Notes:
This document is a deliverable of the OASIS TC – Format of Automotive Repair Information, and uses a standard document template. Not all deliverables are documents, but those that are should use this template to help with tracking and version management.

Deliverables from the project are assigned codes to allow tracking and reference of versions. A list of all coded deliverables and project documents will be maintained by the programme managers. The deliverables will be decided by the project sub-committees and assigned codes by the Programme Managers. The following convention is used for the coding:

<table>
<thead>
<tr>
<th>Deliverable Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SC1-Dx</td>
<td>Deliverables for sub-committee 1 – Use Cases and Requirements</td>
</tr>
<tr>
<td>SC2-Dx</td>
<td>Deliverables for sub-committee 2 – Architecture and Specification</td>
</tr>
<tr>
<td>SC3-Dx</td>
<td>Deliverables for sub-committee 3 – Terminology and Vocabulary</td>
</tr>
<tr>
<td>SC4-Dx</td>
<td>Deliverables for sub-committee 4 – Accessibility</td>
</tr>
</tbody>
</table>

Version numbers should be assigned starting with 1.0 and incremented with each new version circulated by the author(s). A version note should be added for each new version on page 2.

Please put the correct title on the front page and in the header on subsequent pages. The title field can be updated by selecting File|Properties and updating the Title field in the Summary tab. Then update the fields on the front page and header.

Please use heading styles Heading 1, Heading 2, etc for the titles of sections.

The table of contents on page 2 can be updated by right clicking and selecting Update.
OASIS Technical Committee - Format of Automotive Repair Information

Table: Version and Note

<table>
<thead>
<tr>
<th>Version</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>First version – draft for discussion by SC2 prior to TC Meeting on 14\textsuperscript{th} March 2003.</td>
</tr>
<tr>
<td>0.2</td>
<td>Version approved for discussion by members of the SC2 working group (now identified in the authors section of the document control)</td>
</tr>
</tbody>
</table>

Contents

1. Introduction ..................................................................................................................3
2. Definitions ....................................................................................................................3
   - Content ....................................................................................................................3
   - Meta Data ...............................................................................................................3
   - Namespace ..............................................................................................................4
   - Vocabulary ............................................................................................................4
   - Terminology ..........................................................................................................4
   - Ontology ................................................................................................................4
   - Taxonomy ..............................................................................................................4
3. Usage ............................................................................................................................5
4. Technical Framework ......................................................................................................5
   - 4.1 Overview ..........................................................................................................5
   - 4.2 Information Packages ......................................................................................6
   - 4.3 Meta Data Framework .....................................................................................6
5. Namespaces ...................................................................................................................8
6. Conformance Levels ......................................................................................................9
7. Vehicle and Package Identification .............................................................................10
8. Information Package Collections ................................................................................12
   - 8.1 Containers Defined by a URI Pattern ............................................................13
   - 8.2 Statements About Members of a Container ....................................................14
9. Combining Meta Data ...................................................................................................15
Appendix A. References ..................................................................................................17
Appendix B. Namespace Definitions ................................................................................18
Appendix C. Sample Meta Data .......................................................................................19
1. Introduction

This document is a first draft of the deliverable SC2-D5 Architecture and Specification from the OASIS Technical Committee – Format of Automotive Repair Information (OASIS Autorepair).

It is being circulated in order to elicit opinions and feedback, first from members of the SC2 sub-committee and then from the full technical Committee. It is not intended that this document should necessarily be taken forward as the basis for the final deliverable SC2-D5, but it is intended as a model for the form that deliverable may take.

The objective of the Technical Committee, as stated in its Charter, is:

To develop a standard format to enable access to emission-related repair, diagnostic and technical information with respect to the vehicles covered by the scope of Directive 70/220/EEC, i.e. passenger cars and light commercial vehicles.

The purpose of this deliverable is to specify the standard format referred to in the Charter.

2. Definitions

The following definitions provide the basis of the understanding of various terms used in this document.

Content

The text, graphics and other multimedia data that forms the body of information about a particular domain (in our case Automotive Repair Information).

Information Package (package)

A self-contained unit of content. The characteristics of an information package are described in the Autorepair Requirements Specification, Ref 2.5.2:

“Information must be available in chargeable units which are reasonable in comparison to the nature of the repair. When the manufacturer only holds historical information in hard copy format or (for example) a large PDF file, it may be reasonable to only provide the whole document.”

Meta Data

Meta data are data about data. That is:

- Data that describe other data
- Data that describe content

The term may also refer to any file or database that holds information about another database's structure, attributes, processing or changes.
Namespace

An XML namespace is a collection of names, identified by a URI reference[2], which are used in XML documents as element types and attribute names. (Definition taken from Namespaces in XML[3].)

Vocabulary

A vocabulary is a fixed set of names (terms), with a fixed meaning, that are used to describe a particular domain (in our case Automotive Repair Information).

When used in the context of XML, a vocabulary can be:

- A namespace
- The elements and attributes in an XML DTD or Schema
- The properties in an RDF collection

In this sense a vocabulary can be viewed as the terms used for metadata, rather than content.

Terminology

A set of terms (words and phrases), with an agreed definition for a domain, used in content. A full terminology may consist of a set of related entities:

- The base term
- A definition
- Synonyms (allowable alternative words or phrases for the same base term)
- Translations
- Variations (allowable in different contexts, e.g., parts of speech)

Ontology

An ontology is a shared set of terms describing an application domain which has a common understanding by agents in that domain.

Ontologies establish a joint terminology between members of a community of interest. These members can be human or automated agents.

An ontology consists of a vocabulary to represent the terms and a set of relationships between those terms which define the common understanding.

Taxonomy

A hierarchical classification of terms in a domain. A taxonomy can be viewed as a specialisation of an ontology in which the relationship defined between terms is class/sub-class.
3. Usage

The architecture specified here assumes that there are three types of ‘actor’ involved:

1. Information Producers
   - Are the original creators of emissions-related repair information. They are the manufacturers themselves.

2. Information Consumers
   - Any aftermarket organisation that requires access to emissions-related repair information and person entitled to see it.

3. Information Providers
   - Any party which takes original emissions-related repair information from Information Producers and delivers it on to Consumers, perhaps adding some value along the way (eg by putting all information into a common format).

Some manufacturers could be both Producers and Providers.

There are two main ways in which it is envisaged that the meta data described in this could be used:

   - Use Case One: To describe the information sought by Information Consumers in order to make a repair. This includes information about the vehicle given its VIN.
   - Use Case Two: To describe information packages made available by Information Producers and Provider.

4. Technical Framework

4.1 Overview

This specification defines the framework for a common meta data format, based on agreeing a terminology and representation, whereby all information content within the scope of the project can be described in a common way.
The content is split into a series of information packages which are described by collections of meta data.

4.2 Information Packages

The only requirement of the content is that each package must be uniquely identified by a URI and must be in one of the content formats defined in this specification (the properties core:textFormat and core:graphicFormat) or be viewable using a freely available browser plug-in.

The use of a URI to identify each information package does not imply that each package is freely available on the Internet.

4.3 Meta Data Framework

The technical framework used to express meta data for OASIS Autorepair is based on the W3C’s Resource Description Framework (RDF)[5]. RDF defines a framework by which resources can be described by meta data. A resource is defined as any discrete object that can be referenced by a URI. The main type of resource described in this Specification is the Information Package.

Using RDF, an item of meta data (the value of a property) can be expressed as a triple:

```
<schema:property>Value</schema:property>
```

The schema prefix indicates a namespace in which a collection of properties and resources are defined. This Specification defines a number a such namespaces. The properties defined in a namespace may be used to describe the information packages themselves, or other resources defined in the namespace.

For example, the property textFormat specifies the text format of an information package – it is a property of the package itself. That package may also be applicable to a vehicle of a particular model – vehicle is a resource with a property model.

Meta data describing a resource is contained in the RDF description element, whose about attribute specifies the URI of the resource being described.

```
<rdf:Description about="www.bmw.de/USP-EU-SBS"
    <core:textFormat>HTML</core:textFormat>
    <core:graphicFormat>PNG</core:graphicFormat>
    <vid:vehicle rdf:parseType="Resource">
        <vid:model>E46</vid:model>
        <vid:modelYear>2000</vid:modelYear>
    </vid:vehicle>
</rdf:Description>
```

This meta data describes the package identified by the URI www.bmw.de/USP-EU-SBS which is in HTML format with PNG graphics. The package relates to vehicles in the E46 model range for the model year 2000.
A property can take more than one value, in which case it can be repeated. So if the information package above contained both PNG and SVG graphics, it could be described:

```xml
<rdf:Description about="www.bmw.de/USP-EU-SBS">
  <core:textFormat>HTML</core:textFormat>
  <core:graphicFormat>PNG</core:graphicFormat>
  <core:graphicFormat>SVG</core:graphicFormat>
</rdf:Description>
```

When properties are used to describe the information sought by Information Consumers in order to make a repair (Use Case One) they are contained in rdf and rdf:description elements as follows.

```xml
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:ns1="URI of namespace 1"
  xmlns:ns2="URI of namespace 2"
  ...
  xmlns:nsN="URI of namespace N"
>
  <rdf:Description ref:about="www.autorepair.eu.com/query">
    <ns1:element>Meta data from namespace one</ns1:element>
    <ns2:element>Meta data from namespace two</ns2:element>
    ...
    <nsN:element>Meta data from namespace N</nsN:element>
  </rdf:Description>
</rdf:RDF>
```

When properties are used to describe information packages made available by Information Producers and Providers (Use Case Two), the rdf:about attribute is used to identify the target information package.

```xml
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:ns1="URI of namespace 1"
  xmlns:ns2="URI of namespace 2"
  ...
  xmlns:nsN="URI of namespace N"
>
  <rdf:Description rdf:about="URI of information package 1">
    <ns1:element>Meta data from namespace one</ns1:element>
    <ns2:element>Meta data from namespace two</ns2:element>
    ...
    <nsN:element>Meta data from namespace N</nsN:element>
  </rdf:Description>
</rdf:RDF>
```

The same file (RDF document) may contain meta data about more than one information package, as follows:

```xml
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:ns1="URI of namespace 1"
  xmlns:ns2="URI of namespace 2"
  ...
  xmlns:nsN="URI of namespace N"
>
```
5. Namespaces

The following namespaces are defined in this Specification.

<table>
<thead>
<tr>
<th>Namespace</th>
<th>Description</th>
<th>Requirements Covered</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core</td>
<td>Base information about the information packages, their format, language and information type.</td>
<td>2.1, 2.2, 3.9, (3.4)</td>
</tr>
<tr>
<td>Access</td>
<td>How to subscribe to and access the information packages</td>
<td>2.3 2.5, (3.4)</td>
</tr>
<tr>
<td>VehicleID</td>
<td>Meta data to identify the vehicle</td>
<td>3.1</td>
</tr>
<tr>
<td>General</td>
<td>Meta data describing general information on a vehicle</td>
<td>3.3</td>
</tr>
<tr>
<td>Component</td>
<td>Meta data describing information on a component</td>
<td>3.5.1</td>
</tr>
<tr>
<td>Vehicle</td>
<td>Meta data describing vehicle-specific information</td>
<td>3.4 and 3.5.4 to 3.5.8</td>
</tr>
<tr>
<td>Training</td>
<td>Meta data describing training information</td>
<td>3.6</td>
</tr>
<tr>
<td>Tools</td>
<td>Meta data describing special tools</td>
<td>3.7, 3.8</td>
</tr>
<tr>
<td>Navigation</td>
<td>Meta data on the relationships between information packages –</td>
<td>-</td>
</tr>
<tr>
<td>Namespace</td>
<td>Description</td>
<td>Requirements Covered</td>
</tr>
<tr>
<td>-----------</td>
<td>-----------------------------------------------------------------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>Terminology</td>
<td>Representing terminology in multiple languages, covering standard terminology recommended by the OASIS TC and non-standard terminology.</td>
<td>3.9</td>
</tr>
<tr>
<td>Diagnosis</td>
<td>Pass-through programming, Symptoms, DTC.</td>
<td>3.2.1 to 3.2.3, 3.5.2</td>
</tr>
</tbody>
</table>

### 6. Conformance Levels

This specification defines a set of meta data that can be applied to information packages. No assumption is made as to the degree to which future systems may use these meta data. However, it is recognised that not all systems will support the meta data to the same degree.

Hence the specification defines a series of conformance levels, numbered from 1 to 4, where 1 is the lowest level of conformance and 4 the highest. The levels are cumulative, in the sense that conformance at level 2 implies level 1 conformance as well.

Each item of meta data is assigned a conformance level, so that a system conforming, say, to level 3 would be able to handle all meta data designated as levels 1, 2 and 3.

<table>
<thead>
<tr>
<th>Conformance Level</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Meta data that could reasonably be expected to be generated automatically by most information producers and enable them to provide an Information Package Repository service.</td>
</tr>
<tr>
<td>2</td>
<td>Meta data considered necessary by information consumers to request information on emissions related repairs. The minimum meta data required to run a VIN Resolution Service and an Information Package Registry service.</td>
</tr>
<tr>
<td>3</td>
<td>Additional meta data that would be considered important by producers, consumers or providers to improve the quality or timeliness of repairs and enhance the usefulness of an Information Package Registry service.</td>
</tr>
<tr>
<td>4</td>
<td>All other meta data – this category of meta data could be considered ‘nice to have’.</td>
</tr>
</tbody>
</table>

Support at a particular conformance level does not imply that all information packages must be described by every piece of information at that level.

For example, if an information package was applicable to all BMW vehicles it would not be necessary to create meta data listing every model to which that package applied. Similarly, if a package applied to a particular model of vehicle it would not be necessary to list every derivative and trim level.
7. Vehicle and Package Identification

Any vehicle can be uniquely identified by a Vehicle Identification Number (VIN). The unique identifier may be different in different markets within the EU.

It is not expected that information producers (or others) would associate the VIN with each information package as meta data. Instead meta data would describe the vehicle make, model, etc.

Hence there is a requirement for a ‘VIN resolution’ service which returns basic information about a vehicle, given its VIN. This service would be independent of the meta data used to describe information packages, but could use the OASIS Autorepair standard in the way described below.

For information packages, a distinction can be made between a registry which records the information packages available and the meta data about them, and a repository which contains the information packages themselves. There are many scenarios that could be envisaged for the implementation of registries and repositories, but one such scenario is of one or more central registries (ie automotive information portals) and registries run by information producers and information providers, which just served the packages they published.

The diagram shows a sequence of messages between an information consumer and three Internet services:

1. VIN Resolution Service
   - Returns meta data about a vehicle, given its VIN.

2. Information Package Registry
   - Returns meta data about information packages, given a description of the type of information being sought.

3. Information Package Repository
   - Returns identified information packages, subject to necessary payment and access permissions.

Details of how these Internet services are implemented and how the messages are formulated, transported and interpreted are outside the scope of this Specification.

Each message that passes may include OASIS autorepair meta data as shown in the table below.
### VIN Resolution Service

1. Request with VIN

2. Return vehicle id information

3. Request with query

4. Return list of packages

5. Request information package(s)

6. Return information package(s)

---

#### Step

<table>
<thead>
<tr>
<th>Step</th>
<th>Meta data</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Information consumer finds VIN and sends request to VIN Resolution Service</td>
<td><code>&lt;rdf:Description about=&quot;www.autorepair.eu.com/query&quot;&gt; &lt;vid:vin&gt;ABC1234567DEF&lt;/vid:vin&gt; &lt;/rdf:Description&gt;</code></td>
</tr>
<tr>
<td>2. The VIN Resolution Service returns full details of that vehicle, including its make, model, derivative, fuel type, etc</td>
<td><code>&lt;rdf:Description about=&quot;ABC1234567DEF&quot;&gt; &lt;vid:vehicle rdf:parseType=&quot;Resource&quot;&gt; &lt;vid:model&gt;E46&lt;/vid:model&gt; &lt;vid:modelYear&gt;2000&lt;/vid:modelYear&gt; &lt;/vid:vehicle&gt; &lt;/rdf:Description&gt;</code></td>
</tr>
<tr>
<td>3. The information consumer describes the information package(s) they are looking for and this data is combined with the data from the VIN Resolution Service and sent to the Information Package Registry.</td>
<td><code>&lt;rdf:Description about=&quot;www.autorepair.eu.com/query&quot;&gt; &lt;core:infoType&gt;Repair&lt;/core:infoType&gt; &lt;vid:vehicle rdf:parseType=&quot;Resource&quot;&gt; &lt;vid:model&gt;E46&lt;/vid:model&gt; &lt;/rdf:Description&gt;</code></td>
</tr>
</tbody>
</table>
4. The Information Package Registry matches the description of the package being sought with meta data describing the available information packages and returns a list to the consumer.

5. The consumer selects the information package they want, examines the access information and requests the package from the Information Package Repository.

6. The Information Package Repository returns the information package to the consumer.

8. Information Package Collections

A key factor in making this specification implementable, is that information packages should be described usefully without an undue burden on the organisation producing the meta data. Key to this is that there should be some mechanism whereby meta data can be described for a collection of information packages, without having to be repeated for each one.

Where the same meta data can be attributed to more than one information package, the OASIS specification follows the RDF standard, which provides two mechanisms for assigning meta data with collections of resources (information packages).

Both methods use the RDF attribute aboutEach to specify that meta data applies to each member of a collection.

To illustrate the concepts of information package collections we will use six information sample packages from BMW. The first three packages have information on the engine coolant for all BMW vehicles; the other three packages have information on Car and Key Memory for E46 series vehicles.

All packages are available by subscription from BMW in HTML format in the English language. (The URIs used are for the purposes of illustration only).
<table>
<thead>
<tr>
<th>Package URI</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="http://www.bmw.de%5CUSP-EU-SBS%5CSBS1996-170196140_A3">www.bmw.de\USP-EU-SBS\SBS1996-170196140_A3</a></td>
<td>Approved anti-freezing and anti-corrosive agents</td>
</tr>
<tr>
<td><a href="http://www.bmw.de%5CUSP-EU-SBS%5CSBS1996-170196140_A2">www.bmw.de\USP-EU-SBS\SBS1996-170196140_A2</a></td>
<td>Change interval, changing and testing coolant, disposal, cleaning the cooling system</td>
</tr>
<tr>
<td><a href="http://www.bmw.de%5CUSP-EU-SBS%5CSBS1996-170196140_A1">www.bmw.de\USP-EU-SBS\SBS1996-170196140_A1</a></td>
<td>Requirements concerning engine coolant</td>
</tr>
<tr>
<td><a href="http://www.bmw.de%5CUSP-EU-SBT%5CSBT1998-610198293">www.bmw.de\USP-EU-SBT\SBT1998-610198293</a></td>
<td>Car and Key Memory</td>
</tr>
<tr>
<td><a href="http://www.bmw.de%5CUSP-EU-SBT%5CSBT1998-610198293%5C_ANL1">www.bmw.de\USP-EU-SBT\SBT1998-610198293\_ANL1</a></td>
<td>Summary of &quot;Car and Key Memory&quot; functions for E46 vehicles</td>
</tr>
<tr>
<td><a href="http://www.bmw.de%5CUSP-EU-SBT%5CSBT1998-610198293%5C_ANL2">www.bmw.de\USP-EU-SBT\SBT1998-610198293\_ANL2</a></td>
<td>Explanatory notes for the &quot;Car and Key Memory&quot;</td>
</tr>
</tbody>
</table>

The URIs for these six information packages show three levels of classification (taxonomy) for the information:

1. BMW
2. Function/component (ie Engine Coolant and Car & Key Memory)
3. Information package

**8.1 Containers Defined by a URI Pattern**

The first method is to use the structure of the URI to indicate the resource collections. The RDF Description element is used to refer to each level of the URI and any meta data that are specified at one level, apply to any resources at lower levels of the URI hierarchy.

```xml
<rdf:Description aboutEach="www.bmw.de">
  <core:textFormat>HTML</core:textFormat>
  <core:language>EN</core:language>
  <access:medium>Internet</access:medium>
  <access:chargingModel>subscription</access:chargingModel>
</rdf:Description>

<rdf:Description aboutEach="www.bmw.de\USP-EU-SBS">
  <term:term>Coolant</term:term>
</rdf:Description>

<rdf:Description aboutEach="www.bmw.de\USP-EU-SBT">
  <term:term>Car Memory</term:term>
  <term:term>Key Memory</term:term>
</rdf:Description>

<rdf:Description about="www.bmw.de\USP-EU-SBS\SBS1996-170196140_A3">
  <core:title>Approved anti-freezing and anti-corrosive agents</core:title>
</rdf:Description>

<rdf:Description about="www.bmw.de\USP-EU-SBT\SBT1998-610198293">
</rdf:Description>
Car and Key Memory
</core:title>
</rdf:Description>

8.2 Statements About Members of a Container

The second method is to explicitly declare the collection of information packages and then to create meta data that describes the collection. So an RDF sample that describes the six packages above may look like:

```xml
<rdf:Bag ID="bmw">
  <rdf:li resource="www.bmw.de\USP-EU-SBS\SBS1996-170196140_A3"/>
  <rdf:li resource="www.bmw.de\USP-EU-SBS\SBS1996-170196140_A2"/>
  <rdf:li resource="www.bmw.de\USP-EU-SBS\SBS1996-170196140_A1"/>
  <rdf:li resource="www.bmw.de\USP-EU-SBT\SBT1998-610198293"/>
  <rdf:li resource="www.bmw.de\USP-EU-SBT\SBT1998-610198293_ANL1"/>
  <rdf:li resource="www.bmw.de\USP-EU-SBT\SBT1998-610198293_ANL2"/>
</rdf:Bag>

<rdf:Bag ID="coolant">
  <rdf:li resource="www.bmw.de\USP-EU-SBS\SBS1996-170196140_A3"/>
  <rdf:li resource="www.bmw.de\USP-EU-SBS\SBS1996-170196140_A2"/>
  <rdf:li resource="www.bmw.de\USP-EU-SBS\SBS1996-170196140_A1"/>
</rdf:Bag>

<rdf:Bag ID="carandkeymemory">
  <rdf:li resource="www.bmw.de\USP-EU-SBT\SBT1998-610198293"/>
  <rdf:li resource="www.bmw.de\USP-EU-SBT\SBT1998-610198293_ANL1"/>
  <rdf:li resource="www.bmw.de\USP-EU-SBT\SBT1998-610198293_ANL2"/>
</rdf:Bag>

<rdf:Description aboutEach="#bmw">
  <core:textFormat>HTML</core:textFormat>
  <core:language>EN</core:language>
  <access:medium>Internet</access:medium>
  <access:chargingModel>subscription</access:chargingModel>
</rdf:Description>

<rdf:Description aboutEach="#coolant">
  <term:term>Coolant</term:term>
</rdf:Description>

<rdf:Description aboutEach="#carandkeymemory">
  <term:term>Car Memory</term:term>
  <term:term>Key Memory</term:term>
</rdf:Description>

<rdf:Description about="www.bmw.de\USP-EU-SBS\SBS1996-170196140_A3">
  <core:title>Aproved anti-freezing and anti-corrosive agents</core:title>
</rdf:Description>

An advantage of this method is that the information packages do not need to be classified in a strict hierarchy. So we could for example create another bag that defined a collection of information packages related to the E46 model, with members that intersected with members of other collections (ie a package can belong to more than one collection, organised as a full ontology, rather than a taxonomy):
<rdf:Bag ID="E46">
  <rdf:li resource="www.bmw.de\USP-EU-SBS\SBS1996-170196140_A3"/>
  <rdf:li resource="www.bmw.de\USP-EU-SBS\SBS1996-170196140_A2"/>
  <rdf:li resource="www.bmw.de\USP-EU-SBS\SBS1996-170196140_A1"/>
  <rdf:li resource="www.bmw.de\USP-EU-SBT\SBT1998-610198293"/>
  <rdf:li resource="www.bmw.de\USP-EU-SBT\SBT1998-610198293_ANL1"/>
  <rdf:li resource="www.bmw.de\USP-EU-SBT\SBT1998-610198293_ANL2"/>
</rdf:Bag>

<rdf:Description aboutEach="#E46">
  <vid:vehicle rdf:parseType="Resource">
    <vid:model>E46</vid:model>
  </vid:vehicle>
</rdf:Description>

9. Combining Meta Data

Meta data describing an information package may be combined from multiple sources. This allows many different scenarios for the implementation of this Specification, for example:

1. An information producer provides meta data at conformance level 1 and an information provider implements a registry service by adding meta data at conformance level 2.

2. An information producer supplies OASIS autorepair meta data at conformance level 2, and an information provider enhances those meta data with additional level 3 properties.

3. An information consumer adds their own properties at conformance level 4 which can be accessed and used only by persons authorised by that consumer.

This concept can be illustrated by considering in more detail the Information Package Registry introduced earlier.

The Registry services queries from consumers who are seeking information packages to help them repair a vehicle. The response to the query is information on how information packages can be accessed from a Repository.

The Registry also acts as an aggregation engine, collecting meta data from multiple sources and combining them together to build a complete description of information packages that can then be used to match more accurately the requirements of the consumer.

The figure shows a Registry service which aggregates meta data at various conformance levels from producers, providers and consumers.
Appendix A. References


[2] RFC2396

[3] Namespaces in XML. Eds Tim Bray, Dave Hollander, Andrew Layman
http://www.w3.org/TR/REC-xml-names


http://www.w3.org/TR/1999/REC-rdf-syntax-19990222

http://www.w3.org/TR/rdf-schema
Appendix B. Namespace Definitions

The definition of namespaces is currently ‘work in progress’ available as a spreadsheet. Details will appear in this appendix in the final specification.

[to be inserted]
Appendix C. Sample Meta Data

This sample shows a full set of meta data for the information package:

[to be inserted]