject	59
5.3.2 DefinitionObject	60
5.3.3 EquivalenceSet.....	63
5.3.4 ReferencedElement	64
5.3.5 SemanticInformationObject	65
5.4 MetaRelationship Definitions	67
5.4.1 ComponentObject.References.DefinitionObject	67
5.4.2 DefinitionObject.Contains.ComponentObject.....	68
5.4.3 EquivalenceSet.HasMember.ComponentObject	69
5.4.4 ReferencedElement.DefinesPath.ComponentObject	70
6. REFERENCED OBJECT DEFINITIONS	71
6.1 AttributableMetaObject Definitions.....	71
6.1.1 RootObject	71
6.2 MetaEntity Definitions.....	74
6.2.1 RootEntity	74
6.3 MetaRelationship Definitions	75
6.3.1 RootEntity.IsRelatedTo.RootEntity.....	75
APPENDIX A EXAMPLE	76
A1. Introduction.....	76
APPENDIX B¹ QUESTIONS AND ANSWERS	77
GLOSSARY¹.....	79
INDEX.....	86

RELATED EIA STANDARDS	90
------------------------------------	-----------

FIGURES

Figure 1 Position in the CDIF Family of Standards.....	1
Figure 2 Conventions for Representing Meta-entities.....	6
Figure 3 Conventions for Representing Meta-relationships.....	6
Figure 4 Conceptual Diagram and an Instance Diagram.....	7
Figure 5 State decomposition and hierarchy of state machines.....	15
Figure 6 General Structuring Mechanism.....	18
Figure 7 States sharing the same structured definition.....	19
Figure 8 A states decomposition.....	20
Figure 9 Structured Actions.....	21
Figure 10 Shared definition for structured actions.....	23
Figure 11 Referring to a specific instance inside a shared definition.....	24
Figure 12 Denoting the equivalence of two <i>States</i>	25
Figure 13 Alternate State decompositions.....	26
Figure 14 A <i>Transition Split</i> into component <i>Transitions</i>	27
Figure 15 A <i>Transition Split</i>	27
Figure 16 Instance diagram for a <i>Transition Split</i>	28
Figure 17 Interface of a state referencing a reusable definition.....	30
Figure 18 <i>TransitionPorts</i> describing the formal interface of a <i>StateDefinition</i>	31
Figure 19 Interface of a State using <i>TransitionPorts</i>	32
Figure 20 State/Event Subject Area Overview.....	35
Figure 21 Connection between State/Event Model Subject Area and Data Definition Subject Areas.....	37
Figure 22 State/Event Model Subject Area – Expression Definitions.....	37
Figure 23 State/Event Model Subject Area – Action Definitions.....	38
Figure 24 State/Event Model Subject Area – Transition Ports.....	38

TABLES

Table 1	Contents of Subject Area Definition	9
Table 2	Contents of Meta-entity Definition	10
Table 3	Contents of Local Meta-attribute Definition	11
Table 4	Contents of Meta-relationship Definition	12
Table 5	Allowable <i>DefinitionObject</i> components	19

1. SCOPE

The CDIF Family of Standards is primarily designed to be used as a description of a mechanism for transferring information between CASE tools. It facilitates a successful transfer when the authors of the importing and exporting tools have nothing in common except an agreement to conform to CDIF. The language that is defined for the Transfer Format also has applicability as a general language for Import/Export from repositories. The CDIF Integrated Meta-model defined for CASE also has applicability as the basis of standard definitions for use in repositories.

The standards that form the complete family of CDIF Standards are documented in *EIA/IS-106 CDIF - CASE Data Interchange Format - Overview*. These standards cover the overall framework, the transfer format and the CDIF Integrated Meta-model.

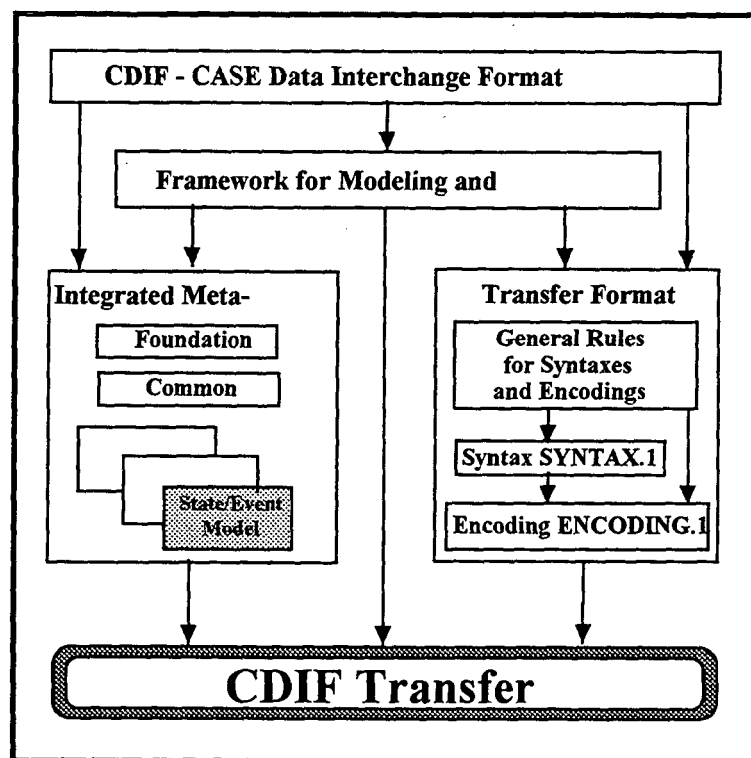


Figure 1
Position in the CDIF Family of Standards

The diagram in Figure 1 depicts the various standards that comprise the CDIF Family of Standards. The shaded box depicts this standard and its position in the CDIF Family of Standards.

2. INTRODUCTION

2.1 Audience

This document is intended to be used by anyone wishing to understand and/or use CDIF. This document provides a definition of a single subject area of the CDIF Integrated Meta-model. It is suitable for:

- those evaluating CDIF
- those who wish to understand the principles and concepts of a CDIF transfer
- those developing importers and exporters.

The document *EIA/IS-106 CDIF - CASE Data Interchange Format - Overview* and the document *EIA/IS-107 CDIF - Framework for Modeling and Extensibility* should be read first when initially exploring CDIF and before attempting to read other documents in the CDIF Family of Standards.

While there are no specific prerequisites for reading this document, it will be helpful for the reader to have familiarity with the following:

- Entity-Relationship-Attribute modeling
- CASE tools
- Information repositories
- Data dictionaries
- Multiple meta-layer modeling.

2.2 Structure of this Document

This document is organized into the following sections:

Section 1 'Scope'

This describes the CDIF Family of Standards and positions the current document within the set of documents.

Section 2 'Introduction'

This describes the intended audience, structure and conventions used in the document.

Section 3 'Structure of Subject Area Definition'

This describes in detail the structure of the subsequent sections of this standard including the notation and layout of the diagrams and the detailed definitions.

Section 4 'Subject Area Overview'

This gives an overview of the coverage of this subject area.

Section 5 'Subject Area Detailed Definitions'

This gives the detailed definitions of all the objects used in the subject area.

Section 6 'Referenced Meta-object Definitions'

This gives the detailed definitions of objects that are not used in the subject area defined in this Standard, but are referenced by objects that are. These definitions are included to avoid the need to reference multiple standards to obtain the complete definition of the current subject area.

Appendix A 'Examples'

This gives one or more examples of how information is represented using this CDIF Subject Area. Where there are different technique variants used within the coverage of a single Subject Area, the examples attempt to show how some of these different variants are represented in the CDIF Integrated Meta-model.

Appendix B 'Questions and Answers'

This includes detailed responses to questions that have been raised by those reviewing this document. It is intended that the inclusion of these will aid the reader in understanding this document, and related issues concerning CDIF.

Glossary

The Glossary contains definitions of terms used within this Standard.

Index

A comprehensive index is provided.

2.3 Conventions

The typographical and naming conventions defined below are used throughout the CDIF Family of Standards.

- Double quotes are used to introduce new terms (e.g., "model layer")
- Bold type is used for emphasis (e.g., this is **important**)
- All meta-objects and meta-meta-objects in CDIF (in meta-models and meta-meta-models) are named by concatenating all the words that name the meta-object or meta-meta-object; the first letter of each word is upper-case, the rest are lower-case. These names are shown in *italic* (e.g., *MetaAttribute*, *AttributeDerivation*, *IsDrawnUsing*, *IsOptional*)

- The names of objects and values for meta-attributes that appear in examples and instance diagrams are shown in bold italic (e.g., ***Customer, True, 100***)

Full details of the CDIF Graphical Notation used in the Meta-model and Meta-meta-model can be found in *EIA/IS-107 CDIF - Framework for Modeling and Extensibility*. A summary of the major concepts is given in Section 3.2.3, Diagram(s).

3. STRUCTURE OF SUBJECT AREA DEFINITION

3.1 Introduction

The following subsections describe how the subject area is defined in the remainder of this document. The subject area is first introduced in the Subject Area Overview Section, and the detailed definitions of the objects contained in the subject area are given in the Subject Area Definitions Section.

3.2 Structure of Subject Area Overview

3.2.1 Summary

The Subject Area Overview Section consists of the following sub-sections:

- Description of Major Concepts
- Diagram(s)
- AttributableMetaObject Hierarchy
- MetaEntity Summary
- MetaRelationship Summary

3.2.2 Description of Major Concepts

This provides a textual description of the major concepts in the subject area. It explains the coverage of this subject area and outlines what it is capable of representing.

3.2.3 Diagram(s)

Included in the Subject Area Overview are various diagrams. These diagrams are intended to graphically present an overview of the meta-entities and meta-relationships used in this subject area. All CDIF subject area diagrams use the following diagramming notation conventions.

Meta-entities used in the subject area are shown as rectangular boxes, with the name of the meta-entity on a white background. An example is the meta-entity *A* in Figure 2. An additional special notation may be used to represent meta-entities on diagrams of meta-models. If the meta-entity is not used in the subject area, but is used in another subject area and is included for clarity, the box shall be shaded. An example is the meta-entity *B* in Figure 2.

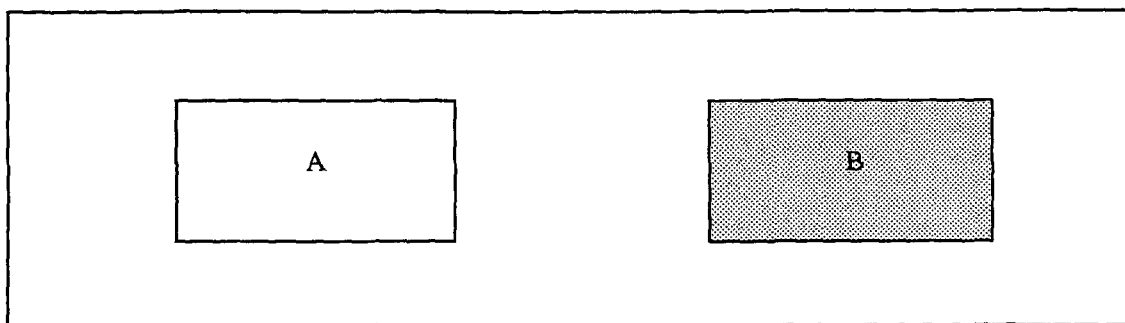


Figure 2
Conventions for Representing Meta-entities

Meta-relationships are normally represented by a line joining the source and destination entities, with the name of the meta-relationship given next to the line. An arrow on the line indicates the direction in which to read the name. The meta-relationship *C* in Figure 3 is an example of this. Additionally, two special notations may be used to represent meta-relationships on diagrams of meta-models.

If a meta-relationship is included in the diagram for clarification, but it is not used in the subject area, the text of the meta-relationship name shall be back-shaded. This is shown for the meta-relationship *D* in Figure 3.

If a supertype meta-entity that participates in a meta-relationship is not included in the diagram, the inherited meta-relationship shall be shown on the diagram as a broken line to/from the inheriting subtype meta-entity. This is shown for the meta-relationship *E* in Figure 3.

Subtype meta-relationships are shown with the meta-relationship name in italics. This is shown for the meta-relationship *F* in Figure 3.

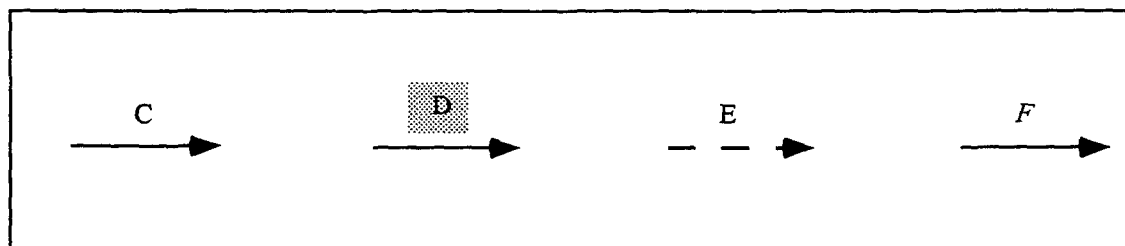


Figure 3
Conventions for Representing Meta-relationships

Full details of the CDIF Graphical Notation used in the Meta-model and the Meta-meta-model can be found in *EIA/IS-107 CDIF - Framework for Modeling and Extensibility*.

To aid in the understanding of some of the concepts modeled in this subject area, instance diagrams have been used. These diagrams differ from those described above in that an instance diagram depicts one or more occurrences or instances of meta-entities and meta-relationships depicted in the diagrams previously described. Where one of the diagrams previously described

may show that the meta-model includes an *Automobile* meta-entity, an instance diagram might depict *Honda*, *Ford*, *Mercedes*, and *Fiat* as instances of *Automobile*.

In an instance diagram, instances of meta-entities are depicted as boxes with a horizontal divider. Above the divider is the name of the meta-entity that is instantiated. Below the divider may be a name or a value assigned to a significant meta-attribute of that meta-entity. Where the name is inherited, inferred, or otherwise not significant to the example (e.g., some associative meta-entity instances), no name or value may be present.

Meta-relationships are depicted as arrows, just as in the previous diagrams described. They do not, however, have cardinality or exclusivity indicators, since none are necessary.

Meta-attributes of both meta-entities and meta-relationships are represented as:

[meta-attribute name :] *value*

where the name of the meta-attribute may be omitted depending on the ease of understanding the example with or without the name (e.g., *Name* might be omitted, *MinLength* might not).

When an instance of a meta-entity or meta-relationship not in this Subject Area is included for clarity, the box/arrow shall be shaded just as explained previously.

Figure 4 shows both the conceptual diagram and an instance diagram.

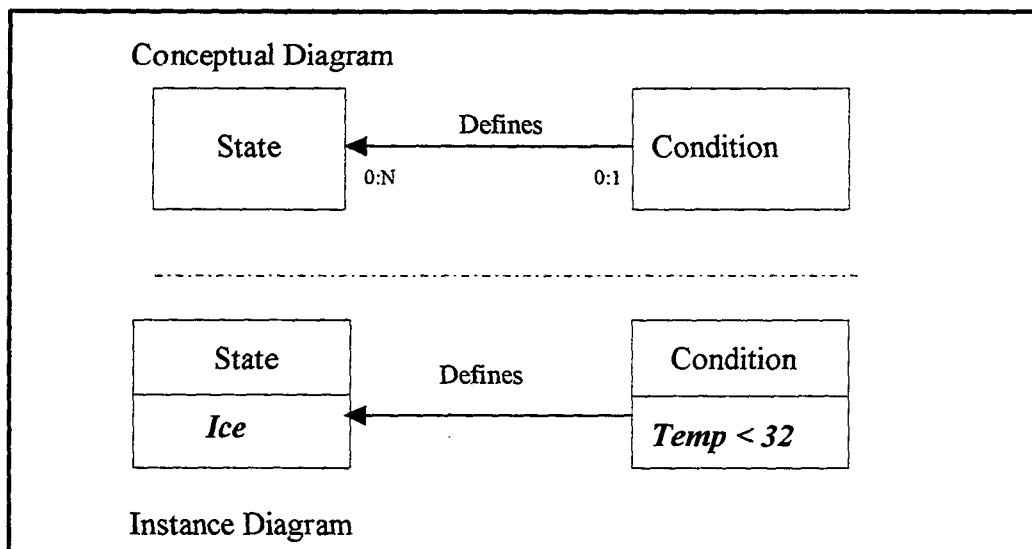


Figure 4
 Conceptual Diagram and an Instance Diagram

3.2.4 AttributableMetaObject Hierarchy

The AttributableMetaObject Hierarchy in the Subject Area Overview section contains all the meta-entities and meta-relationships used in the subject area. It also identifies all supertypes required to relate the meta-objects used back to *RootObject*. Other subtypes of objects shown in

the hierarchy, where neither they nor any of their subtypes are used in this subject area, are not shown.

The `AttributableMetaObject` Hierarchy is presented as an indented list of meta-object names. The indentation indicates the subtyping; all names indented below a given name are subtypes of that meta-object. For example, if *Car* and *Truck* are both subtypes of *Vehicle*, this would be shown as follows:

```
Vehicle
  Car
  Truck
```

A meta-entity or meta-relationship and all its subtypes can be duplicated in the hierarchy due to multiple inheritance. Where this occurs, the duplicate entries are shown in *italics*.

The overall `AttributableMetaObject` supertype, *RootObject*, defined in *EIA/IS-111 CDIF - Integrated Meta-model - Foundation Subject Area*, is shown at the top of the hierarchy.

Where a page break occurs in the list, all the supertypes of the first meta-entity or meta-relationship on the new page are repeated for clarity with a shaded background.

3.2.5 MetaEntity and MetaRelationship Summaries

Summaries are given of all the meta-entities and meta-relationships used in the Subject Area. The meta-entity definitions are given first, in alphabetical order, followed by the meta-relationship definitions, also in alphabetical order.

For each meta-entity used in the subject area, all the meta-attributes, both inherited and local, are listed. The inherited meta-attributes are shown first, in *italics*, sequenced alphabetically within parent meta-entity. The local meta-attributes are then listed alphabetically in normal font. The optionality, whether mandatory or optional, of each meta-attribute is given.

For each meta-relationship all the meta-attributes are listed sequenced alphabetically. Inherited and local meta-attributes are shown in the same way as for meta-entities. The optionality of each meta-attribute is given.

3.3 Structure of Subject Area Detailed Definitions

3.3.1 Summary

The Subject Area Detailed Definitions section consists of the following sub-sections:

- Subject Area Definition
- MetaEntity Definitions
- MetaRelationship Definitions

3.3.2 Subject Area Definition

This is the formal definition of the instance of the meta-entity called *SubjectArea* (SA). It contains the information listed in Table 1.

Table 1
Contents of Subject Area Definition

5.2 SubjectArea Definition	
NAME	subject area name
CDIFMETAIDENTIFIER	
DESCRIPTION	
USAGE	
ALIASES	
CONSTRAINTS	
VERSIONNUMBER	

3.3.3 MetaEntity Definitions

The definition of each meta-entity used in the subject area is structured according to the classes of information listed inTable 2.

For each meta-entity:

- Inherited meta-attributes are listed by supertype meta-entity in a top-down hierarchical order. Within each sub-list, meta-attributes are listed alphabetically.
- Inherited meta-relationships are grouped by source meta-entity (listed in alphabetical order). Within each group, the inherited meta-relationships are sorted alphabetically by "verb" name and then alphabetically by destination meta-entity name. Inherited abstract meta-relationships are not listed. Local meta-attributes are listed alphabetically.
- Detailed local meta-attribute descriptions are listed alphabetically. These definitions are structured according to the classes of information listed in Table 2.

Table 2
Contents of Meta-entity Definition

5.N.N	Meta-entity Name
NAME	meta-entity name
CDIFMETAIDENTIFIER	
SUBTYPEOF	List of direct supertypes of the meta-entity... meta-entity name [not] in this SA
SUPERTYPEOF	List of direct subtypes of the meta-entity within this subject area... meta-entity name
DESCRIPTION	
USAGE	
ALIASES	
CONSTRAINTS	
TYPE	
INHERITED META-ATTRIBUTES	
List of meta-attributes inherited from all supertypes of the meta-entity, with the name of the meta-object for which the meta-attribute is defined...	
meta-attribute namefrom meta-entity name [not] in this SA	
LOCAL META-ATTRIBUTES	
List of meta-attributes defined for the meta-entity...	
meta-attribute name	
INHERITED META-RELATIONSHIPS	
List of instantiable meta-relationships inherited from all supertypes of the meta-entity...	
meta-relationship name [not] in this SA	
LOCAL META-RELATIONSHIPS	
List of meta-relationships defined for the meta-entity...	
meta-relationship name	
Full definitions of all local meta-attributes appear at this point. See Table 3 for format of local meta-attribute definitions.	

Table 3
Contents of Local Meta-attribute Definition

META-ATTRIBUTE NAME..... meta-attribute name
CDIFMETAIDENTIFIER
DESCRIPTION
USAGE
ALIASES
CONSTRAINTS
DATA TYPE
DOMAIN
LENGTH
ISOPTIONAL

3.3.4 MetaRelationship Definitions

The definition of each meta-relationship used in the subject area is structured according to the classes of information listed in Table 4.

For each meta-relationship:

- Inherited meta-attributes are listed by supertype meta-relationship in a top-down hierarchical order. Within each sub-list, meta-attributes are listed alphabetically.
- Local meta-attributes are listed alphabetically.
- Detailed local meta-attribute descriptions are listed alphabetically.

Table 4
Contents of Meta-relationship Definition

5.N.N	Full Meta-relationship Name
NAME	meta-relationship name
CDIFMETAIDENTIFIER	
SUBTYPEOF	List of direct supertypes of the meta-relationship... meta-relationship name[not] in this SA
SUPERTYPEOF	List of direct subtypes of the meta-relationship within this subject area... meta-relationship name [not] in this SA
MINSOURCECARD	Minimum source meta-entity cardinality
MAXSOURCECARD	Maximum source meta-entity cardinality
MINDESTCARD	Minimum destination meta-entity cardinality
MAXDESTCARD	Maximum destination meta-entity cardinality
DESCRIPTION	
USAGE	
ALIASES	
CONSTRAINTS	
INHERITED META-ATTRIBUTES	
List of meta-attributes inherited from all supertypes of the meta-relationship, with the name of the meta-object for which the meta-attribute is defined...	
meta-attribute name	from meta-relationship name [not] in this SA
LOCAL META-ATTRIBUTES	
List of meta-attributes defined for the meta-relationship...	
meta-attribute name	
Full definitions of all local meta-attributes appear at this point. See Table 3 for format of local meta-attribute definitions.	

3.4 Structure of Referenced Meta-object Definitions

This section contains the formal definitions of meta-objects referenced from the definitions of meta-objects used in the subject area.

These will include the *AttributableMetaObject RootObject*, since this is the root of the *AttributableMetaObject* Hierarchy, and it may also include other meta-entities and meta-relationships.

The definitions are given in the following order: *AttributableMetaObject*, meta-entities, then meta-relationships. Within each section, they are listed in alphabetical order. The definition layout for the *AttributableMetaObject* is the same as that of the meta-entity; both meta-entity and meta-

relationship definitions follow the same layout as that defined for the Subject Area Detailed Definitions section.

4. SUBJECT AREA OVERVIEW

4.1 Description

4.1.1 Introduction

A state/event model depicts the dynamic behavior of a system, sub-system, or an entity in the real world which reacts discretely on discrete events. It models the ways in which these systems or entities interact, function, respond, or perform by themselves or in conjunction with others. They effectively represent the life cycle for a system, object or entity. A state/event model is sometimes also referred to as reactive or dynamic system.

Several common means of showing this dynamic behavior are state transition diagrams, object behavior diagrams, dynamic process model diagrams, and entity life history diagrams. The semantics of all of these are addressed by the state/event subject area. The presentation of these diagrams can be represented using the presentation subject areas within the CDIF Integrated Meta-model.

State transition diagrams typically depict the perceived states, allowed transitions between states, and the events, conditions, and actions associated with these transitions. Events may occur, or conditions may become true, which cause a transition from one state to another. Actions may occur as a result of an event or a state transition, either immediately or after a certain time.

Events and conditions may be simple or structured. Events or conditions may also be projections or derivations of other semantic information.

Using variables in a state/event model is supported by providing the concept of an *Attribute* which may have any data type. The value of attributes may be changed, calculated, and/or compared to other values as a part of an action resulting from an event or state transition.

As their name implies, Entity Life History diagrams depict the life history of an entity from creation to destruction. Events occur which cause one or more entities to change their internal state in some way. The actions that occur as a result of events are referred to as operations within the entity life history technique.

This subject area is technique independent but offers broad coverage, striving for a superset of the concepts represented in various techniques. Rules and constraints enforced by specific state/event modeling techniques are not included, since CDIF subject areas must be capable of supporting all the different variants, and the interchange of data (without technique-specific validation) among these variants.

The following sub-sections explain the major concepts supported in this subject area in more detail.

4.1.2 State/Event Model

A state/event model is the root container context for a collection of states, transitions, events, conditions, attributes, and actions. The *StateEventModel* meta-entity, as defined in this subject area, identifies the definition of the entire set of components in a model. This state/event model can be a system, subsystem or unit.

The definitions of states, attributes, and actions may be shared. Therefore different state/event models may share some of the same definitions for these components or even be sub-models within a larger system model.

Figure 5 shows an example of a state decomposition hierarchy in connection with state/event models in a CDIF transfer. States are defined by state definitions which may be structured. State definitions may be declared state/event models.

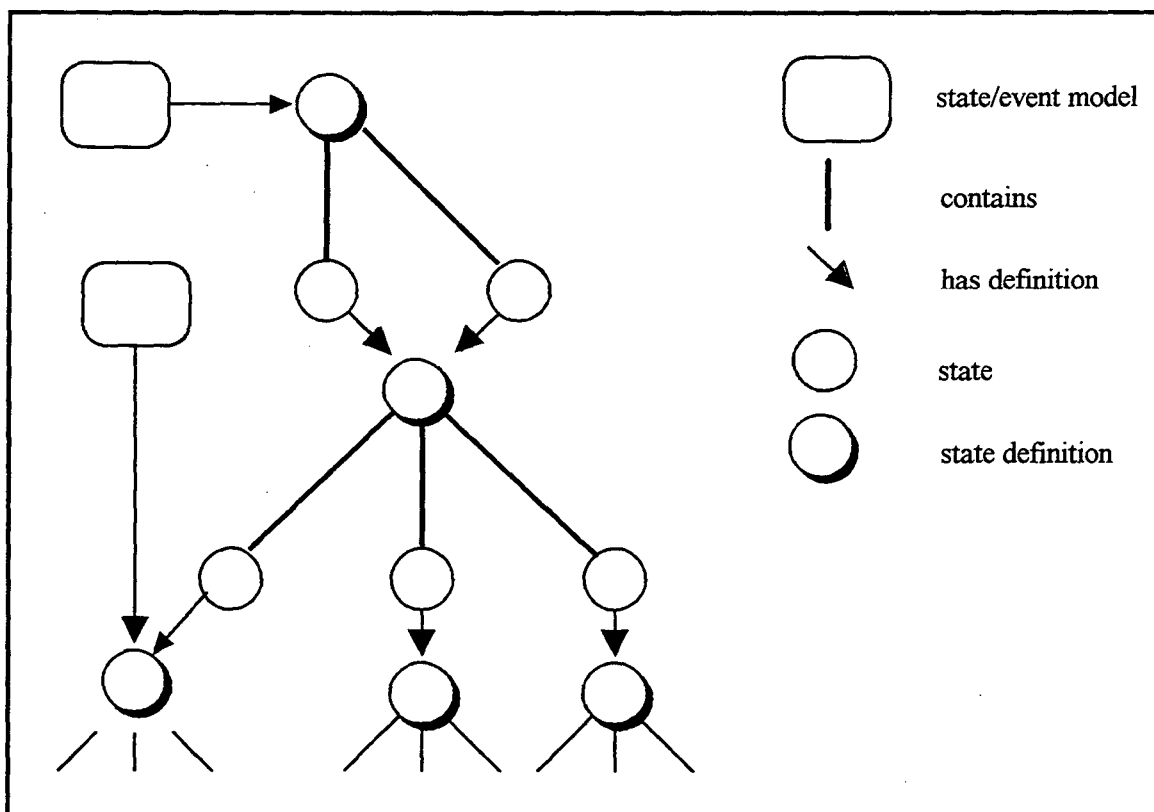


Figure 5
State decomposition and hierarchy of state machines

4.1.3 States

The State/Event Model Subject Area supports the concept of a state. The *State* meta-entity identifies the specific states a system, object, or entity can be in. The occurrence of certain events or changes to conditions will trigger transitions and change the state of the system, object, or entity.

States may be structured (sometimes called state machines); a state at a certain level of abstraction may be refined into a number of lower-level states, transitions, attributes, events and conditions. The higher-level transitions and attributes are bound to the interface the lower level state machine exposes.

If a structured state is entered, at least one of its substates is also entered. For the case where the structured state contains sequentially executing substates, the first (or next if the sequence was interrupted) substate is entered. For the case of concurrently executing substates, all substates are entered. For the case of alternate substates, exactly one substate is entered. When a structured state is active, it is not possible that none of its substates is active.

4.1.4 Transitions

Transitions represent potential traces of state changes between states. At any instant of time, a transition may or may not be taken in a state/event model.

Transitions can be triggered by the occurrence of an event, such as when a condition becomes true or a set time occurs. For a transition to occur, the system, entity or object must be in a state from which the transition can be taken.

As a result of a transition being taken, an action may be performed.

This subject area does not support the concept of decomposing transitions. It does, however, support the concept of splitting and merging transitions.

4.1.5 Action

An action is a procedure that may be performed as a result of an event occurring, such as a condition becoming true, or a transition being taken. *Actions* usually act on *Attributes*. For example, the procedure "assign 5 to Attribute X" could be an *Action*.

Actions may be elemental (primitive) or structured. This subject area includes a number of primitive actions, such as assignment actions.

Structured actions contain other actions which may be performed in sequence, in parallel, or as alternates.

4.1.6 Events and Conditions

At any point in time, a certain event may or may not occur. For example, at any point in time, the event "temperature is crossing the 0 degree mark" may happen.

On the contrary, conditions represent boolean values which persist over time. For example, "temperature is higher than 0 degrees" would be a condition.

Both concepts are supported in this subject area. Also, the concept of a buffered event is supported. A buffered event is an event whose occurrence is stored until it can be processed. For example, in a telecommunication system, the event "customer pressed the 1 key" may be stored.

4.1.7 The General Structuring Mechanism

4.1.7.1 Introduction

Support for the decomposition and structuring of objects into other objects, and the reuse of definitions, is provided by a general structuring mechanism. This mechanism is also used in other CDIF subject areas.

In this subject area the following concepts may be structured: *States*, *Actions*, *Attributes*, *Events* and *Conditions*. Structuring of those concepts is performed using the general structuring mechanism because their definitions may be used by several objects in different contexts.

4.1.7.2 Meta-entities and Meta-relationships

The general structuring mechanism makes use of the following meta-entities:

- *ComponentObject*
- *DefinitionObject*
- *ReferencedElement*
- *EquivalenceSet*

and the following meta-relationships:

- *DefinitionObject.Contains.ComponentObject*
- *ComponentObject.References.DefinitionObject*
- *ReferencedElement.DefinesPath.ComponentObject*
- *EquivalenceSet.HasMember.ComponentObject*

Figure 6 shows the meta-model for the general structuring mechanism.

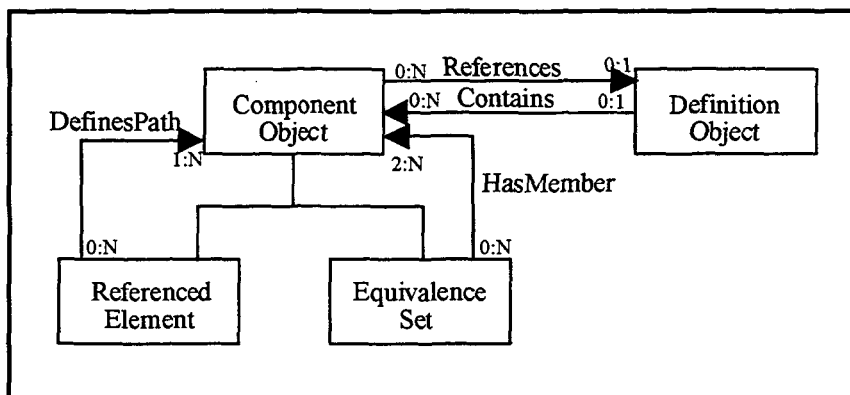


Figure 6
General Structuring Mechanism

4.1.7.3 DefinitionObject and ComponentObject

DefinitionObject serves as an abstract supertype for all decomposition and structure definitions that may be reused. A *DefinitionObject* represents the definition of a *ComponentObject*, using meta-relationship *ComponentObject.References.DefinitionObject*. A *DefinitionObject* may contain *ComponentObjects*, using the *DefinitionObject.Contains.ComponentObject* meta-relationship. The *ComponentObjects* contained in a *DefinitionObject* represent the components of the definition.

Both *ComponentObject* and *DefinitionObject* represent abstract concepts and thus shall not be instantiated. Instead, appropriate subtypes, provided by this or other subject areas, are used.

A subtype of *DefinitionObject* may be referenced by any number of *ComponentObjects* indicating that all those *ComponentObjects* share the same definition.

Instances of the subtypes of *ComponentObject* are used to describe the structure of a *DefinitionObject*. There may be any number of *ComponentObjects* per *DefinitionObject*.

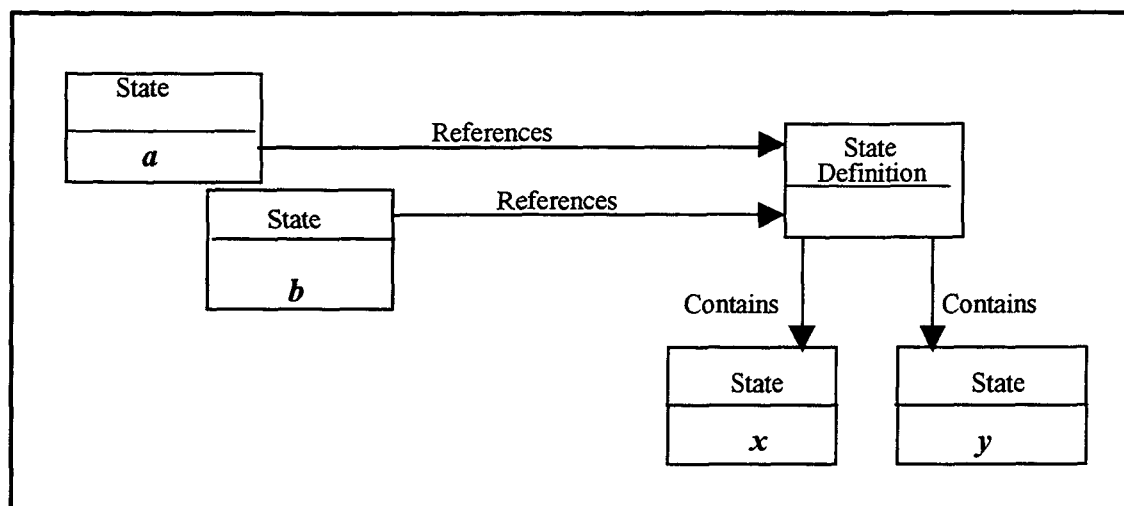


Figure 7
States sharing the same structured definition

For example, if a structure is built containing two components that share the same definition, the structured *DefinitionObject* will contain two distinct *ComponentObjects* which refer to the same shared *DefinitionObject*.

In this subject area, *DefinitionObjects* may contain one or more instances of one or more *ComponentObjects*. The allowed components, in addition to *ReferencedElement* and *EquivalenceSet*, for a given *DefinitionObject* are shown in Table 5. For example, a *State* may reference a *StateDefinition* that contains *States* and *Transitions*. This represents the fact that the *State* is refined into lower-level states and transitions through its definition. The *StateDefinition* may be shared by another *State* which also refers to this *StateDefinition* using *ComponentObject.References.DefinitionObject*.

Table 5
Allowable *DefinitionObject* components

DefinitionObject	ComponentObject
StateDefinition	Action Attribute Condition Event State Transition TransitionPort
ActionDefinition	Action Attribute Condition Event
ConditionDataType	Condition
EventDataType	Event

The general structuring mechanism ensures that, for example, the instances of the components of one *State* are different from those of another *State* even if both *States* share the same *StateDefinition*. This applies equally for all *DefinitionObjects*. It can also be used to specify, using the *EquivalenceSet* meta-entity, that two components in different reuse situations are actually the same instance (see 4.1.11 Identical Instances in Shared Definitions).

4.1.8 State Decomposition and Definition

States are represented using the meta-entity *States* that is a subtype of *ComponentObject*. States definitions are represented using meta-entity *StateDefinition*, a subtype of *DefinitionObject*.

Figure 8 shows an example of a state decomposition. *State P* decomposes into *Q*, *R* and *S*. *R* and *S* share the same decomposition into *T* and *U*. The general structuring mechanism requires that each process refers to a process definition. *States* cannot contain *States* or any other *ComponentObject* directly, the *StateDefinition* contains the *ComponentObjects*. Sharing definitions does not imply anything about the instances. For example, the *States T* in the decomposition of *R* shares only its structure with *S*, not its data or state.

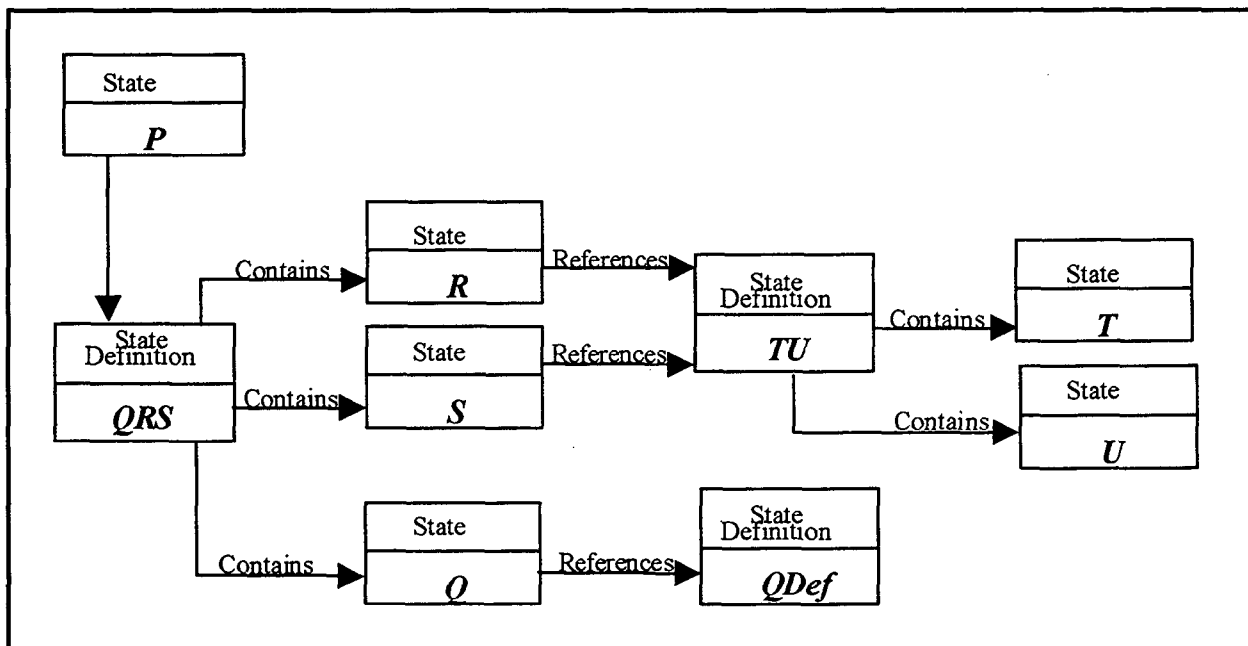


Figure 8
A states decomposition

For the definition of leaf states (i.e., states which are not decomposed into other states) an instance of the meta-entity *StateDefinition* is used which does not contain any *States*, *Attributes*, *Actions* or *Transitions*.

4.1.9 Action Decomposition and Definition

Analogous to the structuring of states, *Actions* are also structured using the general structuring mechanism. An *ActionDefinition* represents the potential structure of a *Action*. An *Action* may be a structured *Action* containing other *Actions* as members or completely unstructured.

Figure 9 shows an example for a structured *Action* where *Action f1* and *f2* share the same structured *ActionDefinition*. This *ActionDefinition* indicates that both *f1* and *f2* each contain an *Action x*, an *Action y* and a *Action z*. Each *Action* has an *ActionDefinition*. *Actions x* and *y* share an *ActionDefinition* called *action-def2*.

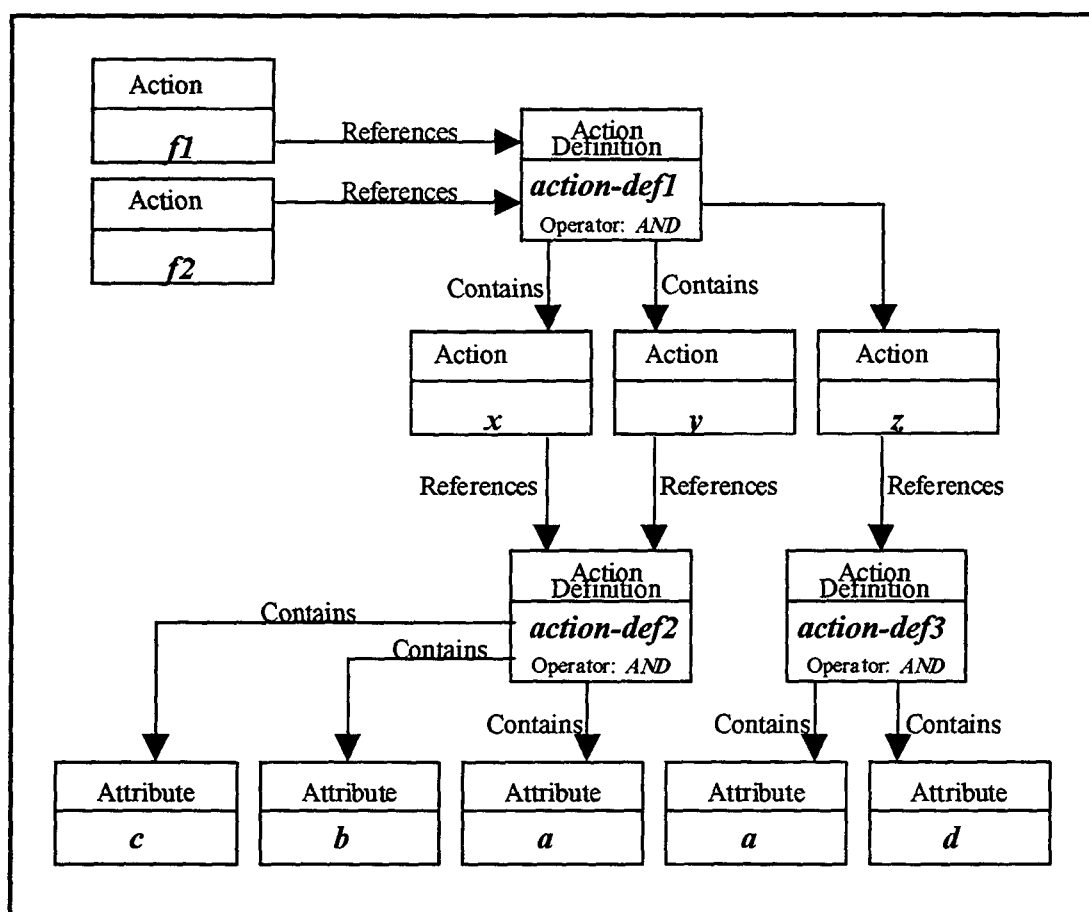


Figure 9
 Structured Actions

The sharing of *ActionDefinitions* does not imply anything about the data on the *Actions* which share the same definitions. In this case, for example, the *Attributes f1.x.a* (*Attribute a* on subaction *x* of action *f1*), *f1.y.a* and *f2.z.a* are completely unrelated. If the values of the *Attributes f1.x.a* and *f1.y.a* are the same, an *EquivalenceSet* with two *ReferencedElements* would be used. This *EquivalenceSet* would be contained in *ActionDefinition action-def1*. (See section 4.1.11, Identical Instances in Shared Definitions.)