Container Catalog

MSR MEDOC, Roman Reimer, XI-Works

26/08/2002 09:42:39 msrcc-eadoc.xml

Alle Rechte bei MSR MEDOC. Jede Verfügungserlaubnis, wie Kopier- und Weitergaberecht liegt bei uns.

Zusammenfassung

In case of joint engineering efforts it is necessary to synchronise the engineering data bases. This document specifies a container and in particular the metadata used for such an exchange.

Note that this document is not perfect[1]. It is the result of a meeting where the participiants tried to harmonize various catalog approaches.

This document must be accompanied by use case specific documentation.

Especially the following information is still missing:

- Examples
- something more which is left as an execise to the reader.

1

26/08/2002 09:42:39 msrcc-eadoc.xml

Alle Rechte bei MSR MEDOC. Jede Verfügungserlaubnis, wie Kopier- und Weitergaberecht liegt bei uns.

Inhaltsverzeichnis

26/08/2002 09:42:39 msrcc-eadoc.xml

Alle Rechte bei MSR MEDOC. Jede Verfügungserlaubnis, wie Kopier- und Weitergaberecht liegt bei uns.

26/08/2002 09:42:39 msrcc-eadoc.xml

Alle Rechte bei MSR MEDOC. Jede Verfügungserlaubnis, wie Kopier- und Weitergaberecht liegt bei uns.

Präliminarien

Projektbeteiligte Firmen

MSR MEDOC [MEDOC]

Name Rollen	Abteilung	Adresse	Kontakt
Bernhard Weichel Bosch			
Oliver Rumpf BMW			
Roman Reimer, XI-Works			

Versionsinformation

Dokumentteil	Herausgeber			
	Firma	Version	Status	Anmerkungen
2002-02-07 Details siehe Nr. 1, Seite 26	Roman Reimer, XI-Works			
	MEDOC	2	RD	

26/08/2002 09:42:39 msrcc-eadoc.xml

Alle Rechte bei MSR MEDOC. Jede Verfügungserlaubnis, wie Kopier- und Weitergaberecht liegt bei uns.

1 Introduction

This document describes the container catalog which is part of a generic approach for exchange of engineering data and its configuration. The container comprises of

- Engineering objects such as source codes, documentation, compiled objects etc.
- Metainformation about the exchanged object such as creator, name, version info and configuration.

The catalog as described in the following sections is inspired by the *RDF* (Ressource Data Framework) as published by the World Wide Web Consortioum. RDF describes thing by making statements about it. These statements, called ”assertions” are the basic concepts of the catalog.

The statements are given in **<ablock>** which itself comprises of metadata and the ”things” istself resp. a pointer to a file containing the ”things”.

Like RDF, the catalog is an application of *XML* as described by the *world wide web consortium* (*www.w3c.org*).

This document describes in general the container *catalog dtd*. The dtd provides a metadata scheme. It is expected, that the reader is informed about XML.

The particulare application of this scheme must be documented separately.

1.1 General approach to the Catalog

In general, the catalog is setup by the following items:

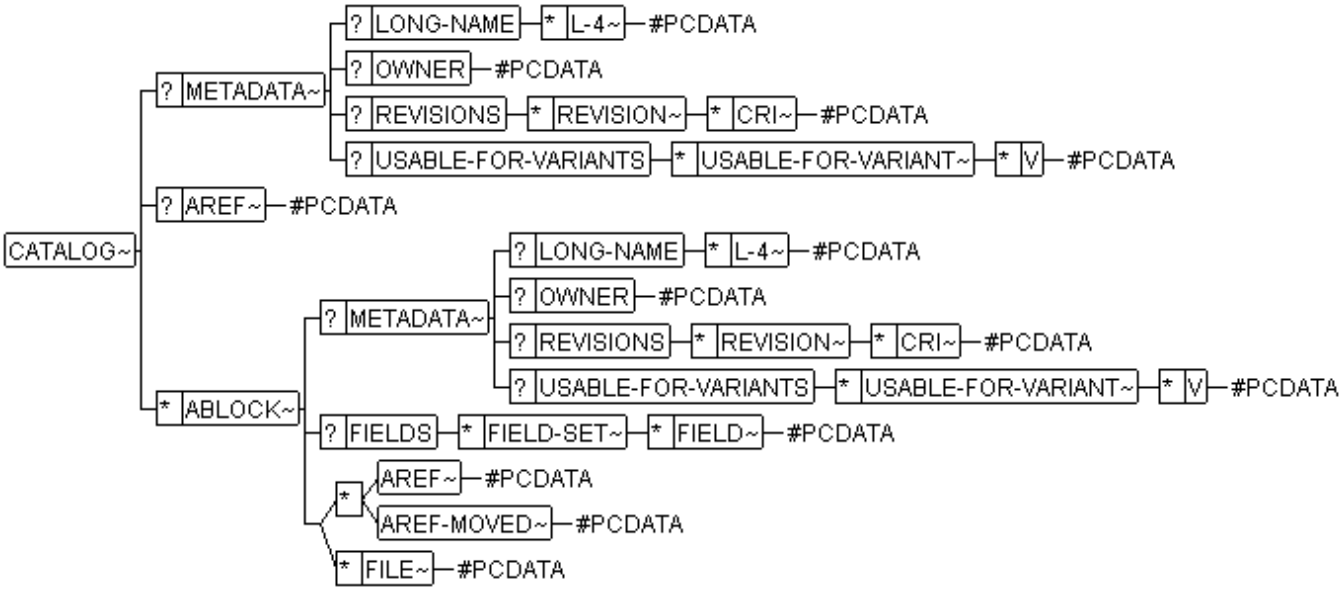

Abbildung 1: Overall structure of the catalog

<catalog> This is the root of the catalog. As of SGML/XML a valid resp. a well formed document must have one and only one root element.

<ablock> This is the assertion about one particular ”work item” or exchange item. This item can either be a file, a set of fields, or a specific configuration which is comprised by **<aref>**.

Alle Rechte bei MSR MEDOC. Jede Verfügungserlaubnis, wie Kopier- und Weitergaberecht liegt bei uns.

The primary identification of an **<ablock>** is the combination of **[class]** and **[name]** . This reflects to the fact that a work item in an engineering process may have a particular role in the process (denoted by **[class]**) and a name.

For ease of processing in SGML/XML-tools catalog also supports the identification by **[id]**.

<aref> This is used to establish a relationship between configuration items and particular exchange items. As an example, a program kit is setup as a set of particular files. This fact is expressed within the **<ablock>** describing the program by referencing (via **<aref>**) the **<ablock>** describing the particular files.

Note that this reference can be expressed using *SGML*'s *ID/IDREF* or by using **[class]** and **[name]** or both. If both mechanisms are use, they must be in sync.

<metadata> This structure receives all metadata formally specified in the catalog. This is mainly a descriptive name delivered in multiple languages and revision information.

<fields> This is a generic structure allowing to transmit any kind of fields. The application domain of a field set can be denoted in **[class]** within **<field-set>**. For an example see Topic 2 Using the catalog for transmitting a revision history S. 9.

For further details refer to the element and attributes documentation in Topic 2.2 Elements and attributes S. 11.

1.2 Incremental data exchange

The catalog can be used for full fledged as well as for incremental data exchange. In both cases, all metadata have to be transmitted. In case of incremental exchange the unchanged physical files can be omitted.

In order to keep all references valid within the catalog the appropriate **<ablock>**s must be provided. Therefore, the **[upd]** attribute allows to give information about the contents:

NEW The object is newly introduced in the system. Therefore the contents file must be in the catalog (in particular, an **<file>** element must occur within **<ablock>**)[2].

REUSED The object is introduced in the system at former time. Like NEW it is not in the reference catalog. Therefore the contents file must be in the catalog (in particular, an **<file>** element must occur within **<ablock>**).

UNUSED The object is no longer used. The **<ablock>** is there in order to indicate, that the object can be removed in the configuration on the receiver's site. The **<ablock>** does not reference anything within the actual catalog[3].

This is used to support the update procedure on the receiver's site[4].

UNCHANGED The object is unchanged. Therefore it must already be available on the receiver's site. Therefore it is possible to omit the file itself.

CHANGED The object is changed. Therefore the contents file must be in the catalog (in particular, an **<file>** element must occur within **<ablock>**).

MOVED The object is **unchanged** but in different Groups. Therefore it must already be available on the receiver's site. So it is possible to omit the file itself.

2

3

4

26/08/2002 09:42:39 msrcc-eadoc.xml

Alle Rechte bei MSR MEDOC. Jede Verfügungserlaubnis, wie Kopier- und Weitergaberecht liegt bei uns.

The object is **changed** and in different Groups. In this case it will be changed.

1.3 Syntactical hints

The general format of the container catalog is SGML. The structure of the container catalog is therefore defined as an *SGML DTD*. The following specifics apply as they are adjusted in the SGML declaration:

- the concrete reference syntax is used (refer to the widely available SGML material (e.g http://www.oasis-open.org).
- It is possible to parse a catalog as well formed xml which is without using a DTD. In this case it is recommended to have the following xml declaration in the first line of the container:

  Note that well formed parsing does not check names, ids, reserved attribute values.
- Names resp. nmtoken (as shown in the attribute tables below, eg. Tabelle 4 Attributes for CATALOG S. 17) are limited to 256 characters, must start with a letter, and may consist of letters, digits, and the characters ”-”, ”_”, ”.”.
- As of March 2000 the DTD is provided in both, XML and SGML form. Instances following the SGML DTD are not case sensitive with respect to element names, attribute names and ids.

  Instances of the XML DTD are case senstitive. Since SGML tools usually deliver uppercase identifiers, the XML version of the DTD uses uppercase element identifiers also.

For further details see the SGML declaration given in Kapitel 2.3 The recommended SGML declaration S. 25.

26/08/2002 09:42:39 msrcc-eadoc.xml

Alle Rechte bei MSR MEDOC. Jede Verfügungserlaubnis, wie Kopier- und Weitergaberecht liegt bei uns.

2 Using the catalog for transmitting a revision history

In order to document change history of subjects, the following approach is recommended:

- Insert one **<ablock>** for each particular revision of a file or a group.
- Use different file names for each version. In particular, it is recommended to code the revision label into the file name such as *myfile-01-07.txt*.
- Place the version information into **<metadata>** as pointed out in this document.
- Add a **<field-set class="revision-history">** to each **<ablock>** and populate it with a **<field name="comment">** to submit the version comment.

2.1 Proposal for general utility routines

In order to handle the catalog, it might be appropriate to implement a with some utility rotines.

2.1.1 Check - Parser and checker

This routine will run the SGML-parser and apply a generic check routine which checks the constraints mentioned above such as:

- format of attributes such as (date)
- references
- values of attributes

2.1.2 CheckXML - Parse in XML and check

This routine runs the XML-parser and applay the generic check routine as mentioned in Topic 2.1.1 Check - Parser and checker S. 9.

2.1.3 Indent

This routine generates an indented version of the catalog. This is merely for the case that the catalog was written by a tool which did not perform pretty printing.

2.1.4 Compare

This routine compares the catalog with a reference catalog files and generates an updated version attributes for incremental data exchange (see Topic 1.2 Incremental data exchange S. 7) are populated. The basic principles are:

- the result is a combination of both catalogs, esp. the **<ablock>**s of both catalogs appear in order to show the now unused objects.
- the sequence of the objects is not taken into account when comparing the catalogs.
- the **<fields>** are not taken into account when comparing the catalogs since there is no mean to handle the results.
- The following strategy can be used to determine the **[upd]** attribute.[5]

[5]

26/08/2002 09:42:39 msrcc-eadoc.xml

Alle Rechte bei MSR MEDOC. Jede Verfügungserlaubnis, wie Kopier- und Weitergaberecht liegt bei uns.

NEW The object is newly introduced in the system. Therefore the contents file must be in the catalog (in particular, an **<file>** element must occur within **<ablock>**).

Detect: Check for all **<ablock>**s in the catalog: There is no **<ablock>** in reference catalog with the same **[name]** attribute.

referenced objects: NEW, MOVED, REUSED

REUSED The object was introduced in the system at former time but like NEW it is not in the reference catalog. Therefore the contents file must be in the catalog (in particular, an **<file>** element must occur within **<ablock>**).

Detect: Check for all **<ablock>**s in the catalog: There is no **<ablock>** in reference catalog with the same **[name]** attribute.

referenced objects: REUSED, MOVED, CHANGED

UNUSED The object is no longer used. The **<ablock>** is there in order to indicate, that the object can be removed in the configuration on the receiver's site. The **<ablock>** does not reference anything within the actual catalog.

Detect: Check for all **<ablock>**s in the reference catalog: There is no **<ablock>** in the catalog with the same **[name]** attribute. For the result, copy the **<ablock>** from the reference catalog, populate **[upd]** and drop all contents execpt **<metadata>**.

referenced objects: no referenced objects

UNCHANGED The object is unchanged. Therefore it must already be available on the receiver's site. Therefore it is possible to omit the file itself.

Detect: Check for all **<ablock>**s in the catalog: The same **<ablock>** appears with the same version information in the reference catalog.

referenced objects: UNCHANGED

CHANGED The object is changed. Therefore the contents file must be there (in particular, an **<file>** element) must occur within **<ablock>**.

Detect: Check for all **<ablock>**s in the catalog: The same **<ablock>** appears in the reference catalog but in another version.

referenced objects: CHANGED, UNCHANGED, REUSED, UNUSED, MOVED

MOVED The object is **unchanged** but in different Groups. Therefore it must already be available on the receiver's site. Therefore it is possible to omit the file itself.

The object is **changed** and in different Groups. In tis case it will be changed.

Detect: Check for all **<ablock>**s in the catalog: The same **<ablock>** appears with the same version information in the reference catalog but was referenced from different **<ablock>**.

referenced objects: MOVED

The following hints apply:

Alle Rechte bei MSR MEDOC. Jede Verfügungserlaubnis, wie Kopier- und Weitergaberecht liegt bei uns.

- The simple approach described here requires, that the name of objects is globally unique and that no two revisions of such an object appear simulatneously in the catalog. If this is not the case, much more complicated algorithms are required.
- If synchroneous versioning applies (using multiple **<cri>** elements), then obvously the revsion info of one particular company must be taken into account. The company is given as a runtime argument of the procedure.

  It could also be determined by a runtime argument, if the revision info of the **<metadata>** or the revision info in **<ablock>** is used.
- **<ablocks>** without revision information shall be flagged as errors.

2.1.5 Inspect

This invokes a graphical utility to inspect the catalog. Requirements are such as:

- The window has two panes, one for a catalog tree, and one for information about the actual node.
- The catalog is expanded, that is all **<ablock>**s are replaced by the referenced contents.
- It is possible to display the attributes of an ablock, the metadata or to invoke a tool displaying the contents of the referenced **<file>**.

Perhaps the other routines mentioned above can be invoked from this tool also.

2.1.6 copy - copy all files referenced in the catalog

This routine copies all files in the catalog to a directory specified as argument. It is invoked using the command:

The routine provides the following features:

- The output-directory is checked to be a directory
- The output-directory does not exist, it will be created
- The files in the catalog are checked to exist and to be plain files (no directories)
- If there are duplicates, a running number is added to the extension
- The messages are logged into the file *copy-mmx.log* in the current directory of the catalog[6]
- If the catalog does not contain a doctype declaration, a information message is submitted.

2.1.7 Import catalog into Data Management System

tbd

6

Alle Rechte bei MSR MEDOC. Jede Verfügungserlaubnis, wie Kopier- und Weitergaberecht liegt bei uns.

2.2 Elements and attributes

2.2.1 ABLOCK ... AREF

2.2.1.1 ABLOCK

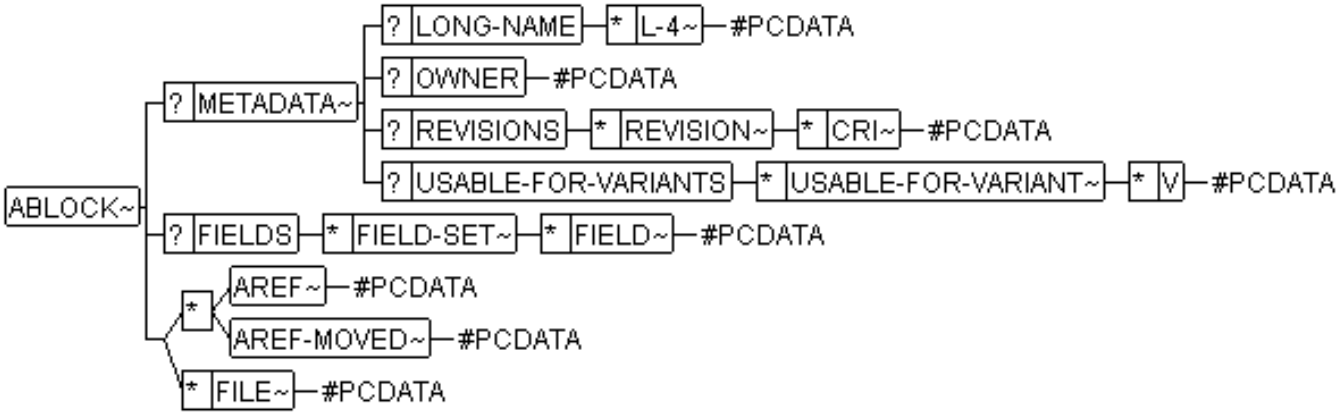

Abbildung 2: DTD-diagram for ABLOCK

Child elements **<metadata> <fields> <aref> <file>**

parent elements **<catalog>**

Tabelle 1: Attributes for ABLOCK

Name	Type	Class	Value	Remark
[CLASS]	name	implied		This denotes the class of the object. This class must be agreed upon for various applications.
[DATE]	cdata	implied		This reflects the date the object was last modified. This is for reference purposes only. It is useful, if the catalog is the only database used for metadata. Usually this attribute is omitted. The value must follow the patterns given in Topic 2.2.7.1 REVISION S. 23

26/08/2002 09:42:39 msrcc-eadoc.xml

Alle Rechte bei MSR MEDOC. Jede Verfügungserlaubnis, wie Kopier- und Weitergaberecht liegt bei uns.

Tabelle 1 (Forts.): Attributes for ABLOCK

Name	Type	Class	Value	Remark
[ID]	id	implied		This is a unique identifer of the object in the catalog. This must be used to refer to a particular version, if more than one version of the object is mentioned in the catalog.
[LABEL]	cdata	implied		This is the revision label string. It should be used, if the two step revisioning scheme provided by **[rev]** and **[var]** is not applied. The syntax of this attribute must be aggreed upon the involved parties.
[NAME]	cdata	required		This denotes the name of the object. As a rule, **[name]** and **[class]** must be unique within the catalog file[7]. This is the name of the object in the process e.g. the base name of a C-Module.
[REV]	cdata	implied		This specifies the revision of the object if the two layer revisioning scheme mentioned above is used.
[STATE]	cdata	implied		This denotes the status of the object such as ”released”, ”tested”. The content must be agreed upon the involved parties. It is primarily intedend to be used if the catalog is the primaray database.

7

26/08/2002 09:42:39 msrcc-eadoc.xml

Alle Rechte bei MSR MEDOC. Jede Verfügungserlaubnis, wie Kopier- und Weitergaberecht liegt bei uns.

Tabelle 1 (Forts.): Attributes for ABLOCK

Name	Type	Class	Value	Remark
[UPD]	nmtkgrp	implied	NEW UNUSED UNCHANGED CHANGED REUSED MOVED	The specifies the update status if incremental data exchange is performed using the catalog.
[VAR]	cdata	implied		This specifies the variant of the object if the two layer revisioning scheme mentioned above is used.

This represents assertions about a particular object. It comprises of metadata, fields. and the physical representation of the object itself.

The content of the object can be:

- a set of references (**<aref>**) to other objects thus establishing a configuration tree.
- a set of physical files referenced by **<file>** representing the object. This set may comprise of different file types with the same contents or the fact that the object is distributed across multiple files. Another method to achieve this, would be to pack these files using compressed archives such as ZIP.
- a set of database fields (**<field-set>**) containing the data for the object. Note that the database fields can be there in additon to the contents mentioned above.

Note that for compatibility reasons there is no wrapper around the elements representing the contents of the ablock.

The version infomation for an **<ablock>** can be given on two different ways:

- implicitly by using attributes such as **[rev]**, **[var]** etc. This is bound for use cases where all participiants are using the same versioning scheme.
- explicitly by using **<metadata>** which among others allows to use separate versioning schemes for each participiant.

Although possible, it is not intended to use both methods at the same time.

2.2.1.2 AREF

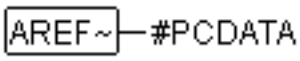

Abbildung 3: DTD-diagram for AREF

Child elements none

parent elements **<ablock> <catalog>**

Tabelle 2: Attributes for AREF

Name	Type	Class	Value	Remark
[CLASS]	name	implied		This is the class of the referenced **<ablock>**.
[IDREF]	idref	implied		This identifies the related **<ablock>** by its **[id]**.

Alle Rechte bei MSR MEDOC. Jede Verfügungserlaubnis, wie Kopier- und Weitergaberecht liegt bei uns.

Tabelle 2 (Forts.): Attributes for AREF

Name	Type	Class	Value	Remark
[SEQUENCE]	cdata	implied		This can take a sequence number which is used to express configurations where the sequence of elements is relevant (e. g. for documentation-fragments)
[VIEW]	nmtokens	implied		This allows to conditionalize the usage of the referenced ablock. As an example this could be used to distinguish different figures with respect to the intended audience (as it is done by *ASCET*).

This refers to another object (**<ablock>**). The reference can be done on two differents mechanisms:

- **[idref]** pointing to an **<ablock>** identified by **<id>**.
- **[class]** in conjunction with the content of ablock pointing to an **<ablock>** identified by **[class]** and **[name]**.

It is not intended to use both mechanisms simultaneously. If however both methods are used at the same time, they must both point to the same object. A reader must flag contradictions as an error, and may continue using ID/IDREF.

The ID based mechanism is there to support exchange of multiple versions of the same object.

2.2.1.3 AREF-MOVED

Abbildung 4: DTD-diagram for AREF-MOVED

Child elements none

parent elements **<ablock>**

Tabelle 3: Attributes for AREF-MOVED

Name	Type	Class	Value	Remark
[CLASS]	name	implied		This is the class of the referenced **<ablock>**.
[IDREF]	idref	implied		This identifies the related **<ablock>** by its **[id]**.

26/08/2002 09:42:39 msrcc-eadoc.xml

Alle Rechte bei MSR MEDOC. Jede Verfügungserlaubnis, wie Kopier- und Weitergaberecht liegt bei uns.

Tabelle 3 (Forts.): Attributes for AREF-MOVED

Name	Type	Class	Value	Remark
[VIEW]	nmtokens	implied		This allows to conditionalize the usage of the referenced ablock. As an example this could be used to distinguish different figures with respect to the intended audience (as it is done by *ASCET*).

This refers to another object which has been part of the actual ablock in a previous version (**<ablock>**). This specific reference is there to provide means to indicate changes within a configuration in greater detail. Otherwise a group can be marked with **[upd]** as *changed* but all components of this group are *unchanged* because one component is moved to somewhere else.

The reference can be done on two differents mechanisms:

- **[idref]** pointing to an **<ablock>** identified by **<id>**.
- **[class]** in conjunction with the content of ablock pointing to an **<ablock>** identified by **[class]** and **[name]**.

It is not intended to use both mechanisms simultaneously. If however both methods are used at the same time, they must both point to the same object. A reader must flag contradictions as an error, and may continue using ID/IDREF.

The ID based mechanism is there to support exchange of multiple versions of the same object.

2.2.2 CATALOG ... CRI

2.2.2.1 CATALOG

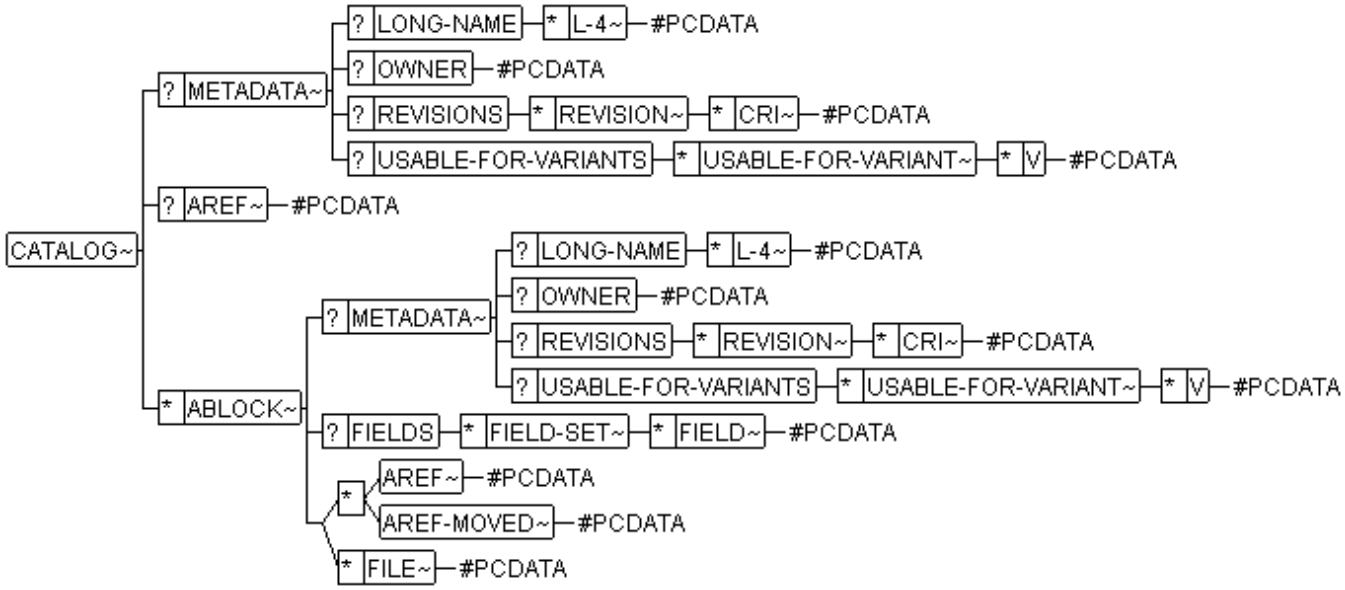

Abbildung 5: DTD-diagram for CATALOG

26/08/2002 09:42:39 msrcc-eadoc.xml

Alle Rechte bei MSR MEDOC. Jede Verfügungserlaubnis, wie Kopier- und Weitergaberecht liegt bei uns.

Child elements **<metadata> <aref> <ablock>**

parent elements none

Tabelle 4: Attributes for CATALOG

Name	Type	Class	Value	Remark
[CATALOG-CLASS]	name	implied		This specifies the particular use case for the catalog.
[CREATOR]	cdata	implied		This denotes the tool (incl. version) which created the actual catalog.
[MODIFICATION-DATE]	cdata	implied		This denotes the date, the catalog was last modified. If the catalog is generated from scratch it is the creation date. If the catalog is the primary database, the date of the last modification is given. The value must follow the patterns given in Topic 2.2.7.1 REVISION S. 23
[NAME]	cdata	required		This is the name of the catalog. It can be used for reference purposes.
[PUBID]	cdata	fixed	-//MSR//DTD container catalog 2.0 CNTCT-L0200.DTD//EN	This gives the public identifier of the catalog. A creator must place it according to the DTD.
[STATE]	cdata	implied		This denotes the state of the contents of the catalog if the catalog is the only database.
[VERSION]	name	implied		This denotes the version of the catalog structure. If omitted, it is 2.0

This is the root of the catalog. It comprises of metadata and assertion blocks. The metadata are related to the catalog itself.

If an entire system is transferrred, the root of the configuration can be denoted in the **<aref>** of **<catalog>**.

The use case of the catalog is denoted in **[catalog-class]** as:

program-kit	used to exchange entire program kits.
tool-kit	used to exchange tool kits

26/08/2002 09:42:39 msrcc-eadoc.xml

Alle Rechte bei MSR MEDOC. Jede Verfügungserlaubnis, wie Kopier- und Weitergaberecht liegt bei uns.

History-overview used to exchange an history overview of one particular work item.

The **[pubid]** attribute must have the value:

2.2.2.2 CRI

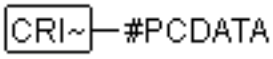

Abbildung 6: DTD-diagram for CRI

Child elements none

parent elements **<revision>**

Tabelle 5: Attributes for CRI

Name	Type	Class	Value	Remark
[COMPANY]	nmtoken	implied		This denotes the company for which the revision information is valid.
[LABEL]	cdata	implied		This is the label of the revision. Details see Topic 2.2.1.1 ABLOCK S. 11
[LABEL-P1]	cdata	implied		This is the label of the previous revision on the main line. Details see Topic 2.2.1.1 ABLOCK S. 11
[LABEL-P2]	cdata	implied		This is the label of the previous revision (secondary line) if the actual revision is the result of a merge process. Details see Topic 2.2.1.1 ABLOCK S. 11
[PRIVATE]	cdata	implied		This allows to keep private information from the particular company. It is necessary, that a receiver stores this information and returns it to the sender in order to allow a resynchronization. As an example, this could be some object identifier in the sender's *data management system* system.

26/08/2002 09:42:39 msrcc-eadoc.xml

Alle Rechte bei MSR MEDOC. Jede Verfügungserlaubnis, wie Kopier- und Weitergaberecht liegt bei uns.

Tabelle 5 (Forts.): Attributes for CRI

Name	Type	Class	Value	Remark
[REV]	cdata	implied		This is the revision number. Details see Topic 2.2.1.1 ABLOCK S. 11
[REV-P1]	cdata	implied		This is the revision number of the previous version on the main line. Details see Topic 2.2.1.1 ABLOCK S. 11
[REV-P2]	cdata	implied		This is the revision number of the previous version (secondary line) if the actual revision is the result of a merge process. Details see Topic 2.2.1.1 ABLOCK S. 11
[STATE]	cdata	implied		This is the state of the revision. Details see Topic 2.2.1.1 ABLOCK S. 11
[VAR]	cdata	implied		This is the variant designator. Details see Topic 2.2.1.1 ABLOCK S. 11
[VAR-P1]	cdata	implied		This is the variant name of the previous version on the main line. Details see Topic 2.2.1.1 ABLOCK S. 11
[VAR-P2]	cdata	implied		This is the variant name of the previous version (secondary line) if the actual revision is the result of a merge process. Details see Topic 2.2.1.1 ABLOCK S. 11

<cri> is used to specify company specific revision information. This is used for synchronization, if the participants of an exchange are using different versioning schemes.

2.2.3 FIELD ... FILE

2.2.3.1 FIELD

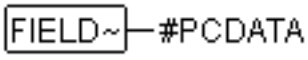

Abbildung 7: DTD-diagram for FIELD

26/08/2002 09:42:39 msrcc-eadoc.xml

Alle Rechte bei MSR MEDOC. Jede Verfügungserlaubnis, wie Kopier- und Weitergaberecht liegt bei uns.

Child elements none

parent elements **<field-set>**

Tabelle 6: Attributes for FIELD

Name	Type	Class	Value	Remark
[NAME]	cdata	implied		This is the name of the field.

This is a particular field within a field set (see Topic 2.2.3.2 FIELD-SET S. 20). Note that a particular field may also contain documentation information such as revision notes etc. For this purpose it is important to keep sure that the tools consider (and do not manipulate) white space[8].

2.2.3.2 FIELD-SET

Abbildung 8: DTD-diagram for FIELD-SET

Child elements **<field>**

parent elements **<fields>**

Tabelle 7: Attributes for FIELD-SET

Name	Type	Class	Value	Remark
[CLASS]	nmtoken	required		This denotes the particular use case for the field-set. As a rule it is the name of the application creating this set.

This is a set of application specific fields. This is intended to transmit arbitrary data which are kept in database records rather than in particular files. It is possible to submit multiple field sets to support multi tier applications where the catalog is read by more than one application.

The particular application is denoted in **[class]**.

2.2.3.3 FIELDS

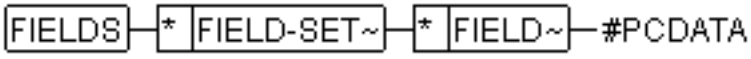

Abbildung 9: DTD-diagram for FIELDS

Child elements **<field-set>**

parent elements **<ablock>**

This allows to submit data which are not transferred as separate files. As of july 1999 it is intentionally not possible to define hierarchical data in order to keep it simple.

8

26/08/2002 09:42:39 msrcc-eadoc.xml

Alle Rechte bei MSR MEDOC. Jede Verfügungserlaubnis, wie Kopier- und Weitergaberecht liegt bei uns.

2.2.3.4 FILE

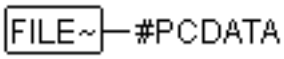

Abbildung 10: DTD-diagram for FILE

Child elements none

parent elements **<ablock>**

Tabelle 8: Attributes for FILE

Name	Type	Class	Value	Remark
[CREATION-DATE]	cdata	implied		This is the creation date of the file for reference purposes. It is not intended, that a container unpack routine must make sure that the files finally receive this time stamp. Usually this attribute is omitted[9]. The value must follow the patterns given in Topic 2.2.7.1 REVISION S. 23
[CREATOR]	cdata	implied		This denotes the createor of the file. The values must be agreed upon the involved participiants.
[ROLE]	nmtkgrp	implied	PRIMARY SECONDARY	The specifies the role of the particular file. Primary means, that this is the file being maintained. Secondary means this file is derived from the primary file. If this attribute is not there, all files are treated as primary.
[TYPE]	name	implied		This denotes the type of the file. The values must be agreed upon the involved participiants.

This points to the contents of the object represented as a physical file. The file is identified by the contents of **<file>**. It can be interpreted as simple file relative to the position of the catalog itself. It can also be interpreted as a universal ressource identifier.

9

26/08/2002 09:42:39 msrcc-eadoc.xml

Alle Rechte bei MSR MEDOC. Jede Verfügungserlaubnis, wie Kopier- und Weitergaberecht liegt bei uns.

If more than one **<file>** is there, all mentioned files belong to the ablock.

The attribute **[ROLE]** separates the primary files from those being derived from the primary files.

2.2.4 L-4 ... LONG-NAME

2.2.4.1 L-4

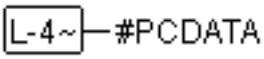

Abbildung 11: DTD-diagram for L-4

Child elements none

parent elements **<long-name>**

Tabelle 9: Attributes for L-4

Name	Type	Class	Value	Remark
[L]	cdata	implied		This denotes the language of the contents.

This is a language specific text. The language is given in the attribute **[l]**:

de german

en englisch

fr french

it italian

2.2.4.2 LONG-NAME

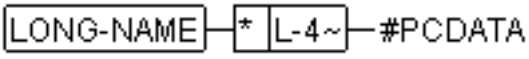

Abbildung 12: DTD-diagram for LONG-NAME

Child elements **<l-4>**

parent elements **<metadata>**

This specifies the long name of the object.

2.2.5 METADATA ... METADATA

2.2.5.1 METADATA

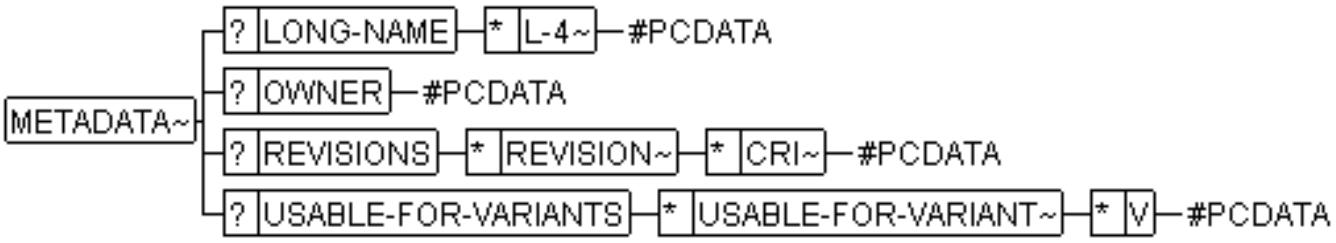

Abbildung 13: DTD-diagram for METADATA

26/08/2002 09:42:39 msrcc-eadoc.xml

Alle Rechte bei MSR MEDOC. Jede Verfügungserlaubnis, wie Kopier- und Weitergaberecht liegt bei uns.

Child elements **<long-name> <owner> <revisions> <usable-for-variants>**

parent elements **<ablock> <catalog>**

Tabelle 10: Attributes for METADATA

Name	Type	Class	Value	Remark
[IDENT]	nmtoken	implied		This is used to reidentify the object within the system which created the catalog.

<metadata> is used to carry exhaustive metadata about the object such as synchronized version numbers, variant information etc. This is complementary to the implied versioning information within **<ablock>** provided by attributs such as **[label]**.

2.2.6 OWNER ... OWNER

2.2.6.1 OWNER

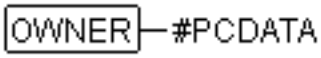

Abbildung 14: DTD-diagram for OWNER

Child elements none

parent elements **<metadata>**

This specifies the legal owner of the object. This is used to prevent the information to be used in illegal contexts. As an example, if the function specification was provided by a particular company, it must not be used in the context of another company.

2.2.7 REVISION ... REVISIONS

2.2.7.1 REVISION

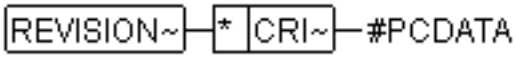

Abbildung 15: DTD-diagram for REVISION

Child elements **<cri>**

parent elements **<revisions>**

Tabelle 11: Attributes for REVISION

Name	Type	Class	Value	Remark
[DATE]	cdata	implied		This denotes the date, when the revision was issued. The value must follow the patterns given below.

26/08/2002 09:42:39 msrcc-eadoc.xml

Alle Rechte bei MSR MEDOC. Jede Verfügungserlaubnis, wie Kopier- und Weitergaberecht liegt bei uns.

This specifies revision information to the particular object. There may also be information about previous revisions in order to support resynchronization. In this case, the most actual revision information is the one for the actual object. The actual information can be identified using the attribute **[DATE]**.

[DATE] must follow one of the patterns in oder to provide a sequenctial order by performing simply a lexical sort.

The last pattern is the most preferred one, since it reflects a common use in US.

2.2.7.2 REVISIONS

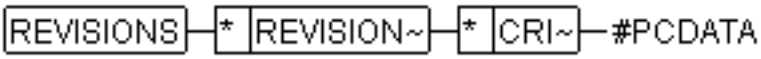

Abbildung 16: DTD-diagram for REVISIONS

Child elements **<revision>**

parent elements **<metadata>**

This is the explicit versioning (revisioning) information. It my comprise of the revision metadata of predecessors as well as of the actual object. The revision info of the actual object ist the one with the latest date denoted in **[date]** within **<revision>**.

2.2.8 USABLE-FOR-VARIANT ... USABLE-FOR-VARIANTS

2.2.8.1 USABLE-FOR-VARIANT

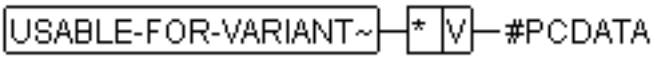

Abbildung 17: DTD-diagram for USABLE-FOR-VARIANT

Child elements **<v>**

parent elements **<usable-for-variants>**

Tabelle 12: Attributes for USABLE-FOR-VARIANT

Name	Type	Class	Value	Remark
[NAME]	cdata	implied		This is the name of the variant characteristic in question.

This specifies the possible values of the variant-criterium denoted by **[name]** the object in question can be applied to.

2.2.8.2 USABLE-FOR-VARIANTS

Abbildung 18: DTD-diagram for USABLE-FOR-VARIANTS

Child elements **<usable-for-variant>**

parent elements **<metadata>**

This denotes all the variants, the object in question can be used for.

26/08/2002 09:42:39 msrcc-eadoc.xml

Alle Rechte bei MSR MEDOC. Jede Verfügungserlaubnis, wie Kopier- und Weitergaberecht liegt bei uns.

2.2.9 V ... V

2.2.9.1 V

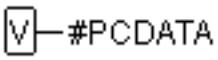

Abbildung 19: DTD-diagram for V

Child elements none

parent elements **<usable-for-variant>**

This is the particular value of the variant characteristic.

2.3 The recommended SGML declaration

The catalog can be used with *XML* and with *SGML*. In the second case, the following declaration is recommended.

26/08/2002 09:42:39 msrcc-eadoc.xml

Alle Rechte bei MSR MEDOC. Jede Verfügungserlaubnis, wie Kopier- und Weitergaberecht liegt bei uns.

Dokumentverwaltung

Tabelle : Beteiligte Personen

Name	Firma
Bernhard Weichel	MSR MEDOC
Oliver Rumpf	MSR MEDOC
Roman Reimer, XI-Works	MSR MEDOC

Tabelle : Versionsübersicht

Datum	Herausgeber
2002-02-07	Roman Reimer, XI-Works
12.7.2000	Bernhard Weichel

Tabelle : Änderungen

Änderung	Bezug
Erstellen von Indexen, Technischen Begriffen und Querverweisen. Konvertieren nach MSRREP V210 XML. Grund: Aufbereitung für Abschlußdokumentation	Inhalt
Working draft cicrulated within MSR-MEDOC Grund: as scheduled by MEDOC steering comitee	Inhalt

Tabelle : Enthaltene Änderungen

Datum	Kapitel	Änderung	Bezug
Nr. 1, 2002-02-07	Gesamt	Erstellen von Indexen, Technischen Begriffen und Querverweisen. Konvertieren nach MSRREP V210 XML. Grund: Aufbereitung für Abschlußdokumentation	Inhalt
Nr. 2, 12.7.2000	Gesamt	Working draft cicrulated within MSR-MEDOC Grund: as scheduled by MEDOC steering comitee	Inhalt

26/08/2002 09:42:39 msrcc-eadoc.xml

Alle Rechte bei MSR MEDOC. Jede Verfügungserlaubnis, wie Kopier- und Weitergaberecht liegt bei uns.