Interim Report

of the

Components 1.1 Revision Task Force

to the

Platform Technical Committee

of the

Object Management Group

September 9, 2002

Document Numbers:
ptc/2002-08-02 (this document)
ptc/2002-08-03 (edited CCM chapters)

Table of Contents

Summary of Components 1.1 RTF Activities

Formation

- Chartered By: Platform Technical Committee
- Meeting Location: Yokohama, Japan, April 22 - 26, 2002
- Voting List Deadline Date: April 26, 2002
- Interim Comments Deadline Date: July 1, 2002
- Interim Report Due Date: October 7, 2002
- Final Comments Due Date: February 3, 2003
- Final Report and Revision Due Date: April 14, 2003

Revision Task Force Membership:

Member	Organization	Status
Julien Maisonneuve	Alcatel	Charter
J. Scott Evans	Computational Physics, Inc.	Charter
Tom Ritter	Fraunhofer FOKUS	Charter
Garry Turkington	Government Communications Headquarters (GCHQ)	Charter
Jishnu Mukerji	Hewlett-Packard	Charter
Harald Böhme	Humboldt Universitaet	Charter
Nawel Sabri	INRIA	Charter
Philippe Merle	LIFL	Charter - CHAIR
Jim Kulp	Mercury Computer Systems	Charter
Rudolf Schreiner	Object Security Ltd.	Charter
Hakim Souami	THALES	Charter
Diego Sevilla Ruiz	Universidad de Murcia	Charter

Shahzad Aslam-Mir	Vertel Corporation	Charter

Initial Issues from Architecture Board Review:

- None to date.

Issue Disposition:

Disposition Value	Number of Occurrences	Meaning of Disposition
Accepted	21	Change made as requested
Unresolved	24	RTF couldn't agree on a resolution
Rejected	0	RTF agreed NOT to do what was requested, or anything like it
Duplicate	0	Duplicate to another issue (“n” is OMG Issue Number in “Referenced Issue” field.)

Profile of Changes Made in Response to Issues

The Components 1.1 RTF made changes that:

- Fixed editorial issues (issues 5493, 5494, 5395, 5497, 5506, 5583, 5585),
- Fixed two generic component operations (chapter 1 – issue 4983),
- Fixed the CIDL grammar (chapter 2 – issues 5498, 5499, 5500),
- Fixed CIF language mapping (chapter 3 – issue 5340),
- Fixed XML DTDs (chapters 6 and 7 – issues 5091, 5092, 5093, 5429, 5492, 5496, 5584),
- Fixed component deployment (chapter 6 – issue 5577),
- Fixed the Notification Service IDL (issue 4986).

The following is a table that categorizes the issues as to the degree of changes that were made in resolving them.

Extent of IDL Change	**Number of Issues**	**OMG Issue No. (e.g. 2040)**
Significant Changes	2	5092, 5499
Minor Changes	9	4983, 5091, 5093, 5429, 5496, 5498, 5500, 5577, 5584
Support Text Changes	10	4986, 5340, 5492, 5493, 5494, 5495, 5497, 5506, 5583, 5585

Voting Results

Company	Vote	OMG Issue No
Alcatel	Yes	4983, 4986, 5091, 5092, 5093, 5340, 5429, 5492, 5493, 5494, 5495, 5496, 5497, 5498, 5499, 5500, 5506, 5577, 5583, 5584, 5585
	No	
	Abstain	
Computational Physics, Inc.	Yes	4983, 4986, 5091, 5092, 5093, 5340, 5429, 5492, 5493, 5494, 5495, 5496, 5497, 5498, 5499, 5500, 5506, 5577, 5583, 5584, 5585
	No	
	Abstain	
Fraunhofer FOKUS	Yes	4983, 4986, 5091, 5092, 5093, 5340, 5429, 5492, 5493, 5494, 5495, 5496, 5497, 5498, 5499, 5500, 5506, 5577, 5583, 5584, 5585
	No	
	Abstain	
Government Communications Headquarters (GCHQ)	Yes	
	No	
	Abstain	
Hewlett-Packard	Yes	
	No	
	Abstain	4983, 4986, 5091, 5092, 5093, 5340, 5429, 5492, 5493, 5494, 5495, 5496, 5497, 5498, 5499, 5500, 5506, 5577, 5583, 5584, 5585
Humboldt Universitaet	Yes	4983, 4986, 5091, 5092, 5093, 5340, 5429, 5492, 5493, 5494, 5495, 5496, 5497, 5498, 5499, 5500, 5506, 5577, 5583, 5584, 5585
	No	
	Abstain	
INRIA	Yes	4983, 4986, 5091, 5092, 5093, 5340, 5429, 5492, 5493, 5494, 5495, 5496, 5497, 5498, 5499, 5500, 5506, 5577, 5583, 5584, 5585
	No	
	Abstain	
LIFL	Yes	4983, 4986, 5091, 5092, 5093, 5340, 5429, 5492, 5493, 5494, 5495, 5496, 5497, 5498, 5499, 5500, 5506, 5577, 5583, 5584, 5585
	No	
	Abstain	
Mercury	Yes	4983, 4986, 5091, 5092, 5093, 5340, 5429, 5492,

Computer Systems		5493, 5494, 5495, 5496, 5497, 5498, 5499, 5500, 5506, 5577, 5583, 5584, 5585
	No	
	Abstain	
Object Security Ltd.	Yes	4983, 4986, 5091, 5092, 5093, 5340, 5429, 5492, 5493, 5494, 5495, 5496, 5497, 5498, 5499, 5500, 5506, 5577, 5583, 5584, 5585
	No	
	Abstain	
THALES	Yes	4983, 4986, 5091, 5092, 5093, 5340, 5429, 5492, 5493, 5494, 5495, 5496, 5497, 5498, 5499, 5500, 5506, 5577, 5583, 5584, 5585
	No	
	Abstain	
Universidad de Murcia	Yes	4983, 4986, 5091, 5092, 5093, 5340, 5429, 5492, 5493, 5494, 5495, 5496, 5497, 5498, 5499, 5500, 5506, 5577, 5583, 5584, 5585
	No	
	Abstain	
Vertel Corporation	Yes	4983, 4986, 5091, 5092, 5093, 5340, 5429, 5492, 5493, 5494, 5495, 5496, 5497, 5498, 5499, 5500, 5506, 5577, 5583, 5584, 5585
	No	
	Abstain	

Voting History

Vote	Date	Non-Voting Companies	OMG Issue No
Vote 1	September 2, 2002	GCHQ	4983, 4986, 5091, 5092, 5093, 5340, 5429, 5492, 5493, 5494, 5495, 5496, 5497, 5498, 5499, 5500, 5506, 5577, 5583, 5584, 5585

Disposition: Accepted

OMG Issue No: 4983

Title: Generic operations for subscribing/unsubscribing at publishing ports

Source:

Humboldt University of Berlin, Mr. Bertram Neubauer, neubauer@informatik.hu-berlin.de

Summary:

generic operations for subscribing and unsubscribing at publishing ports of components should have the same funtionality as the specific ones. Therefore the generic unsubscribe operation should return the unsubscribed consumer reference like the specific operation does. proposal: change

```
void unsubscribe (in FeatureName publisher_name, in
Cookie ck) raises (InvalidName, InvalidConnection);
```

to

```
EventConsumerBase unsubscribe (in FeatureName
publisher_name, in Cookie ck)  raises (InvalidName,
InvalidConnection);
```

Resolution:

In order to be consistent, the generic operation for unsubscribing to publisher ports of components must return the unsubscribed consumer reference like the specific generated operations do.

Moreover, the generic operation for disconnecting from receptacle ports of components must return the previously connected object reference like the specific generated operations do.

Revised Text:

In formal/02-06-65, page 1-29, replace

```
void unsubscribe ( . . . );
```

by

```
EventConsumerBase unsubscribe ( . . . );
```

In formal/02-06-65, page 1-30, add at the end of the first sentence of the **`unsubscribe`** description

, and returns the reference of the subscriber

In formal/02-06-65, page 1-19, replace

```
void disconnect ( . . . );
```

by

```
Object disconnect ( . . . );
```

In formal/02-06-65, page 1-20, add after the 3rd sentence of the **`disconnect`** description the following sentence

> In both cases, the **`disconnect`** operation returns the previously connected object reference.

Disposition: **Accepted**

Disposition Parameter: None

Disposition: Accepted

OMG Issue No: 4986

Title: IDL3 keyword "eventtype" conflicts with struct "CosNotification::EventType

Source:

Washington University, Mr. Nanbor Wang, nanbor@cs.wustl.edu

Summary:

The "eventtype" keyword defined in Section 3.2.4 of the "CORBA 3.0 New Components Chapters" (ptc/2001-11-03) will make the Notification Service IDL un-compilable because Notification Service defines a struct called "EventType" in Section 3.1 of formal/00-06-20.

I guess one of the specs will have to change.

Resolution:

The Notification Service IDL defined in formal/00-06-20 could not be compiled with a CORBA 3.0 IDL compiler because the identifier **`EventType`** is now interpreted as the **`eventtype`** keyword.

In OMG IDL, the only way to escape from the keyword interpretation is to put an underscore in front of identifiers. The generated code will remain the same.

Revised Text:

Change all occurrences of **`EventType`** by **`_EventType`** in the latest Notification Service Specification.

Disposition: **Accepted**

Disposition Parameter: None

Disposition: Unresolved

OMG Issue No: 5091

Title: repository id in software package descriptor

Source:

Alcatel, Mr. Frank Pilhofer, fp@fpx.de

Summary:

The id attribute of the idl element in a Software Package Descriptor currently contains the Repository Id of the component.

I suggest to change this to the Repository Id of the home, as the home implies the component, but not vice versa. This makes it easier for the deployment application to find out about the home interface, e.g. for the purpose of configuring properties.

Resolution:

The Repository Id of the home should be added as an attribute of the **idl** element.

Revised Text:

In formal/02-06-65, section 6.3.1, page 6-3, replace

```
<idl id="IDL:M1/Bank:1.0">
```

by

```
<idl id="IDL:M1/Bank:1.0"
homeid="IDL:M1/BankHome:1.0">
```

In formal/02-06-65, section 6.3.2.14, page 6-9, replace

> The **idl** element points to file or repository containing an idl definition.
>
> ```
> <!ELEMENT idl
>     ( link
>     | fileinarchive
>     | repository
>     ) >
> <!ATTLIST idl
>     id CDATA #REQUIRED >
> ```
>
> The **id** attribute is a repository Id which uniquely identifies the IDL equivalent interface for the software component.

by

The **idl** element points to a file or repository containing IDL definitions. If a component and its home are defined in different files or repositories, then the **idl** element refers to the file or repository where the home is defined.

```
<!ELEMENT idl
    ( link
    | fileinarchive
    | repository
    ) >
<!ATTLIST idl
    id CDATA #REQUIRED
    homeid CDATA #REQUIRED >
```

The **id** attribute is a repository Id that uniquely identifies the IDL equivalent interface for the software component. The **homeid** attribute is a repository Id that uniquely identifies the IDL equivalent interface for the home of the software component.

In formal/02-06-65, section 6.3.2.14, page 6-9, replace

```
<!ATTLIST idl
    id CDATA #REQUIRED >
```

by

```
<!ATTLIST idl
    id CDATA #REQUIRED
    homeid CDATA #REQUIRED >
```

Disposition: Accepted

Disposition Parameter: None

Disposition: Accepted

OMG Issue No: 5092

Title: registercomponent element

Source:

Alcatel, Mr. Frank Pilhofer, fp@fpx.de

Summary:

On page 69-532 (ptc/01-11-03), the registercomponent element of an Assembly Descriptor is defined to optionally contain an emitsidentifier and publishes identifier element, in order to register an event source in the Naming or Trading Service. There is also a note saying, "In the case of events, what gets registered?"

I suggest to replace the emitsidentifier and publishesidentifier elements with the consumesidentifier element, and to change the first two paragraphs to read:

"The registercomponent element is to specify that a component, a provided interface, or an event consumer interface should be registered with a naming service or trader.

If a consumesidentifier or publishesidentifier is specified, then that element is registered. If none of the above are specified, then it is implied that the component itself is to be registered."

Resolution:

Emitter or publisher ports could not be registered with the Naming or Trader services because these ports have no associated object references. In fact, only components, facets, and event consumers have an associated object reference and could be registered with these two services. Then the description of the **`registercomponent`** element must be updated to be consistent.

Moreover, registering an object reference with the Trader Service requires to provide a trader service type name and a set of trader properties. This is reflected by the **`traderexport`** element. But the **`registerwithtrader`** only contains a set of trader properties and no trader service type name. Then the description of the **`registerwithtrader`** must be updated to be consistent.

Revised Text:

In formal/02-06-65, page 6-57, replace

The **`registercomponent`** element is used to specify that a component, a provided interface, or a published event should be registered with a naming service or trader.

If an **`emitsidentifier`**, **`providesidentifier`**, or **`publishesidentifier`** is specified, then that element is registered. If none of the above are specified, then it is implied that the component itself is to be registered.

by

The **`registercomponent`** element is used to specify that a component, a provided interface, or an event consumer interface should be registered with a naming service or trader.

If a **`providesidentifier`** or **`consumesidentifier`** is specified, then that element is registered. If none of the above are specified, then it is implied that the component itself is to be registered.

In formal/02-06-65, page 6-57 and page 7-18, replace

```
<!ELEMENT registercomponent
     ( ( emitsidentifier
       | providesidentifier
       | publishesidentifier
     )?
     , ( registerwithnaming
       | registerwithtrader
     )+
)>
```

by

```
<!ELEMENT registercomponent
     ( ( providesidentifier
       | consumesidentifier
     )?
     , ( registerwithnaming
       | registerwithtrader
     )+
)>
```

In formal/02-06-65, page 6-58 and 7-18, replace

```
<!ELEMENT registerwithtrader
( traderproperties ) >
```

by

```
<!ELEMENT registerwithtrader
```

```
     ( traderexport ) >
```

In formal/02-06-65, page 6-42, replace

```
     <registerwithtrader>
       <traderproperties>
         . . .
       </traderproperties>
     </registerwithtrader>
```

by

```
     <registerwithtrader>
       <traderexport>
         <traderservicetypename>
           aTraderServiceTypeName
         </traderservicetypename>
         <traderproperties>
           . . .
         </traderproperties>
       </traderexport>
     </registerwithtrader>
```

Disposition: **Accepted**

Disposition Parameter: None

Disposition: Accepted

OMG Issue No: 5093

Title: connectevent element

Source:

Alcatel, Mr. Frank Pilhofer, fp@fpx.de

Summary:

On page 69-521 (ptc/01-11-03), the connectevent element of an Assembly Descriptor is defined as

```
<!ELEMENT connectevent (consumesport,
                         (emitsport |
                          publishesport))>
```

This does not allow for emits and publishes ports to be connected to existing consumer interfaces that are registered in the naming or trading service. I suggest to change that element in the spirit of connectinterface to

```
<!ELEMENT connectevent (emitsport |
                         publishesport) ,
                        (consumesport |
                         existinginterface)>
```

Resolution:

Apply the proposed change allowing for emits and publishes ports to be connected to existing consumer interfaces.

However, do not reverse consumesport and emits/publishesport to be compatible with already existing and used XML descriptors.

Revised Text:

In formal/02-06-65, section 6.7.2.11, first paragraph, page 6-47, add after “of one component,”

or an existing interface

In formal/02-06-65, page 6-47 and page 7-14, replace

```
<!ELEMENT connectevent
     ( consumesport
     , ( emitsport
       | publishesport
```

```
          )
        )>
```

by

```
<!ELEMENT connectevent
     ( ( consumesport
       | existinginterface
       )
     , ( emitsport
       | publishesport
       )
     )>
```

In formal/02-06-65, section 6.7.2.23, page 6-51, add at the end of the first sentence

, or to an *emits* or *publishes* port within a **`connectevent`** element

Disposition: **Accepted**

Disposition Parameter: None

Disposition: Accepted

OMG Issue No: 5340

Title: Issue regarding language mapping for keyless homes

Source:

Computational Physics, Inc., Mr. J. Scott Evans, evans@cpi.com

Summary:

In 01-11-03.pdf page 61-255

Given a base home definition with a primary key (H, T, K), and a derived home definition with no primary key (H', T'), such that H' is derived from H, then the definition of H' implicitly includes a primary key specification of type K, becoming (H', T', K). The implicit interface for H' shall have the form specified for an implicit interface of a home with primary key K and component type T'.

In same document section 615.3.3.6 Home I see no mention that a keyless home that inherits from a keyed home is implicitly a keyed home and thus the Home Implicit Executor Interface must be constructed for a keyed home using the key type declared for the base keyed home. Everyone agree?

Resolution:

A keyless home that inherits from a keyed home is implicitly a keyed home and thus the Home Implicit Executor interface must be constructed as a keyed home using the key type declared for the base keyed home.

Revised Text:

In formal/02-06-65, section 3.3.3.6, page 3-46, replace

> ***Implicit Executor Interface for Keyed Homes***
>
> For a keyed home <**home name**> with a key of <**key type**>,

by

> ***Implicit Executor Interface for Explicitly or Implicitly Keyed Homes***
>
> For a keyed home <**home name**> with a key of <**key type**> or a keyless home <**home name**> that derives from a keyed home with a key of <**key type**>,

Disposition: **Accepted**

Disposition Parameter: **None**

Disposition: Accepted

OMG Issue No: 5429

Title: simple type element of the property file issue

Source:

Computational Physics, Inc., Mr. J. Scott Evans, evans@cpi.com

Summary:

In document 01-11-03.pdf section 69.8.2.8 the simple type element of the property file descriptor excludes the types:

long long
unsigned long long
long double
wchar

Is there any reason why we can't add these types to the simple element?

Resolution:

Add long long, unsigned long long, long double, wchar, wstring, and fixed to the **`type`** attribute of the **`simple`** element in order to allow configuration of component and home attributes of these types.

Moreover remove the second occurrence of the **`short`** type.

Revised Text:

In formal/02-06-65, page 6-63 and page 7-12, replace

```
<!ATTLIST simple
     name CDATA #IMPLIED
     type ( boolean
          | char
          | double
          | float
          | short
          | long
          | objref
          | octet
          | short
          | string
          | ulong
          | ushort
          ) #REQUIRED >
```

by

```
<!ATTLIST simple
     name CDATA #IMPLIED
     type ( boolean
          | char
          | double
          | float
          | short
          | long
          | objref
          | octet
          | string
          | ulong
          | ushort
          | longlong
          | ulonglong
          | longdouble
          | wchar
          | wstring
          | fixed
          ) #REQUIRED >
```

Disposition: **Accepted**

Disposition Parameter: None

Disposition: Accepted

OMG Issue No: 5492

Title: Issues related to CCM's XML descriptors: chapter 69.4.4

Source:

Laboratoire d'Informatique Fondamentale de Lille, Dr. Philippe Merle, Philippe.Merle@lifl.fr

Summary:

chapter 69.4.4

the sample componentkind definition given as an entity with "process" lifetime seems unadapted as described in 69.4.5.49, we can give a best lifetime with "container"

Resolution:

“process” is not a possible value for the **`lifetime`** attribute of the **`servant`** element (see formal/02-06-65 page 6-37). Possible values are **`component`**, **`method`**, **`transaction`**, or **`container`**. Then the example must be updated in order to be coherent.

Revised Text:

In formal/02-06-65, page 6-18, replace

```
<servant lifetime=”process”/>
```

by

```
<servant lifetime=”container”/>
```

Disposition: **Accepted**

Disposition Parameter: None

Disposition: Accepted

OMG Issue No: 5493

Title: Issues related to CCM's XML descriptors: chapter 69.4.5.4

Source:

Laboratoire d'Informatique Fondamentale de Lille, Dr. Philippe Merle, Philippe.Merle@lifl.fr

Summary:

chapter 69.4.5.4

"The information obtained by (...) it allows component assembly tools to decide WHAT ports on a component are capable (...)"
it seems that the word "which" would be better than "what" in this sentence

Resolution:

Replace “what” by “which”.

Revised Text:

In formal/02-06-65, page 6-23, section 6.4.5.4, 3rd paragraph, replace “what” by “which”.

Disposition: **Accepted**

Disposition Parameter: None

Disposition: Accepted

OMG Issue No: 5494

Title: Issues related to CCM's XML descriptors: chapter 69.4.5.16

Source:

Laboratoire d'Informatique Fondamentale de Lille, Dr. Philippe Merle, Philippe.Merle@lifl.fr

Summary:

chapter 69.4.5.16 this eventpolicy element is said to be child element of corbacomponent in fact it can be child element of: consumes, emits, publishes but it's NOT a child element of corbacomponent

Resolution:

Correct the false sentence as **`eventpolicy`** is a child element of **`consumes`**, **`emits`**, **`publishes`** and not a child element of **`corbacomponent`**.

Revised Text:

In formal/02-06-65, page 6-26, replace

> Child element of **`corbacomponent`**.

by

> Child element of **`consumes`**, **`emits`**, **`publishes`**.

Disposition: **Accepted**

Disposition Parameter: None

Disposition: Accepted

OMG Issue No: 5495

Title: Issues related to CCM's XML descriptors: chapter 69.7.2.25

Source:

Laboratoire d'Informatique Fondamentale de Lille, Dr. Philippe Merle, Philippe.Merle@lifl.fr

Summary:

chapter 69.7.2.25 the given reference to the fileinarchive element is wrong (69.3.2.11); the right one is 69.3.2.12

Resolution:

Correct the false cross-reference.

Revised Text:

In formal/02-06-65, page 6-51, replace

See Section 6.3.2.11, “The extension Element,” on page 6-8.

by

See Section 6.3.2.12, “The fileinarchive Element,” on page 6-8.

Disposition: **Accepted**

Disposition Parameter: None

Disposition: Accepted

OMG Issue No: 5496

Title: Issues related to CCM's XML descriptors: chapter 69.7.2.38

Source:

Laboratoire d'Informatique Fondamentale de Lille, Dr. Philippe Merle, Philippe.Merle@lifl.fr

Summary:

chapter 69.7.2.38 the processcollocation element given in this chapter lacks a DESTINATION child element, right one:

```
<!ELEMENT processcollocation
     ( usagename?
     , impltype?
     , ( homeplacement
       | extension
       )+
       ,destination?
     )>
<!ATTLIST processcollocation
     id      ID      #IMPLIED
     cardinality CDATA "1">
```

Resolution:

As described in formal/02-06-65, page 6-49 and page 7-17, **`destination`** is a child element of **`processcollocation`**. Then the **`processcollocation`** element must be updated to be coherent.

Revised Text:

In formal/02-06-65, page 6-56, replace

```
<!ELEMENT processcollocation
     ( usagename?
     , impltype?
     , ( homeplacement
       | extension
       )+
     )>
```

by

```
<!ELEMENT processcollocation
     ( usagename?
     , impltype?
     , ( homeplacement
       | extension
       )+
     , destination?
     )>
```

Disposition: **Accepted**

Disposition Parameter: None

Disposition: Accepted

OMG Issue No: 5497

Title: Issues related to CCM's XML descriptors: chapter 695.4

Source:

Laboratoire d'Informatique Fondamentale de Lille, Dr. Philippe Merle, Philippe.Merle@lifl.fr

Summary:

chapter 695.4 the elements aren't in the alphabetical order proxyhome must be placed before publishesidentifier

Resolution:

Apply the alphabetical order on the **`proxyhome`** element defined in the chapters 6 and 7 of the formal/02-06-65 document.

Revised Text:

In formal/02-06-65, page 6-58, replace

```
<!ELEMENT remotehome
```

by

```
<!ELEMENT proxyhome
```

In formal/02-06-65, page 6-58, move the whole section 6.7.2.47 before section 6.7.2.41 page 6-57.

In formal/02-06-65, page 7-18, move the **`proxyhome`** element before the **`publishesidentifier`** element.

Disposition: Accepted

Disposition Parameter: None

Disposition: Accepted

OMG Issue No: 5498

Title: CIDL Grammar problems: Productions must be renumbered : 134 -> 1, ...

Source:

Laboratoire d'Informatique Fondamentale de Lille, Dr. Philippe Merle, Philippe.Merle@lifl.fr

Summary:

Productions must be renumbered : 134 -> 1, ...

In all productions, replace ...storage_home... by ...storagehome... and ...storage_type... by storagetype...

Resolution:

Require to update the numbering of CIDL productions in the chapter 2 of the Adopted CORBA Components Specification (formal/02-06-65).

Require to replace all occurrences of **`storage_home`** by **`storagehome`** in CIDL productions (chapter 2 in formal/02-06-65).

Moreover, there is no occurrence of **`storage_type`** in formal/02-06-65.

Revised Text:

In formal/02-06-65, page 2-3, replace **`storage_home`** by **`storagehome`** in production 2.

In the whole chapter 2 of formal/02-06-65, replace all occurrences of …**`storage_home`**… by …**`storagehome`**… in all CIDL productions.

In formal/02-06-65, page 2-5, section 2-4, replace **`(41)`**, **`(42)`** and **`(43)`** by **`(1)`**, **`(2)`** and **`(3)`**.

In formal/02-06-65, page 2-5, section 2-4, replace **`storage_home`** by **`storagehome`** in production 2.

In formal/02-06-65, page 2-5, section 2-5, replace **`(44)`**, **`(45)`** and **`(46)`** by **`(4)`**, **`(5)`** and **`(6)`**.

In formal/02-06-65, page 2-8, section 2-7, replace **`(51)`** and **`(52)`** by **`(11)`** and **`(12)`**.

In formal/02-06-65, page 2-9, section 2-8, replace **`(53)`** and **`(54)`** by **`(13)`** and **`(14)`**.

In formal/02-06-65, page 2-9, section 2-9, replace **`(55)`**, **`(56)`** and **`(57)`** by **`(15)`**, **`(16)`** and **`(17)`**.

In formal/02-06-65, page 2-10, section 2-10, replace **`(58)`** by **`(18)`**.

In formal/02-06-65, page 2-10, section 2-11, replace **`(59)`**, **`(60)`** and **`(61)`** by **`(19)`**, **`(20)`** and **`(21)`**.

In formal/02-06-65, page 2-10, section 2-12, replace **`(62)`** and **`(63)`** by **`(22)`** and **`(23)`**.

In formal/02-06-65, page 2-11, section 2-13, replace **`(64)`** by **`(24)`**.

In formal/02-06-65, page 2-12, section 2-14, replace **`(65)`** by **`(25)`**.

In formal/02-06-65, page 2-12, section 2-15, replace **`(66)`** to **`(70)`** by **`(26)`** to **`(30)`**.

In formal/02-06-65, page 2-13, section 2-16, replace **`(71)`** to **`(74)`** by **`(31)`**, **`(33)`**, **`(34)`**, and **`(35)`**.

In formal/02-06-65, page 2-15, section 2-17, replace **`(75)`** by **`(32)`**.

In formal/02-06-65, page 2-16, section 2-18, replace **`(76)`** and **`(77)`** by **`(36)`** and **`(37)`**.

In formal/02-06-65, page 2-16, section 2-19, replace **`(78)`**, **`(79)`** and **`(80)`** by **`(38)`**, **`(39)`** and **`(40)`**.

Disposition: Accepted

Disposition Parameter: None

Disposition: Accepted

OMG Issue No: 5499

Title: paragraph 60.2.1 : There is two mistakes in keywords

Source:

Laboratoire d'Informatique Fondamentale de Lille, Dr. Philippe Merle, Philippe.Merle@lifl.fr

Summary:

- Capital letters are not appropriated. Correct "storageHome" with "storagehome".
- The keyword "catalog" must be removed as it was removed from PSDL.

Resolution:

Replace **`storageHome`** by **`storagehome`** in Table 2-2, page 2-3, formal/02-06-65.

As the **`catalog`** keyword was removed from the Persistent State Definition Language, remove all references to the catalog concept in the Adopted CORBA Components Specification (formal/02-06-65).

Revised Text:

All the following changes must be applied to the formal/02-06-65 document.

At page 2-1, remove “Catalog Usage Declaration” from the table.

At page 2-3, remove the **`catalog`** keyword from table 2-2.

At page 2-3, table 2-2, replace **`storageHome`** by **`storagehome`**.

At page 2-4, production 6, remove “ **`[<catalog_use_dcl>]`** “

At page 2-4, remove productions 7, 8, 9, and 10.

At page 2-4, remove **`<catalog_label> “.”`** from production 16.

At page 2-6, production 6, remove “ **`[<catalog_use_dcl>]`** “

At page 2-6, remove the item

An optional catalog usage declaration,

At page 2-7 and 2-8, remove the section 2.6.

At page 2-9, remove **`<catalog_label> “.”`** from production 16.

At page 2-9, remove the 3rd paragraph in section 2.9.

At page 3-11, section 3.2.7, remove the first item

> • One or more catalogs that provide the storage homes to the composition implementation. Each specified catalog is assigned an alias, or label, that identifies the catalog within the context of the composition.

At page 3-12, replace

```
composition <category> <composition_name> {
        uses catalog {
                <catalog_type> <catalog_label>;
        };
        home executor <home_executor_name> {
                implements < home_type> ;
                bindsTo <catalog_label. abstract_storage_home>;
                manages <executor_name>;
        };
};
```

where the additional elements are as follows: <*catalog_type*> identifies the type of a catalog previously defined in PSDL, <*catalog_label*> is an alias by which the catalog can be identified in the composition definition, and <*catalog_label.abstract_storage_home*> denotes a particular abstract storage home provided by the catalog.

by

```
composition <category> <composition_name> {
        home executor <home_executor_name> {
                implements < home_type> ;
                bindsTo <abstract_storage_home>;
                manages <executor_name>;
        };
};
```

where the additional element is as follows: < *abstract_storage_home*> denotes a particular abstract storage home.

At page 3-13, in Figure 3-2, remove

```
uses catalog {
        <catalog_type> <catalog_label>;
};
```

At page 3-13, in Figure 3-2, replace

```
bindsTo <catalog_label.storage_home>;
```

by

> **bindsTo <storage_home>;**

At page 3-13, in Figure 3-2, remove the box named “catalog”, the arrow to this box, and the arrow to the box named “storage home”.

At page 3-16, replace

> *The CIDL now defines abstract storage types, abstract storage homes, and a catalog.*

by

> *The CIDL now defines an abstract storage type and an abstract storage home.*

At page 3-16, remove

> **catalog** ToonCatalog {
>   **provides** ToonStateHome TSHome;
> };

and

> **uses catalog** { ToonCatalog store; };

At page 3-16, replace

> **bindsTo** store.TSHome;

by

> **bindsTo** ToonStateHome;

At page 3-17, remove

> *The interface for the catalog ToonCatalog.*

At page 3-23, replace

> *The CIDL now defines abstract storage types, abstract storage homes, and a catalog.*

by

> *The CIDL now defines abstract storage types and abstract storage homes.*

At page 3-17, remove

> **uses catalog** { ToonCatalog store; };

At page 3-17, replace

> **bindsTo** store.TSHome;

by

> **bindsTo** ToonStateHome;

At page 3-27, replace

> **storedOn <catalog_label. abstract_storage_home>;**

by

> **storedOn <abstract_storage_home>;**

At page 3-30, replace

> *The CIDL now defines abstract storage types, abstract storage homes, and a catalog.*

by

> *The CIDL now defines abstract storage types and abstract storage homes.*

At page 3-30, remove

```
catalog ToonCatalog {
        provides ToonStateHome TSHome;
        provides BirdSegStateHome BSSHome;
};
```

and

> **uses Catalog** { ToonCatalog store; };

At page 3-30, replace

```
bindsTo store.TSHome;
. . .
        storedOn ToonPS.BSSHome;
```

by

```
bindsTo ToonStateHome;
. . .
        storedOn BirdSegStateHome;
```

At page 3-31, replace

> *The storage home BSSHome on the ToonCatalog catalog is bound to the . . .*

by

The storage home BirdSegStateHome is bound to the . . .

At page 3-31, replace

The mappings of the CIDL abstract storage types, abstract storage homes, and the catalog are not presented, . . .

by

The mappings of the CIDL abstract storage types and abstract storage homes are not presented, . . .

At page 3-35, replace

bindsTo <catalog_label.abstract_ storage_home>;

by

bindsTo <abstract_ storage_home>;

At page 6-18, remove

<catalog type="PSDL:BookCatalog:1.0" />

At pages 6-22 and 6-23, remove the whole section 6.4.5.3.

At page 6-25, section 6.4.5.10, remove the item

• **catalog** specifies the catalog type.

and in the XML DTD

, catalog?

At page 7-5, remove

```
<!ELEMENT catalog EMPTY>
<!ATTLIST catalog
type CDATA #REQUIRED >
```

At page 7-6, remove in the **`containermanagedpersistence`** element

```
, catalog?
```

At page Index-1, remove

catalog Element 6-22

Catalog usage declaration 2-7

Disposition: Accepted

Disposition Parameter: None

Disposition: Accepted

OMG Issue No: 5500

Title: Productions 140, 141, 142 and 143 must be removed

Source:

Laboratoire d'Informatique Fondamentale de Lille, Dr. Philippe Merle, Philippe.Merle@lifl.fr

Summary:

Productions 140, 141, 142 and 143 must be removed. Catalog has been removed from the PSS specification.

Production 145 : <home_executor_body> refers to <stored_on_dcl> which is defined in production 151 as <home_persistence_dcl>. Pick one or the other and changes references as required ...

Productions 149 and 150 are not valid any more. Remove them and add a new production :
<abstract_storage_home_name> ::= <identifier>
identifier must be the name of an abstract storagehome previously declared

Resolution:

As the **`catalog`** keyword was removed from the Persistent State Definition Language, remove all references to the catalog concept in the Adopted CORBA Components Specification (formal/02-06-65).

Replace **`stored_on_dcl`** by **`home_persistence_dcl`** in the CIDL production 12 in order to be consistent.

Remove the CIDL **`abstract_storage_home_name`** and **`abstract_storage_home_label`** productions as the PSDL already defines the **`abstract_storagehome_name`** production. Replace all occurrences of the **`abstract_storage_home_name`** production by the **`abstract_storagehome_name`** production.

Revised Text:

Apply revised text of issue 5499 in order to remove all references to the **`catalog`** concept.

In formal/02-06-65, page 2-4, replace **`stored_on_dcl`** by **`home_persistence_dcl`** in production 12.

In formal/02-06-65, page 2-4, remove CIDL productions 16 and 17.

In formal/02-06-65, page 2-8, replace **`stored_on_dcl`** by **`home_persistence_dcl`** in production 12.

In formal/02-06-65, page 2-9, remove CIDL productions 16 and 17.

Disposition: **Accepted**

Disposition Parameter: None

Disposition: Accepted

OMG Issue No: 5506

Title: Typo (??) in chapter 61

Source:

Laboratoire d'Informatique Fondamentale de Lille, Dr. Philippe Merle, Philippe.Merle@lifl.fr

Summary:

In the CCM specification document ptc/2001-11-03 page 61-227 the 'session' word should be removed from the OMG IDL 3.0 example, i.e.:

```
component baz session {
```

should be replaced by:

```
component baz {
```

Resolution:

Correct the typo by removing the '**session**' word.

Revised Text:

In formal/02-06-65, page 1-12, replace

```
component baz session {
```

by

```
component baz {
```

Disposition: **Accepted**

Disposition Parameter: None

Disposition: Unresolved

OMG Issue No: 5507

Title: 69.8.2 Property File XML Elements

Source:

Raytheon, Mr. Gerald Lee Bickle, Gerald_L_Bickle@raytheon.com

Summary:

1. 69.8.2.1 The properties Root Element, page 69-537

Properties Element Cardinality. Suggested change is to replace "*" with "+".

Why does it make sense to have an empty properties file, since the properties are optional references in the Software Package DTD, CORBA Component DTD, and Component Assembly DTD?

Current Format

```
<!ELEMENT properties
  ( description?
  , ( simple
    | sequence
    | struct
    | valuetype
    )*
  ) >
```

Suggested New Format

```
<!ELEMENT properties
  ( description?
  , ( simple
    | sequence
    | struct
    | valuetype
    )+
  ) >
```

Resolution:

None as this is deferred to the final report of the Components 1.1 RTF.

Revised Text:

None as this is deferred to the final report of the Components 1.1 RTF.

Disposition: Unresolved

Disposition Parameter: None

Disposition: Unresolved

OMG Issue No: 5508

Title: 69.8.2.8 The simple Element, page 69-538

Source:

Raytheon, Mr. Gerald Lee Bickle, Gerald_L_Bickle@raytheon.com

Summary:

2. 69.8.2.8 The simple Element, page 69-538

Simple Extensions and Changes

a) Enumerations - Add an optional enumerations element to the simple definition.

This allows a simple enumeration property to be defined. Each enumeration label has an associated value. The values are all of the same simple type.

Change simple element to:

```
<!ELEMENT simple
          ( description?
          , value
          , choices?
          , enumerations?
          , defaultvalue?
) >
```

Define new enumerations element

```
<!ELEMENT enumerations
  ( enumeration+
)>

<!ELEMENT enumeration EMPTY>

<!ATTLIST enumeration
     label     CDATA     #REQUIRED
     value     CDATA     #IMPLIED>
```

b) Short appears twice in the simple type attribute definition. Remove second occurrence.

```
<!ELEMENT simple
     ( description?
     , value
     , choices?
```

```
          , defaultvalue?
          ) >

     <!ATTLIST simple
          name CDATA #IMPLIED
          type ( Boolean
               | char
               | double
               | float
               | short        -- 1st occurrence
               | long
               | objref
               | octet
               | short       -- 2nd occurrence
               | string
               | ulong
               | ushort
               ) #REQUIRED >
```

c) Units - add an optional units element to indicate the units of measurement for a simple property.

```
     <!ELEMENT simple
          ( description?
          , value
          , choices?
          , units?
          , defaultvalue?
          ) >

     <!ELEMENT units (#PCDATA)>
```

d) Property Kind - The properties file for a component should not be restricted to only initial configuration properties. A component has many different types of properties and when defining a component one should be able to define these types of properties in a generic way. Add a kind element or attribute for a property definition. A component has readonly, writeonly, and readwrite properties. Simple properties can also be used for a component factory's create options parameter (home) or executable parameters/arguments. Only simple properties can be used for executable parameters.

New simplekind element

```
     <!ELEMENT simplekind EMPTY>

     <!ATTLIST simplekind
          kindtype  (configure_writeonly |
                     configure_readwrite | execparam |
                     query_readonly | factoryparam)
                    "configure_readwrite">
```

Change simple definition to:

```
<!ELEMENT simple
     ( description?
     , value
     , choices?
     , simplekind*
     , defaultvalue?
     ) >
```

The propertykind can be an optional element or attribute for the stuct and sequence elements.

New propertykind element

```
<!ELEMENT propertykind EMPTY>

<!ATTLIST propertykind
     kindtype  (configure_writeonly
               | configure_readwrite
               | query_readonly | factoryparam
               ) "configure">
```

Change Struct and Sequence to

```
<!ELEMENT struct
     ( description?
     , ( simple
       | sequence
       | struct
       | valuetype
       )*
     , propertykind?
     ) >

<!ATTLIST struct
     name CDATA #IMPLIED
     type CDATA #REQUIRED >

<!ELEMENT sequence
     ( description?
     , ( simple*
       | struct*
       | sequence*
       | valuetype*
       )
```

```
          , propertykind?
          ) >

     <!ATTLIST sequence
          name CDATA #IMPLIED
          type CDATA #REQUIRED >
```

Resolution:

None as this is deferred to the final report of the Components 1.1 RTF.

Revised Text:

None as this is deferred to the final report of the Components 1.1 RTF.

Disposition: **Unresolved**

Disposition Parameter: None

Disposition: Unresolved

OMG Issue No: 5509

Title: 69.8.2.7 The range Element, pages 69-537/538

Source:

Raytheon, Mr. Gerald Lee Bickle, Gerald_L_Bickle@raytheon.com

Summary:

Range Element - why is the min and max order not specified?

```
<!ELEMENT range (value, value) >
```

I understand the software logic can do a compare to determine the min and max values, but it seems the following definition would be easier to understand from human reader.

Change Range to

```
<!ELEMENT range EMPTY>

<!ATTLIST range
     min       CDATA     #REQUIRED
     max       CDATA     #REQUIRED>
```

Resolution:

None as this is deferred to the final report of the Components 1.1 RTF.

Revised Text:

None as this is deferred to the final report of the Components 1.1 RTF.

Disposition: **Unresolved**

Disposition Parameter: None

Disposition: Unresolved

OMG Issue No: 5510

Title: 69.8.2.3 The choices Element, page 69-537

Source:

Raytheon, Mr. Gerald Lee Bickle, Gerald_L_Bickle@raytheon.com

Summary:

Choices Element - It appears multiple ranges do not make sense for a simple definition. If so, then remove the range element from the choices element definition and make range an optional element for a simple.

Current definition

```
<!ELEMENT choices ( choice | range )+
```

Change to

```
<!ELEMENT choices ( choice )+

<!ELEMENT simple
     ( description?
     , value
     , choices?
     , range?
     , defaultvalue?
     ) >
```

Resolution:

None as this is deferred to the final report of the Components 1.1 RTF.

Revised Text:

None as this is deferred to the final report of the Components 1.1 RTF.

Disposition: **Unresolved**

Disposition Parameter: None

Disposition: Unresolved

OMG Issue No: 5511

Title: 69.8.2.9 The sequence Element

Source:

Raytheon, Mr. Gerald Lee Bickle, Gerald_L_Bickle@raytheon.com

Summary:

Simple Sequence -The current definitions for sequence allow invalid definitions to be built but be syntactically correct. A better definition for a simple sequence would be as follows:

Current sequence element

```
<!ELEMENT sequence
     ( description?
     , ( simple*
       | struct*
       | sequence*
       | valuetype*
       )
     ) >

<!ATTLIST sequence
     name CDATA #IMPLIED
     type CDATA #REQUIRED >
```

Change to

Add a new simplesequence element:

```
<!ELEMENT simplesequence
      ( description?,
      , values?
      , choices?
      , range?
      , enumerations?
      , units?
      ) >

<!ATTLIST simplesequence
     name CDATA     #IMPLIED
     type  ( boolean | char   | double | float | short
           | long | objref | octet | string | ulong
           | ushort )     #REQUIRED
```

```
<!ELEMENT values   ( value+   )>
```

Change sequence to:

```
<!ELEMENT sequence
     ( description?
     , ( simplesequence
       | struct*
       | sequence*
       | valuetype*
       )
     ) >

<!ATTLIST sequence
     name CDATA #IMPLIED
     type CDATA #REQUIRED >
```

One does not have to keep repeating the same simple definition. This definition then has the added advantage where simple name attribute can now be made mandatory.

```
<!ELEMENT simple
     ( description?
     , value
     , choices?
     , defaultvalue?
     ) >

 <!ATTLIST simple
     name CDATA #REQUIRED
     type ( Boolean
          | char
          | double
          | float
          | short
          | long
          | objref
          | octet
          | string
          | ulong
          | ushort
          ) #REQUIRED >
```

Resolution:

None as this is deferred to the final report of the Components 1.1 RTF.

Revised Text:

None as this is deferred to the final report of the Components 1.1 RTF.

Disposition: **Unresolved**

Disposition Parameter: **None**

Disposition: Unresolved

OMG Issue No: 5512

Title: Test Property - add a test property definition to the properties DTD

Source:

Raytheon, Mr. Gerald Lee Bickle, Gerald_L_Bickle@raytheon.com

Summary:

This allows one to define test properties for a component in a generic way. The Test property is based upon simple element.

Add new test element

```
<!ELEMENT test
     ( description
     , inputvalue?
     , resultvalue  )>

<!ATTLIST test
     name CDATA #REQUIRED>

<!ELEMENT inputvalue ( simple+ )>

<!ELEMENT resultvalue ( simple+ )>
```

Change properties element to:

```
<!ELEMENT properties
     ( description?
     , ( simple
       | sequence
       | struct
       | test
       | valuetype
       )+
     ) >
```

Resolution:

None as this is deferred to the final report of the Components 1.1 RTF.

Revised Text:

None as this is deferred to the final report of the Components 1.1 RTF.

Disposition: **Unresolved**

Disposition Parameter: None

Disposition: Unresolved

OMG Issue No: 5513

Title: Add the capability to define a component artifact property

Source:

Raytheon, Mr. Gerald Lee Bickle, Gerald_L_Bickle@raytheon.com

Summary:

7. Component Artifact - Add the capability to define a component artifact property. For example, a logical device component artifact could be used to specify the capacity for a device or the characteristic of a device. The artifacts for a component are referenced by a component's implementation dependency for using or deployed on relationships.

The component artifact property could be defined in two ways: 1) a new element called artifact or modifying the simple definition with optional new artifact element.

Artifacts are simple types. The action element defines the action that can be performed on an artifact. When a dependency references an artifact with a specified value this is the action that can be performed against a component's artifact. The ID is a DCE UUID to guarantee uniqueness of the artifact within the system; so 2 different artifacts, which are different, are not viewed to be the same artifact when they are not. The action denoted as external indicates the artifact is managed by the component. A referenced value for an artifact would have to be applied against the component.

Example 1 of New Artifact Element

```
<!ELEMENT artifact
     ( description?
     , simple
     , action
     )>

<!ATTLIST artifact
     id   ID #REQUIRED>

<!ELEMENT action EMPTY>

<!ATTLIST action
     type ( eq | ne | gt | lt | ge | le | external )
"external">
```

Change properties element to

```
<!ELEMENT properties
     ( description?
     , ( simple
       | sequence
       | struct
       | artifact
       | valuetype
       )+
     ) >
```

Example 2 of Modified Simple Element with new Artifact element

```
<!ELEMENT simplekind EMPTY>

<!ATTLIST simplekind
     kindtype  (configure_writeonly |
configure_readwrite | execparam |
     query_readonly | factoryparam | artifact)
"configure_readwrite">

<!ELEMENT simple
     ( description?
     , value
     , choices?
     , simplekind*
     , artifact?
     , defaultvalue?
     ) >

<!ELEMENT artifact ( action ) >

<!ATTLIST artifact
     id   ID #REQUIRED
     action ( eq | ne | gt | lt | ge | le | external )
"external">
```

Resolution:

None as this is deferred to the final report of the Components 1.1 RTF.

Revised Text:

None as this is deferred to the final report of the Components 1.1 RTF.

Disposition: **Unresolved**

Disposition Parameter: None

Disposition: Unresolved

OMG Issue No: 5514

Title: 69.3 Software Package Descriptor

Source:

Raytheon, Mr. Gerald Lee Bickle, Gerald_L_Bickle@raytheon.com

Summary:

69.3.2.1 The softpkg Root Element, page 69-473 Softpkg Element Cardinality. Why does it make sense to have an empty softpkg file or to allow multiple elements such as title, pkgtype, description, license, or to have no implementations? Suggested changes are to place the correct cardinality on the softpkg elements to be similar to the CORBA Component and Component Assembly Descriptor definitions. The suggested elements cardinality changes will make parsing the XML much simpler and reduce code footprint, and the XML more understandable.

Current Format

```
<!ELEMENT softpkg
     ( title
     | pkgtype
     | author
     | description
     | license
     | idl
     | propertyfile
     | dependency
     | descriptor
     | implementation
     | extension
     )* >

<!ATTLIST softpkg
     name ID #REQUIRED
     version CDATA #OPTIONAL >
```

Suggested New Format for Softpkg

- Zero or one title ? for example, a book only has one title, not multiple titles.

- One or more authors ? for example, a book can have multiple authors, but at least one.

- Zero or one description, the CAD and CORBA Components DTDs define the description this way.
- Zero or one package type.
- Zero or one property file reference. Why are multiple property files needed?
- One or more implementations, since the softpkg is an implementation for a component definition then it must have at least one implementation.
- Zero or more dependencies for all implementations.
- Zero or more descriptors for all implementations. An implementation may contain multiple different components.
- Zero or more IDL files for all implementations.
- Zero or more extensions.

```
<!ELEMENT softpkg
     (  title?
     , author+
     , description?
     , propertyfile?
     , license?
     , pkgtype?
     , implementation+
     , ( idl
       | dependency
       | descriptor
       | extension
       )*
     ) >

<!ATTLIST softpkg
     name      ID   #REQUIRED
     version        CDATA     #IMPLIED >
```

Resolution:

None as this is deferred to the final report of the Components 1.1 RTF.

Revised Text:

None as this is deferred to the final report of the Components 1.1 RTF.

Disposition: **Unresolved**

Disposition Parameter: None

Disposition: Unresolved

OMG Issue No: 5515

Title: 69.3.2.15 The implementation Element, pages 69-478/479

Source:

Raytheon, Mr. Gerald Lee Bickle, Gerald_L_Bickle@raytheon.com

Summary:

Implementation Element Cardinality. Why does it make sense to have an empty implementation or to allow multiple elements such as code, description, humanlanguage, or to have no code? Suggested changes are to place the correct cardinality on the implementation elements. The suggested elements cardinality changes will make parsing the XML much simpler and reduce code footprint, and the XML more understandable.

Current Format

```
<!ELEMENT implementation
     ( description
     | code
     | compiler
     | dependency
     | descriptor
     | extension
     | programminglanguage
     | humanlanguage
     | os
     | propertyfile
     | processor
     | runtime
     )* >

<!ATTLIST implementation
     id ID #IMPLIED
     variation CDATA #IMPLIED >
```

Suggested New Format for implementation

- Zero or one description, the CAD and CORBA Components DTDs define the description this way.

- Zero or one property file reference. Why are multiple property files needed?

- One code element is required for a given implementation

- Zero or one complier element. A given source code element is compiled by a specific compiler. Specifying the compiler is optional.

- Zero or one programminglanguage element. Specifying the programminglanguage is optional.

- Zero or one humanlanguage element. Specifying the humanlanguage is optional.

- Zero or more dependencies for a specific implementation.

- Zero or more descriptors for a specific implementation. An implementation may contain multiple different components.

- Zero or more runtimes. A given code element may be compatible with multiple runtime environments.

- Zero or more os. A given code element may be compatible with multiple operating systems.

- Zero or more processors. A given code element may be compatible with multiple processors.

- Zero or more extensions.

```
<!ELEMENT implementation
     ( description?
     , code
     , compiler?
     , humanlanguage?
     , programminglanguage?
     , propertyfile?
     , (  dependency
       | descriptor
       | extension
       | os
       | processor
       | runtime
       | usescomponent
       )*
     )>

<!ATTLIST implementation
     id ID #IMPLIED
     variation CDATA #IMPLIED >
```

Resolution:

None as this is deferred to the final report of the Components 1.1 RTF.

Revised Text:

None as this is deferred to the final report of the Components 1.1 RTF.

Disposition: Unresolved

Disposition Parameter: None

Disposition: Unresolved

OMG Issue No: 5516

Title: 69.8.2.7 The code Element, pages 69-474

Source:

Raytheon, Mr. Gerald Lee Bickle, Gerald_L_Bickle@raytheon.com

Summary:

a) Add the optional stacksize and priority elements to code element definition. The stack size and priority are options parameters for an operating system's execute() operation. The stack size is the memory space needed by the executable and the OS priority for the executable. Data types for the values of these options are unsigned long. Priority number should be based upon industry standard such as POSIX.

Current Format

```
<!ELEMENT code
     ( ( codebase
       | fileinarchive
       | link
       )
     , entrypoint?
     , usage?
) >

<!ATTLIST code
     type CDATA #IMPLIED >
```

New Format for code

```
<!ELEMENT code
     ( ( codebase
       | fileinarchive
       | link
       )
     , entrypoint?
     , stacksize?
     , priority?
     , usage?
     ) >

<!ATTLIST code
     type CDATA #IMPLIED >
```

```
<!ELEMENT stacksize (#PCDATA)>

<!ELEMENT priority (#PCDATA)>
```

b) Other valid values for type attribute

Additional valid values for the type attribute are: "KernelModule", "SharedLibrary", and "Driver".

Resolution:

None as this is deferred to the final report of the Components 1.1 RTF.

Revised Text:

None as this is deferred to the final report of the Components 1.1 RTF.

Disposition: **Unresolved**

Disposition Parameter: None

Disposition: Unresolved

OMG Issue No: 5517

Title: 69.3.2.15 The implementation Element, pages 69-478/479

Source:

Raytheon, Mr. Gerald Lee Bickle, Gerald_L_Bickle@raytheon.com

Summary:

Add license element to implementation element. Why is the license element restricted to the softpkg level only? This is saying all implementations have the same license. Cannot each implementation have its own different license? Suggest adding license element to implementation element.

Current Format

```
<!ELEMENT implementation
     ( description
     | code
     | compiler
     | dependency
     | descriptor
     | extension
     | programminglanguage
     | humanlanguage
     | os
     | propertyfile
     | processor
     | runtime
     )* >

<!ATTLIST implementation
     id ID #IMPLIED
     variation CDATA #IMPLIED >
```

New Format

```
<!ELEMENT implementation
     ( description
     | code
     | compiler
     | dependency
     | descriptor
     | extension
     | license
```

```
          | programminglanguage
          | humanlanguage
          | os
          | propertyfile
          | processor
          | runtime
          )* >

     <!ATTLIST implementation
          id ID #IMPLIED
          variation CDATA #IMPLIED >
```

Resolution:

None as this is deferred to the final report of the Components 1.1 RTF.

Revised Text:

None as this is deferred to the final report of the Components 1.1 RTF.

Disposition: **Unresolved**

Disposition Parameter: None

Disposition: Unresolved

OMG Issue No: 5518

Title: 69.3.2.25 The propertyfile Element, page 69-482

Source:

Raytheon, Mr. Gerald Lee Bickle, Gerald_L_Bickle@raytheon.com

Summary:

Propertyfile clarification is needed for consistent behavior. The only statement made about propertyfile is that the implementation's propertyfile has precedence over the same propertyfile types at the softpkg level. Why are multiple property files needed at the softpkg and implementation levels? If more than one propertfile exist at any level, which property file has precedence in the list? If multiple property files exists are they merged together? Are the softpkg's descriptor element property files merge with the softpkg property files and which one has precedence? Are the implementation's descriptor property files merge with the implementation property files, and which one has precedence? Are implementation property files merge with the softpkg property files?

Resolution:

None as this is deferred to the final report of the Components 1.1 RTF.

Revised Text:

None as this is deferred to the final report of the Components 1.1 RTF.

Disposition: **Unresolved**

Disposition Parameter: None

Disposition: Unresolved

OMG Issue No: 5519

Title: 69.3.2.14 The idl Element, page 69-478

Source:

Raytheon, Mr. Gerald Lee Bickle, Gerald_L_Bickle@raytheon.com

Summary:

a) Add idl element to implementation element. Why is idl only at the softpkg level? This is saying that all implementations use the same IDL. This is inconsistent with descriptor element. An implementation can specify a descriptor, why not idl? Cannot an implementation use a specific IDL for its implementation?

b) Why is IDL defined in the software package descriptor instead of the CORBA Component descriptor?

Resolution:

None as this is deferred to the final report of the Components 1.1 RTF.

Revised Text:

None as this is deferred to the final report of the Components 1.1 RTF.

Disposition: **Unresolved**

Disposition Parameter: None

Disposition: Unresolved

OMG Issue No: 5520

Title: Descriptor

Source:

Raytheon, Mr. Gerald Lee Bickle, Gerald_L_Bickle@raytheon.com

Summary:

How does the Assembly Descriptor support multiple components within an implementation?

Resolution:

None as this is deferred to the final report of the Components 1.1 RTF.

Revised Text:

None as this is deferred to the final report of the Components 1.1 RTF.

Disposition: **Unresolved**

Disposition Parameter: None

Disposition: Unresolved

OMG Issue No: 5521

Title: 69.3.2.2 The author Element, page 69-474

Source:

Raytheon, Mr. Gerald Lee Bickle, Gerald_L_Bickle@raytheon.com

Summary:

Change the format of author to group related authors together from a company. One does not need the capability to list multiple companies and web sites together in the author element, since there can be many authors listed in the softpkg element. The suggested elements cardinality changes will make parsing the XML much simpler and reduce code footprint, and the XML more understandable.

Current format

```
<!ELEMENT author
     ( name
     | company
     | webpage
     )* >
```

New format

```
<!ELEMENT author
     ( name*
     , company?
     , webpage?
     )>
```

Resolution:

None as this is deferred to the final report of the Components 1.1 RTF.

Revised Text:

None as this is deferred to the final report of the Components 1.1 RTF.

Disposition: **Unresolved**

Disposition Parameter: None

Disposition: Unresolved

OMG Issue No: 5522

Title: Component Artifact Dependency

Source:

Raytheon, Mr. Gerald Lee Bickle, Gerald_L_Bickle@raytheon.com

Summary:

An implementation may request a dependency on a specific component based upon its artifacts. Add artifactdependency to the dependency element.

Current Format

```
<!ELEMENT dependency
     ( softpkgref
     | codebase
     | fileinarchive
     | localfile
     | name
     | valuetypefactory
     ) >

<!ATTLIST dependency
     type CDATA #IMPLIED
     action (assert | install)"assert">
```

New Format

```
<!ELEMENT dependency
     ( softpkgref
     | codebase
     | fileinarchive
     | localfile
     | name
     | valuetypefactory
     )
     , artifactdependency*
>

<!ATTLIST dependency
     type CDATA #IMPLIED
     action (assert | install)"assert">

<!ELEMENT artifactdependency EMPTY>
```

```
<!ATTLIST artifactdependency
    artifactrefid       CDATA     #REQUIRED
    artifactvalue       CDATA     #REQUIRED>
```

Note: This concept is tied to the concept of component artifact property issue. The artifactrefid is a reference to a component's artifact property defined in a component's property file. The artifactvalue is the dependency value being requested or needed.

Resolution:

None as this is deferred to the final report of the Components 1.1 RTF.

Revised Text:

None as this is deferred to the final report of the Components 1.1 RTF.

Disposition: Unresolved

Disposition Parameter: None

Disposition: Unresolved

OMG Issue No: 5523

Title: Device Artifact Dependency

Source:

Raytheon, Mr. Gerald Lee Bickle, Gerald_L_Bickle@raytheon.com

Summary:

A component's implementation may have additional dependencies on a device's artifacts (e.g., capacity and/or characteristics) to ensure the right type of device is chosen for deployment and/or the device has sufficient capacities for deployment. To allow for this capability add a deviceartifactdependency element to the implementation element.

Current Format

```
<!ELEMENT implementation
     ( description
     | code
     | compiler
     | dependency
     | descriptor
     | extension
     | programminglanguage
     | humanlanguage
     | os
     | propertyfile
     | processor
     | runtime
     )* >

<!ATTLIST implementation
     id ID #IMPLIED
     variation CDATA #IMPLIED >
```

New Format

```
<!ELEMENT implementation
     ( description
     | code
     | compiler
     | dependency
     | descriptor
     | extension
```

```
          | programminglanguage
          | humanlanguage
          | os
          | propertyfile
          | processor
          | runtime
          | deviceartifactdependency
          )* >

     <!ATTLIST implementation
          id ID #IMPLIED
          variation CDATA #IMPLIED >

     <!ELEMENT deviceartifactdependency EMPTY>

     <!ATTLIST deviceartifactdependency
          artifactrefid       CDATA     #REQUIRED
          artifactvalue       CDATA     #REQUIRED>
```

Note: This concept is tied to the concept of component artifact property issue. The artifactrefid is a reference to a component's artifact property defined in a component's property file. The artifactvalue is the dependency value being requested or needed.

Resolution:

None as this is deferred to the final report of the Components 1.1 RTF.

Revised Text:

None as this is deferred to the final report of the Components 1.1 RTF.

Disposition: **Unresolved**

Disposition Parameter: None

Disposition: Unresolved

OMG Issue No: 5524

Title: Uses Relationships

Source:

Raytheon, Mr. Gerald Lee Bickle, Gerald_L_Bickle@raytheon.com

Summary:

The softpkg element only deals with deployed on and library load dependency relationships for implementations. Component implementations may also have specific using relationships with another component, such as a device within the system. This relationship can be stated at the softpkg or implementation level.

Current Format

```
<!ELEMENT softpkg
     ( title
     | pkgtype
     | author
     | description
     | license
     | idl
     | propertyfile
     | dependency
     | descriptor
     | implementation
     | extension
     )* >

<!ATTLIST softpkg
     name ID #REQUIRED
     version CDATA #IMPLIED >

<!ELEMENT implementation
     ( description
     | code
     | compiler
     | dependency
     | descriptor
     | extension
     | programminglanguage
     | humanlanguage
     | os
     | propertyfile
```

```
          | processor
          | runtime
          )* >

     <!ATTLIST implementation
          id ID #IMPLIED
          variation CDATA #IMPLIED >
```

New Format

```
     <!ELEMENT softpkg
          ( title
          | pkgtype
          | author
          | description
          | license
          | idl
          | propertyfile
          | dependency
          | descriptor
          | implementation
          | extension
          | usescomponent
          )* >

     <!ATTLIST softpkg
          name ID #REQUIRED
          version CDATA #IMPLIED >

     <!ELEMENT implementation
          ( description
          | code
          | compiler
          | dependency
          | descriptor
          | extension
          | programminglanguage
          | humanlanguage
          | os
          | propertyfile
          | processor
          | runtime
          | usescomponent
          )* >

     <!ATTLIST implementation
          id ID #IMPLIED
          variation CDATA #IMPLIED >
```

```
<!ELEMENT  usescomponent
     ( artifactdependency+ )>

<!ATTLIST usescomponent
     id        ID              #REQUIRED
     type      CDATA     #REQUIRED>

<!ELEMENT artifactdependency EMPTY>

<!ATTLIST artifactdependency
     artifactrefid       CDATA     #REQUIRED
     artifactvalue       CDATA     #REQUIRED>
```

Note: This concept is tied to the concept of component artifact property issue. The artifactrefid is a reference to a component's artifact property defined in a component's property file. The artifactvalue is the dependency value being requested or needed.

Resolution:

None as this is deferred to the final report of the Components 1.1 RTF.

Revised Text:

None as this is deferred to the final report of the Components 1.1 RTF.

Disposition: **Unresolved**

Disposition Parameter: None

Disposition: Unresolved

OMG Issue No: 5576

Title: 69.3 AssemblyFactory Interface

Source:

Raytheon, Mr. Gerald Lee Bickle, Gerald_L_Bickle@raytheon.com

Summary:

Suggested changes to the AssemblyFactory interface.

AssemblyFactory Issues.

1. Ease of use Issue. After the create operation is performed, one is force to call lookup to get the Assembly that just got just created. Why is a cookie returned by the create operation instead of an Assembly? Is the reason because of security? If the interface were more open this would still allow a secure implementation to be implemented.

Suggested change is to return an Assembly instead of a Cookie. Change destroy operation to take in an Assembly parameter instead of Cookie.

Change lookup operation to take in a name parameter. These changes make it consistent with the other CCM interfaces, such as Container, KeyLessCCMHome, ComponentServer, and ServerActivator.

2. Multiple users Issue. For multiple users, how does a client know what assemblies are created. Add a get_assemblies operation that returns a list of assemblies. These changes make it consistent with other CCM interfaces, such as Container, ComponentServer, and ServerActivator.

3. User-friendly identifier for Assembly Instance issue. Add an input name parameter that can be assigned to the Assembly instance that gets created. Add a read only name or label attribute to the Assembly interface.

Resolution:

None as this is deferred to the final report of the Components 1.1 RTF.

Revised Text:

None as this is deferred to the final report of the Components 1.1 RTF.

Disposition: **Unresolved**

Disposition Parameter: None

Disposition: Accepted

OMG Issue No: 5577

Title: create operation of AssemblyFactory interface

Source:

Fraunhofer FOKUS, Mr. Tom Ritter, ritter@fokus.gmd.de

Summary:

The name of the create operation in the AssemblyFactory Interface shall be renamed to a more specific identifier. Create is often used in other interfaces and this may lead to problems e.g. in inheritance relationships.

suggested change:

create -> create_assembly

Resolution:

The name of the **`create`** operation in the **`AssemblyFactory`** interface must be renamed to a more specific identifier, i.e. **`create_assembly`**, as the **`create`** identifier is often used in other interfaces and this may lead to problems e.g. in inheritance relationships.

Revised Text:

In formal/02-06-65, page 6-72, section 6.9.3, replace

```
     Cookie create( . . . );
```

by

```
     Cookie create_assembly( . . . );
```

In formal/02-06-65, page 6-73, replace **`create`** by **`create_assembly`** two times.

Disposition: **Accepted**

Disposition Parameter: None

Disposition: Accepted

OMG Issue No: 5583

Title: Remove section 4.4.1.4 in formal/02-06-65

Source:

Laboratoire d'Informatique Fondamentale de Lille, Dr. Philippe Merle, Philippe.Merle@lifl.fr

Summary:

Proposed resolution:

The Adopted CORBA Components Specification (formal/02-06-65) always contains the section 4.4.1.4 at page 4-36.

However, this section was removed in the ptc/01-11-03 document by the resolution of the issue 3937 of the Final Report of Components December 2000 FTF (ptc/01-11-02).

Proposed revised text:

In formal/02-06-65 page 4-36, remove the section 4.4.1.4.

Resolution:

The Adopted CORBA Components Specification (formal/02-06-65) always contains the section 4.4.1.4 at page 4-36.

However, this section was removed in the ptc/01-11-03 document by the resolution of the issue 3937 of the Final Report of Components December 2000 FTF (ptc/01-11-02).

Moreover, the **`CCM2Context`** interface in section 4.4.1.1 page 4-33 always contains the **`get_event`** operation which was removed in document ptc/01-11-03 page 62-358.

Revised Text:

In formal/02-06-65 page 4-33, remove the 3rd item named " **`Event`** ".

In formal/02-06-65 page 4-33, remove the **`get_event`** operation in the **`CCM2Context`** interface.

In formal/02-06-65 page 4-36, remove the section 4.4.1.4.

Disposition: **Accepted**

Disposition Parameter: None

Disposition: Accepted

OMG Issue No: 5584

Title: Corrections in XML DTDs for packaging

Source:

Laboratoire d'Informatique Fondamentale de Lille, Dr. Philippe Merle, Philippe.Merle@lifl.fr

Summary:

Proposed resolution:

In the Adopted CORBA Components Specification (formal/02-06-65), page 6-4, section 6.3.2.1, the version attribute of the softpkg element is tagged as #OPTIONAL and is tagged as #IMPLIED at page 7-5.

As #OPTIONAL is not valid in XML, then replace it by #IMPLIED.

In chapter 7, the softpkg XML DTD does not contain the ins and objref elements which they are used by the repository element described at page 7-4. Add them.

Proposed revised text:

In formal/02-06-65:

At page 6-4, section 6.3.2.1, replace #OPTIONAL by #IMPLIED for the version attribute of the softpkg element.

At page 7-3, add before the humanlanguage element

```
<!ELEMENT ins EMPTY >
<!ATTLIST ins
   name CDATA #REQUIRED >
```

At page 7-4, add before the os element

```
<!ELEMENT objref EMPTY >
<!ATTLIST objref
   string CDATA #REQUIRED >
```

Resolution:

In the Adopted CORBA Components Specification (formal/02-06-65), page 6-4, section 6.3.2.1, the **`version`** attribute of the **`softpkg`** element is tagged as **`#OPTIONAL`** and is tagged as **`#IMPLIED`** at page 7-5.

As **`#OPTIONAL`** is not valid in XML, then replace it by **`#IMPLIED`**.

In chapter 7, the **`softpkg`** XML DTD does not contain the **`ins`** and **`objref`** elements which they are used by the **`repository`** element described at page 7-4. Add them.

Revised Text:

In formal/02-06-65:

At page 6-4, section 6.3.2.1, replace **`#OPTIONAL`** by **`#IMPLIED`** for the **`version`** attribute of the **`softpkg`** element.

At page 7-3, add before the **`humanlanguage`** element

```
<!ELEMENT ins EMPTY >
<!ATTLIST ins
    name CDATA #REQUIRED >
```

At page 7-4, add before the **`os`** element

```
<!ELEMENT objref EMPTY >
<!ATTLIST objref
    string CDATA #REQUIRED >
```

Disposition: Accepted

Disposition Parameter: None

Disposition: Accepted

OMG Issue No: 5585

Title: Editorial issues in formal/02-06-65

Source:

Laboratoire d'Informatique Fondamentale de Lille, Dr. Philippe Merle, Philippe.Merle@lifl.fr

Summary:

Proposed resolution:

The Adopted CORBA Components Specification (formal/02-06-65) always contains some texts removed by the Components December 2000 RTF (ptc/01-11-02 and ptc/01-11-03). See at

formal/02-06-65	ptc/01-11-03
page 1-24 and 25	page 61-241
page 1-26	page 61-243
page 1-28	page 61-245
page 4-37	page 62-363
page 6-67	page 69-542
page 6-72	page 69-548
page 8-23	page 70-601

Remove these old texts once again.

Proposed revised text:

Following must be applied on formal/02-06-65.

At pages 1-24 and 1-25, remove

```
module <module_name> {
  module <component_name>EventConsumers {
    interface <event_type>Consumer;
  };
  interface <component_name> : Components::CCMObject {
    Components::Cookie subscribe_ <source_name> (
      in <component_name>EventConsumers::
              <event_type>Consumer  consumer)
      raises (Components::ExceededConnectionLimit);
    <component_name>EventConsumers::<event_type>
```

```
     Consumer unsubscribe_<source_name> (in
     Components::Cookie ck)
           raises (Components::InvalidConnection);
       };
       module <component_name>EventConsumers {
         interface <event_type>Consumer :
     Components::EventConsumerBase {
           void push (in <event_type> evt);
         };
       };
     };
```

At page 1-26, remove

```
    module <module_name> {
      module <component_name>EventConsumers {
        interface <event_type>Consumer;
      };
      interface <component_name> : Components::CCMObject {
        void connect_ <source_name> ( in <component_name>
        EventConsumers:: <event_type>Consumer consumer )
        raises (Components::AlreadyConnected);
        <component_name>EventConsumers::<event_type>
        Consumer disconnect_<source_name>()
        raises (Components::NoConnection);
      };
      module <component_name>EventConsumers {
        interface <event_type> Consumer : Components::
EventConsumerBase {
          void push (in <event_type> evt);
        };
      };
    };
```

At page 1-28, remove

```
    module <module_name> {
      module <component_name>EventConsumers {
        interface <event_type>Consumer;
      };
      interface <component_name> : Components::CCMObject {
        <component_name>EventConsumers::<event_type>
        Consumer get_consumer _<sink_name>();
      };
      module <component_name>EventConsumers {
        interface <event_type>Consumer :
Components::EventConsumerBase {
```

```
          void push (in <event_type> evt);
        };
      };
    };
```

At page 4-37, remove in the **`Session2Context`** interface the two occurrences of **`PortableServer::ObjectId`**.

At page 6-67, remove

> Note Of the interfaces described below, only ComponentInstallation, AssemblyFactory, and Assembly are required by this specification; the other interfaces are included for illustrative purposes and to support an end-to-end scenario.

At page 6-72, remove

```
exception InvalidLocation { };
```

At page 8-23, remove Figure 8-20.

Resolution:

The Adopted CORBA Components Specification (formal/02-06-65) always contains some texts removed by the Components December 2000 RTF (ptc/01-11-02 and ptc/01-11-03). See at

formal/02-06-65	ptc/01-11-03
page 1-24 and 25	page 61-241
page 1-26	page 61-243
page 1-28	page 61-245
page 4-37	page 62-363
page 6-67	page 69-542
page 6-72	page 69-548
page 8-23	page 70-601

Remove these old texts once again.

Revised Text:

Apply the proposed removals listed in the summary body.

Disposition: Accepted

Disposition Parameter: None

Disposition: Unresolved

OMG Issue No: 5588

Title: Update Table 5-13 in the EJB Chapter of formal/02-06-65

Source:

Laboratoire d'Informatique Fondamentale de Lille, Dr. Philippe Merle, Philippe.Merle@lifl.fr

Summary:

In ptc/01-11-03, page 64-410, there is the following note

> Issue This table will be completed after the Interface Repository chapter is ready.

Then Table 5-13 in formal/02-06-65 would be completed.

Resolution:

None as this is deferred to the final report of the Components 1.1 RTF.

Revised Text:

None as this is deferred to the final report of the Components 1.1 RTF.

Disposition: **Unresolved**

Disposition Parameter: None

Disposition: Unresolved

OMG Issue No: 5589

Title: Description for the impltype Element?

Source:

Laboratoire d'Informatique Fondamentale de Lille, Dr. Philippe Merle, Philippe.Merle@lifl.fr

Summary:

In formal/02-06-65, page 6-54, there is the following text

Placeholder for future version.

The section 6.7.2.33 would be written.

Resolution:

None as this is deferred to the final report of the Components 1.1 RTF.

Revised Text:

None as this is deferred to the final report of the Components 1.1 RTF.

Disposition: **Unresolved**

Disposition Parameter: None

Disposition: Unresolved

OMG Issue No: 5590

Title: Checking XML DTD elements related to the trader service

Source:

Laboratoire d'Informatique Fondamentale de Lille, Dr. Philippe Merle, Philippe.Merle@lifl.fr

Summary:

In ptc/01-11-03, page 69-533, there is the following note

> Issue The trader elements have to be reviewed to make sure that they serve the purpose intended. Also, consider using a property file.

XML DTD elements related to the trader service would be checked.

Resolution:

None as this is deferred to the final report of the Components 1.1 RTF.

Revised Text:

None as this is deferred to the final report of the Components 1.1 RTF.

Disposition: **Unresolved**

Disposition Parameter: None

Disposition: Unresolved

OMG Issue No: 5591

Title: Using Configurator on CCMHome or any CORBA objects?

Source:

Laboratoire d'Informatique Fondamentale de Lille, Dr. Philippe Merle, Philippe.Merle@lifl.fr

Summary:

In ptc/01-11-03, page 69-545, there is the following note

> Issue The Configurator interface takes an argument of type CCMObject and therefore cannot be used to configure component homes. I see no reason not to widen the type to CORBA::Object so that the Configurator can be used for objects other than CCMObjects. The StandardConfigurator is after all a basic name value pair configurator that simply executes calls on mutator operations resulting from attribute declarations. J.S.E.

The Configurator interface could be updated to allow configuration of any CORBA objects.

Resolution:

None as this is deferred to the final report of the Components 1.1 RTF.

Revised Text:

None as this is deferred to the final report of the Components 1.1 RTF.

Disposition: Unresolved

Disposition Parameter: None

Disposition: Unresolved

OMG Issue No: 5594

Title: [CCM] Interface Repository Metamodel

Source:

Teiniker Egon, egon.teiniker@tugraz.at

Summary:

in the BaseIDL there is a class StructDef which has the Attribute members of Type Field. How can I model a IDL struct with more than one entry? I think there should be a aggregation from StructDef (1<>----->*) to the Field class (Page 8-10 of the CCM Spec).

) With EnumDef there is the same problem, I guess a assotiation from EnumDef to string (1<>----->) would solve it (Page 8-10 of the CCM Spec).

*) Also with ExceptionDif and its attribute members (Page 8-11 of the CCM Spec).

Resolution:

None as this is deferred to the final report of the Components 1.1 RTF.

Revised Text:

None as this is deferred to the final report of the Components 1.1 RTF.

Disposition: **Unresolved**

Disposition Parameter: None