EC Spontaneous Grant
Contract SUB/99/502500

I-SEEC
Data Representation and API Specification
Report 200370/2000

Prepared for:
F. Callewaert, DG Enterprise, EC

Prepared by:
Dr R. Amor, A. Hutchison, Dr L. Newnham, BRE
G. Gudnasson, IBRI, J. Hyvarinen, VTT,
Prof. Z. Turk, IKPIR

29 May 2000

Final approval on behalf of BRE:

Signed _______________

Date _______________

BRE
Bucknalls Lane
Garston
Watford
WD2 7JR
UK

Tel: 01923 664000
Fax: 01923 664010
Email: enquiries@bre.co.uk

CONTENTS

INTRODUCTION

The I-SEEC project builds upon the results of the previous CONNET project. The gateway and services developed in I-SEEC will be presented on the Web as part of the Construction Information Service Network (CONNET). Consequently, the term CONNET will appear frequently throughout this report referring to central CONNET web nodes and associated CONNET web services; the term I-SEEC will be used only for the project itself rather that its outputs.

This report documents the developed data models and application protocol interfaces (APIs) for the CONNET thematic node as well as the seven services which sit inside this network. These data models and APIs provide the central representation required to implement the CONNET services. These data models and API form the core of the intellectual property right (IPR) developed in CONNET. Through the replication of these data models and APIs, new services of the same type as these seven could be established anywhere in Europe, and work seamlessly with the CONNET central services.

The initial part of this report covers the methods used to document developed data models and surveys existing standards, or recommendations, for structuring of meta-data for similar types of services.

The majority of this report is in the appendices, which detail the data models used by CONNET and each of the services. The models are shown in graphical form (EXPRESS-G), in their full textual form (EXPRESS) and then with basic descriptions of the semantics of the classes and their properties. The second half of the appendices provides similar detail for the API offered by CONNET and each of the services. This provides the definition of how services can interact with each other as well as how the central CONNET services can be accessed.

METHODS USED

I-SEEC has inherited its modelling formalism from the previous CONNET project. In the analysis and development of services for the CONNET thematic node a range of development methods have been used by each of the teams. While the CONNET consortium initially agreed to standardise on the unified modelling language (UML), previous expertise with EXPRESS and EXPRESS-G from ISO-STEP development (ISO 10303) led to this formalism being used to meet tight development deadlines. To provide a consistent documentation of CONNET in the appendices the EXPRESS and EXPRESS-G formalism (ISO 10303-11) have been used throughout to describe the data models.

RELEVANT META-DATA STANDARDS

This section reviews a range of meta-data 'standards' (either current standards or those under development) which impact on the types of services which are being created in I-SEEC.

ISO 10303 (STEP)

STEP is the abbreviation for the Standard for the Exchange of Product Model Data. It is defined by the International Standards Organisation (ISO) Technical Committee 184, Sub-Committee 4.

Work on STEP started in 1984. It resulted from the realisation that graphics exchange specifications would not be sufficient in the long term and that some of the problems that had been experienced with the existing Initial Graphics Exchange Specification could be handled more effectively by a more complete product data approach. Until 1989, efforts were focussed on the creation of a single, pan industry schema referred to as the Integrated Product Information Model (IPIM). However, the complexity of this approach was recognised and, as a result, the current Application Protocol based architecture was introduced. This is still the current STEP approach.

The purpose of STEP is ‘to specify a form for the unambiguous representation and exchange of computer interpretable information throughout the life of a product, enabling consistent implementations across multiple applications and systems’. It is concerned with the transfer of product data and also with the storage, access and archiving of such data and with ensuring that implementations can be tested for conformance.

There is currently only one part directly relevant to building construction requirements in STEP. This is part 225 (Building Elements Using Explicit Shape Representation). This currently has Final Draft International Standard (FDIS) status.

IAI-IFC

In August 1994, twelve companies set up a project to determine the potential for interoperability between different software applications based on the use of the (then) new ARX object technology which was being developed for AutoCAD release 13. In June 1995, they demonstrated the results of this project at the AEC Systems Show in Washington.

In carrying out the project, the partners realised that one of the most significant obstacles to interoperability was gaining a common understanding of the semantics and relationships of the classes that were being developed. They considered that this was an industry wide problem that related to more than one software platform. They therefore decided that what had been a ‘private club’ up to that point should be opened out to wider industry involvement and that other software vendors should be encouraged to participate.

The International Alliance for Interoperability (IAI) came into existence in North America in October 1995 and rapidly gained new members. Further Chapters were established in a number of other countries. A first international meeting of IAI Chapters was held in London in May 1996.

Membership of the Alliance is encouraged from all organisations who have an interest in the development and use of interoperable software in the building construction industry.

The purpose of the IAI is to define Industry Foundation Classes (IFCs) which enable the development of information exchange not only by means of file based exchange and sharing of database repositories as in STEP but also by promoting the development of interoperable software applications which use the newer technology of client/server interfacing. This is expressed as an ideal in the IFC Specification Development Guide.

Development of the IFC Object Model draws extensively on model schema that form part of the STEP standard. In particular, the following parts are used extensively:

- ISO 10303 part 11 (EXPRESS) defines the data definition language in which the formal IFC specification for implementation is delivered;
- ISO 10303 part 21 defines the physical format of files used for data exchange;
- ISO 10303 part 22 defines access to databases that store IFC based information;
- ISO 10303 parts 41, 42 and 43 specify a number of model schemas that are used directly within IFCs or that are modified for IFC use.

Many of the concepts developed in the IFCs over versions 1.5.1 through to 2.0 have been incorporated into the CONNET system. Special mention must be made of the property sets for dynamic specification of product properties and the core organisation, and classification definitions.

Dublin Core

"The Dublin Core Metadata Workshop Series began in 1995 with a workshop which brought together, by invitation, librarians, digital library researchers, content experts, and text-markup experts to promote better discovery standards for electronic resources. The Dublin Core is a 15-element set of descriptors that has emerged from this effort in interdisciplinary and international consensus building." (Weibel et al, 1998). The consensus was reached on the semantics of 15 elements.

The metadata elements fall into three groups which roughly indicate the class or scope of information stored in them: (1) elements related mainly to the Content of the resource, (2) elements related mainly to the resource when viewed as Intellectual Property, and (3) elements related mainly to the Instantiation of the resource.

```
        Content          Intellectual Property       Instantiation
        -----------      ---------------------       -------------
        Title                 Creator                  Date
        Subject               Publisher                Format
        Description           Contributor              Identifier
        Type                  Rights                   Language
        Source
        Relation
        Coverage
```

Dublin Core is designed as a means for 'publishers' and authors to provide metadata at the point of mounting information on the Web. It is in the interest of these publishers to make metadata available that can be harvested by commercial and selective search services as a means to ensure their publications become publicised.

The standard for the Dublin core defines semantics only. Different syntaxes are proposed for encoding Dublin core in text files and more importantly in HTML and XML. The proposed way of encoding Dublin Core metadata in HTML is by using the META tags like this (http://www.roads.lut.ac.uk/Metadata/DC-Proposal.html):

```
<META NAME="DC.Title" CONTENT="Moby Dick">
```
Note: all names of Dublin Core fields are prepended with a DC.

Linux Software Map (LSM) templates

The Linux software archives at the SunSITEs addressed their need for a metadata standard with structured templates that contain the following 12 attributes appropriate to the archive needs:

- Title
- Version
- Entered-date
- Description
- Keywords
- Author
- Maintained-by
- Primary-site
- Alternate-site
- Original-site
- Platforms
- Copying-policy

The form of the entries is similar to Internet Mail headers with colon separated attribute-value pairs (RFC822) that can wrap over several lines. There is a short description of the valid values for each field but little concrete data form, most of it is free form text. Later on, tools were built to process these templates, index them and create such things as the Linux Software Map. At present, when people are submitting something to the archive, they may be rejected by the maintainers if they do not have LSM templates written. LSM templates are intended for software packages that are replicated at different sites and not particularly appropriate for indexing a much richer set of files.

IAFA templates

The Internet Engineering Task Force (IETF) Working Group on Internet Anonymous FTP Archives (IAFA), later called IIIR, have produced the IAFA templates Internet Draft (Becket 1995). This defines a range of indexing information that can be used to describe the contents and services provided by anonymous FTP archives. The draft has a rich range of templates, attributes and values that can be used to describe common and useful elements. The goal is that these are to be used to index archives, made available publicly in them to allow searching, indexing and sharing of information on the archive contents, services and administrative data.

This template scheme is based on the same RFC822 form like the LSM templates, with colon separated attributes-values known as data elements. One or more data elements are collected into templates that have a single Template-Type field to describe the type of the basic template. Multiple templates can be collected in index files by separating with blank lines. The attributes can be structured in several ways: Variant information which is used to support multiple languages, formats, etc of a document, for example: Language-v0: English and Language-v1.

There are 14 currently defined template types: SITEINFO, LARCHIVE, MIRROR, USER, ORGANIZATION, SERVICE, DOCUMENT, IMAGE, SOFTWARE, MAILARCHIVE, USENET, SOUND, VIDEO, FAQ and each has appropriate attributes defined for them. Most of the types are self explanatory apart from SITEINFO, which is a description of the FTP site and LARCHIVE, which is a description of a logical (sub-)archive. More types can be defined if necessary having the same basic attributes as DOCUMENT. It also turns out that LSM Templates were based on an early draft of the IAFA Templates (June 1992) but modified to have more consistent elements.

An example for a software template:

```
Template-Type:          SOFTWARE

Title:                  Beethoven's Fifth Player

Version:                67

Author-Name:            Ludwig Van Beethoven

Author-Email:           beet@romantic.power.org

Author-Work-Fax:        +43 1 123 4567

Admin-Handle:           berlioz01

Description:            The program provides the novice
                        to Transitional Classical-
                        Romantic music a V-window inter-
                        face to the author's latest com-
                        position

Requirements:           Requires the V-Window system
                        version 10 or higher

Discussion:             USENET rec.music.classical

Copyright:              Freely redistributable for non-
                        commercial use.  Copyright held
                        by author

Keywords:               Classical music, V-windows

Format:                 LZ compressed

URI:                    gopher://power.org/00/pub/Vfifth.tar.Z
```

SOIF / RDM

The SOIF (Summary Object Interchange Format) is a record format used by the Harvest software. Harvest software was developed at the University of Colorado at Boulder, and is distributed by them as shareware. It is documented at http://harvest.cs.colorado.edu/Harvest/.

Most SOIF records are generated by robots, although as they are based on simple attribute:value pairs they can easily be generated by hand. SOIF records can also be used as an aid to creation of other metadata formats. A broker can support different attributes, depending on the data it holds. Often brokers will hold the full text of documents as well as metadata. A list of common attributes is provided in the documentation as follows: Bibliographic type attributes:

- Abstract
- Author
- Description
- Title
- Type
- URL
- File Size
- Full-Text
- Keywords

- Last-Modification-Time.

Administrative metadata attributes:

- Gatherer-Host
- Gatherer-Name
- Gatherer-Version
- MD5 (checksum)
- Refresh-Rate
- Time-to-Live
- Update-Time.

Harvest has been widely taken up within the academic community as a basis for search services. Of significant importance has been the recent adoption of Harvest technologies by Netscape. In 1996 Netscape announced they would use SOIF as a basis for their Catalog Server product. In a significant extension to the Harvest architecture, Netscape are working on 'Resource Description Messages' which provide a framework for the creation and communication of metadata. Resource Description Messaging (RDM) is a messaging format, which can be used as the basis of a query syntax. RDM evolved into RDF which is addressed in the next section.

RDF

RDF is the Resource Description Framework proposed by Netscape as an open industry standard for describing how metadata for content is defined in web documents. This metadata is descriptive information about the structure and content of information in a document. RDF will be an application of XML. Many companies that provide information, like ABC News, CNN and Time Inc support the RDF proposal. The search engines like AltaVista, Yahoo and Webcrawler also support RDF.

"The Resource Description Framework (RDF) is a foundation for processing metadata; it provides interoperability between applications that exchange machine-understandable information on the Web. RDF emphasises facilities to enable automated processing of Web resources. RDF can be used in a variety of application areas; for example: in resource discovery to provide better search engine capabilities, in cataloguing for describing the content and content relationships available at a particular Web site, page, or digital library, by intelligent software agents to facilitate knowledge sharing and exchange, in content rating, in describing collections of pages that represent a single logical "document", for describing intellectual property rights of Web pages, and for expressing the privacy preferences of a user as well as the privacy policies of a Web site. RDF with digital signatures will be key to building the "Web of Trust" for electronic commerce, collaboration, and other applications." (http://www.w3.org/RDF/)

As implied by the name, RDF does not define a set of attributes with which to label a resource on the Web, but defines a framework (object model, language, syntax, guidelines) for how such attribute sets can be defined. In this respect it is close to information modelling languages, such as EXPRESS or NIAM.

RDF uses XML to encode both schema level and object level information. Below is part of the definition of the Dublin Core using RDF. Note the definition of the TITLE element.

```
<?xml version='1.0'?>
               <rdf:RDF
                  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
                  xmlns:rdfs="http://www.w3.org/TR/1999/PR-rdf-schema-
19990303#"
                  xmlns:dc="">

               <rdf:Description about = "">
                 <dc:Title> The Dublin Core Element Set </dc:Title>
                 <dc:Creator>   The   Dublin   Core   Metadata   Inititative
</dc:Creator>
                 <dc:Description> The Dublin Core is a simple metadata
element
                     set intended to facilitate discovery of electronic
                     resources. </dc:Description>
                 <dc:Date> 1995-03-01 </dc:Date>
               </rdf:Description>

               <rdf:Description ID="Title">
                 <rdf:type  rdf:resource="http://www.w3.org/1999/02/22-rdf-
syntax-ns#Property"/>
                 <rdfs:label>Title</rdfs:label>
                 <rdfs:comment>The name given to the resource, usually by
the Creator
                 or Publisher.</rdfs:comment>
                 <rdfs:isDefinedBy rdf:resource = ""/>
               </rdf:Description>
```

Other less relevant meta-data developments

ISO 7200:1984

ISO 7200:1984 Technical documentation - Document headers and title blocks. This standard covers drawings, specifications, technical drawings, and title blocks in CAD drawings.

ISO 11442:1993

ISO 11442:1993 Technical documentation. Looks at the handling of technical based computer information.

IEC 82 045

IEC 82 045 Management data (Metadata) associated with technical documents.

MAJOUR SGML DTD

This standard is for journal article headers. MAJOUR (Modular Application for Journals) was developed by a European Workgroup on SGML linked to the Scientific, Technical and Medical Publishers group, Amsterdam, and published in 1991. The group consisted of Elsevier Science Publishers, Kluwer, Springer etc. No full consensus has been reached among journal publishers on an agreed DTD for article headers. Journal publishers apparently have not been able to agree on MAJOUR as anything more than a basis for individual variants.

SSSH SGML DTD

This standard is for article headers. SSSH (Simplified SGML for Serial Headers) is used by journal publishers in all fields, particularly for the purpose of communication from publishers to users. The aim of the SSSH is to harmonise MAJOUR with OASIS' requirement for a simpler set of elements.

BIC SGML DTD

This standard is for non-serial publications. Intended for use throughout the "book" supply chain where detailed bibliographic, trade and promotional information must be communicated. Although this is commonly referred to as a book DTD it is intended to cover a wide range of non-serial electronic and print publications. The intention is for the DTD to accommodate a variety of media, but that the first version should be targeted at printed material.

EDIFACT EDI formats

EDIFACT is the accepted international standard messaging format for trading transactions in all industries. It is appropriate where bibliographic and other product information is communicated in the context of a trading relationship. EDIFACT is a mature and fully accepted standard although its practical application in the "book" trade has begun only in 1995/6. The EDIFACT syntax is maintained and developed by a world-wide process co-ordinated by a UN agency. The "book" trade has chosen to adopt an EDIFACT subset maintained by EAN International (EANCOM).

REFERENCES

Heery, Rachel (1996). BIBLINK - LB 4034 D1.1 Metadata Formats, http://hosted.ukoln.ac.uk/biblink/wp1/d1.1/doc0000.htm

Weibel, S., J. Kunze, C. Lagoze, M. Wolf (1988). Dublin Core Metadata for Resource Discovery, RFC 2413, http://sunsite.cnlab-switch.ch/ftp/doc/standard/rfc/24xx/2413

APPENDIX A: Schema CONNET_Core

The CONNET core schema defines data structures for two areas. The first are common data structures which can be used by all applications attaching to the CONNET system. The second are data structures specific to the CONNET node which enable the central services to be maintained and offered to CONNET users and services alike.

EXPRESS-G Diagrams

Figure 1Services

Figure 2 Organisations

Figure 3 Dublin Core

Figure 4 Orders

Figure 5 Classification and Property Sets

Figure 6 Users

TYPE: Action

Describes possible actions a user would make inside a service, or the CONNET node, which would be tracked by the system to provide summary statistics of the usage of services and CONNET itself.

EXPRESS specification:
```
TYPE Action = ENUMERATION OF
       (AddedInformation,
        UsedTechnicalAdvisory,
        ExtendedQuery,
        NavigatedTo,
        UsedHelpDesk,
        QueryBetweenServices,
        SearchResult,
        SearchString,
        SentToSite);
END_TYPE;
```

TYPE: Association

Defines the possible connections a service would have with CONNET. This allows CONNET to determine what interactions would be possible with the service.

EXPRESS specification:
```
TYPE Association = ENUMERATION OF
      (CONNETService,
       CONNETInterface,
       Linked,
       Associated,
       Source);
END_TYPE;
```

TYPE: Charging

Defines the possible charging mechanisms of a service when determining whether to pass a user through to a particular service.

EXPRESS specification:
```
TYPE Charging = ENUMERATION OF
      (Free,
       Commercial,
       Membership);
END_TYPE;
```

TYPE: Country

Defines all possible countries (not shown here for brevity). This is based on the ISO 3166 three letter country code.

EXPRESS specification:
```
TYPE COUNTRY = ENUMERATION OF
      ();
END_TYPE;
```

TYPE: CreditCards

Defines all major credit cards.

EXPRESS specification:
```
TYPE CREDITCARDS = ENUMERATION OF
      (Amex,
       Diners,
       Discovery,
       Enroute,
       JCB,
       MasterCard,
       Switch,
       Visa,
       Unclassified);
END_TYPE;
```

TYPE: Currency

Defines the currency used in a particular service (not shown here for brevity). This is based on the ISO 4217 currency specification.

EXPRESS specification:

```
TYPE CURRENCY = ENUMERATION OF
      ();
END_TYPE;
```

TYPE: DeliveryMode

Defines the mode that a user would like information delivered to them for a particular user profile.

EXPRESS specification:
```
TYPE DeliveryMode = ENUMERATION OF
      (Email,
       Letter,
       Fax,
       Phone,
       Web);
END_TYPE;
```

TYPE: ItemStatus

Defines the status of any order item.

EXPRESS specification:
```
TYPE ORDERSTATUS = ENUMERATION OF
      (cancelled,
       charged,
       creditCardRefund,
       selected,
       ordered,
       delivered,
       rejected);
END_TYPE;
```

TYPE: Language

Defines the language that a site is written in (not shown here for brevity). This is based on the ISO 639 language codes.

EXPRESS specification:
```
TYPE LANGUAGE = ENUMERATION OF
      ();
END_TYPE;
```

TYPE: OrderStatus

Defines the status of any order.

EXPRESS specification:
```
TYPE ORDERSTATUS = ENUMERATION OF
      (cancelled,
       charged,
       creditCardRefund,
       selected,
       ordered,
       delivered,
       rejected);
END_TYPE;
```

TYPE: ServiceType

Defines the type of service provided. Currently this reflects the seven services supported in I-SEEC as well as the CONNET node service.

EXPRESS specification:
```
TYPE ServiceType = ENUMERATION OF
      (BPC,
       CSC,
       ESC,
       NS,
       TIC,
       WEC,
       WWC,
       CONNET);
END_TYPE;
```

TYPE: StandardResourceTypes

Defines the standard resource types as in Dublin Core for published material.

EXPRESS specification:
```
TYPE StandardResourceTypes = ENUMERATION OF
      (Advertisement,
       Article,
       Bibliography,
       Book,
       Booklet,
       Collection,
       CourseMaterial,
       Dataset,
       HonoursThesis,
       Image,
       InBook,
       InCollection,
       InProceedings,
       Journal,
       Magazine,
       Manual,
       MastersThesis,
       MessageOnModeratedMailingList,
       MessageOnUnmoderatedMailingList,
       Misc,
       Music,
       Newspaper,
       OrganisationInfo,
       PhDThesis,
       PersonalInfo,
       Poem,
       PostingToModeratedNewsgroup,
       PostingToUnmoderatedNewsgroup,
       Preprint,
       Proceedings,
       ResearchPaper,
       Service,
       TechReport,
       Unpublished,
       UnrefereedArticle,
       Video);
END_TYPE;
```

TYPE: TimePeriod

Defines possible time periods against which a user would like information delivered for their particular user profile.

EXPRESS specification:
```
TYPE TimePeriod = ENUMERATION OF
      (DAY,
       WEEK,
       FORTNIGHT,
       MONTH,
       TWOMONTHS,
       QUARTER,
       HALFYEAR,
       YEAR);
END_TYPE;
```

ENTITY: AvailabilityStatistics

Allows CONNET to capture the down-time of every service to build up a log of availability statistics for every service and the system as a whole.

EXPRESS specification:
```
ENTITY AvailabilityStatistics;
      FailureType : STRING;
      StartTime : DATETIME;
      Duration : INTEGER;
 UNIQUE
      UR1:  FailureType;
      UR2:  StartTime;
END_ENTITY;
```

Attribute definitions:
FailureType: defines the type of failure which was experienced. Currently power, hardware, and software are expected.
StartTime: captures the time at which the failure started.
Duration: captures the duration of a failure in minutes.

ENTITY: ClassificationCode

Class representing a single classification code according to the related classification system and table in that system. Each instance of ClassificationCode must have a unique combination of Code attribute value and referenced CassificationTable instance.

EXPRESS specification:
```
ENTITY ClassificationCode;
      Code : STRING;
      SearchableClasses : SET [0:?] OF ObjectClass;
      Description : OPTIONAL STRING;
      SuperCode : OPTIONAL ClassificationCode;
      Table : ClassificationTable;
 UNIQUE
      UR1:  Code;
      UR2:  Table;
END_ENTITY;
```

Attribute definitions:

Code: the full code for the particular classification.

SearchableClasses: provides a link to named ObjectClasses which could be associated with the defined classification code.

Description: is the full text description of the code as provided by the classification system specifier.

SuperCode: links to the parent of a particular code.

Table: specifies which table within a classification code that the individual code belongs to.

ENTITY: ClassificationSystem

Class representing a complete classification system. Each instance of ClassificationSystem must have a unique Id (in the scope of CONNET services).

EXPRESS specification:
```
ENTITY ClassificationSystem;
      Name : STRING;
      Description : OPTIONAL STRING;
      Language : LANGUAGE;
 INVERSE
      Tables : SET[1:?] OF ClassificationTable FOR System;
 UNIQUE
      UR1:  Name;
END_ENTITY;
```

Attribute definitions:

Name:the common name of the classification system (e.g., UNICLASS, EPIC, CI/SfB).

Description: the full text description of the classification system and its uses as supplied by the classification system specifier.

Tables: the set of tables that exist in the classification system. The majority of systems are table based. Where a classification system has no tables then there will be a single dummy entry in the table field to link through to all classification codes.

ENTITY: ClassificationTable

Class representing a unique classification table in the related classification system. Each instance of Classification table must have a unique combination of TableId attribute value and referenced CassificationSystem instance.

EXPRESS specification:
```
ENTITY ClassificationTable;
      System : ClassificationSystem;
      Table : STRING;
      Description : OPTIONAL STRING;
 INVERSE
      Codes : SET[1:?] OF ClassificationCode FOR Table;
 UNIQUE
      UR1:  System;
      UR2:  Table;
END_ENTITY;
```

Attribute definitions:
System: provides the link to the classification system.
Table: defines the name of the table in the classification system.
Description: description of the table.
Codes: points to all the codes which exist in the named table.

ENTITY: CommerceUser

Parent KnownUser captures information about users of the CONNET system and its services. If products or services are bought then billing and delivery details need to be captured.

EXPRESS specification:
```
ENTITY CommerceUser
      SUBTYPE OF (KnownUser);
      BillingDetails : Organisation;
      DeliveryDetails : Organisation;
END_ENTITY;
```

Attribute definitions:

BillingDetails: who to bill.
DeliveryDetails: who to deliver to.

ENTITY: CONNET

Defines the CONNET system at its top level. This is the starting point to access all information about CONNET and the services it provides.

EXPRESS specification:
```
ENTITY CONNET
      ABSTRACT SUPERTYPE OF (ONEOF(CONNETNational,CONNETMultiNational));
      Owner : Organisation;
      HelpDeskURL : STRING;
      TechnologyObservatoryURL : STRING;
      SystemName : STRING;
      SystemID : INTEGER;
      Certificate : STRING;
END_ENTITY;
```

Attribute definitions:
HelpDeskURL: points to the helpdesk system which is utilised by CONNET.
TechnologyObservatoryURL: points to the technology observatory system which is maintained as part of the CONNET core services.
Users: points to the set of known users in the CONNET system. This will be a mix of registered users (really known users) and unique accesses to the CONNET system against which user details have not been gathered (though they may be gathered in the future).
Owner: points to general information of the owner/maintainer of the CONNET system as well as all methods of contact for that person.
SystemName: Name for future expansion to differentiate between CONNET systems.
SystemID: ID for future expansion to differentiate between CONNET systems.

Certificate: contains the certificate information for the service used to establish a secure connection with the service for the transfer of sensitive information.

ENTITY: CONNETCommerceService

CONNETCommerceService is a sub class of CONNETService, it contains extra information required for commerce services.

EXPRESS specification:
```
ENTITY CONNETCommerceService
      SUBTYPE OF (CONNETService);
      PurchasingEnabled : BOOLEAN;
      CurrencyUsed : CURRENCY;
      Users : SET [1:?] OF CommerceUser;
      Orders : SET [0:?] OF UserOrder;
END_ENTITY;
```

Attribute definitions:
PurchasingEnabled: flag to enable purchasing.
CurrencyUsed: name of currency.
Users: Set of registered users.
Orders: Set of all orders made through service.

ENTITY: CONNETMultiNational

This is an international gateway for the national gateways. There may be many of these.

EXPRESS specification:
```
ENTITY CONNETMultiNational
      SUBTYPE OF (CONNET);
      NationalGateways : SET [1:?] OF CONNETNational;
END_ENTITY;
```

Attribute definitions:
NationalGateways: A set of national gateways.

ENTITY: CONNETNational

This is the usual entry point to the system, being a national gateway for services in that country.

EXPRESS specification:
```
ENTITY CONNETNational
      SUBTYPE OF (CONNET);
      Services : SET [1:?] OF CONNETService;
      CountryServed : COUNTRY;
      Language : LANGUAGE;
      Users : SET [1:?] OF KnownUser;
 INVERSE
      ParentGateways : SET[1:?] OF CONNETMultiNational FOR
NationalGateways;
END_ENTITY;
```

Attribute definitions:

Services: points to the set of services which are currently in operation under the CONNET umbrella.
CountryServed: country the gateway is situated in.
Language: Main language used in services.
Users: Set of system users.
ParentGateways: Set of multinational gateways that point to this national gateway.

ENTITY: CONNETService

This is the top level definition of an individual service in the CONNET system. This captures enough information about the service in order to allow CONNET to pass users through to perform particular functions. It also allows CONNET to link up related services, for example to share information across similar services in different countries.

EXPRESS specification:
```
ENTITY CONNETService
      SUPERTYPE OF (CONNETCommerceService);
      ServiceName : STRING;
      URL : STRING;
      Country : COUNTRY;
      ServiceType : ServiceType;
      Parent : OPTIONAL CONNETService;
      ServiceLevel : STRING;
      Owner : Organisation;
      Tables : SET [0:?] OF TableOfCodes;
      Classifications : SET [0:?] OF ClassificationSystem;
      UpTime : SET [1:?] OF AvailabilityStatistics;
      HelpDeskURL : STRING;
      ElecEntryURL : STRING;
      Certificate : STRING;
      Language : LANGUAGE;
      ServiceID : INTEGER;
      Users : SET [1:?] OF KnownUser;
      AssociationStatus : Association;
      ChargingStrategy : Charging;
      SimpleQueryURL : STRING;
      QueryReloadURL : STRING;
 INVERSE
      System : CONNETNational FOR Services;
 UNIQUE
      UR1:  ServiceID;
END_ENTITY;
```

Attribute definitions:
ServiceName: the name of the individual service as specified by the service developer.
URL: the starting location for the service through the Internet.
Country: the country which the service addresses.
ServiceType: the type of service provided as categorised by the initial CONNET developed services.
Parent: a higher level service which can be contacted for answers if this service can not provide a satisfactory result. This allows the connection from organisation based services through to national services through to international services.
ServiceLevel: a description of the level at which this service operates. Currently this defines divisions, organisations, national, and international.

Owner: points to information on the owner of the system and their contact details.
Tables: points to information on all tables of codes (apart from classifications) which are used by the system. The full set of codes will be stored in the PropertySet definitions.
Classifications: points to information on all classification systems used by the service.
UpTime: points to the availability statistics for this particular service.
HelpDeskURL: contains the starting point for the helpdesk associated with this service. This allows both bespoke helpdesks or centralised or shared helpdesks to be linked in.
ElecEntryURL: contains the starting point for an electronic entry system for the particular service. This allows both bespoke electronic entry services or centralised or shared electronic entry services to be linked in.
Certificate: contains the certificate information for the service used to establish a secure connection with the service for the transfer of sensitive information.
Language: main language of the service.
ServiceID: this is the unique ID allocated to the service by the CONNET central system.
Users: Set of users.
AssociationStatus: defines the status this service has with the CONNET central system in terms of how integrated it is with the CONNET methodologies.
ChargingStrategy: defines the charging strategy employed by the service in order for CONNET to determine whether a particular user could use the service.
SimpleQueryURL: Each service has a simple search box. This the URL of this query in 'get' format.
QueryReloadURL: URL of query for 'post' format.
System : The national gateway this service falls under.

ENTITY: Document

Captures basic information about a document which enables CONNET and other systems to determine how to handle particular documents, or to identify documents which offer a particular type of information.

EXPRESS specification:
```
ENTITY Document;
      Locator : STRING;
      MIMEType : OPTIONAL STRING;
      ContentType : OPTIONAL STRING;
END_ENTITY;
```

Attribute definitions:
Locator: describes where the document is located. In most cases this will be a URL, though it could give a physical location as well.
MIMEType: for electronic documents this specifies the format of the document in order for applications to determine how to display the document.
ContentType: provides for the specification of simple content descriptions to be matched with the MIMEType to enable documents suiting a particular purpose to be identified (e.g., thumbnail pictures, or detail drawings, etc).

ENTITY: DublinCore

Describes the meta data which can be associated with electronic and published resources. This is being taken forward as a standard through W3C and the XML/RDF initiatives.

EXPRESS specification:
```
ENTITY DublinCore;
      Language : LANGUAGE;
      Date : DATE;
      Title : STRING;
      Author : STRING;
      SubjectKeyword : STRING;
      Description : STRING;
      Publisher : STRING;
      OtherContributor : STRING;
      ResourceType : StandardResourceTypes;
      Format : STRING;
      ResourceIdentifier : STRING;
      Source : SET [0:?] OF DublinCore;
      Coverage : STRING;
      RightsManagement : STRING;
      Relation : SET [0:?] OF DublinCore;
END_ENTITY;
```

Attribute definitions:

Language: defines the language in which the resource is written.

Date: specifies the date of publication of the resource.

Title: the title of the resource as specified by creator or publisher.

Author: describes the person(s) responsible for the intellectual content of the specified resource.

SubjectKeyword: the subject descriptor or a set of keywords that describe the contents of the resource.

Description: a textual description of the contents of the resource, which could include abstract in the case of a document.

Publisher: provides information about the person or organisation which publishes the resource.

OtherContributor: provides information about any other person or organisation involved in the production or distribution of the resource.

ResourceType: defines the resource type which is being described drawn from a standard set of predefined types.

Format: this is the format in which the resource is published, which would be a MIME type for an electronic resource.

ResourceIdentifier: an identifier for the resource which is being described in the associated meta data.

Source: this describes other resources which have the same intellectual content as this resource (e.g., from which this resource is drawn).

Coverage: defines the spatial and temporal coverage of the resource.

RightsManagement: provides a link to copyright information about the resource.

Relation: points to other resources which are associated with this resource (e.g., a chapter of a book).

ENTITY: IntegerProperty

Subclass of property, used when related PropertyDefiniton has datatype 'integer'. Attribute Min holds the actual value, or min. bound of range when also the attribute Max is populated.

EXPRESS specification:
```
ENTITY IntegerProperty
      SUBTYPE OF (Property);
      Min : INTEGER;
      Max : OPTIONAL INTEGER;
END_ENTITY;
```

Attribute definitions:
Min: the actual value of the property, or the minimum bound if Max is different.
Max: the actual value of the property, or the maximum bound if Min is different.

ENTITY: ItemOrder

Contains information on the item that has been ordered information about the order.

EXPRESS specification:
```
ENTITY ItemOrder;
      ItemOrdered : OrderableItem;
      Selected : DATETIME;
      Status : ItemStatus;
      NumberOrdered : INTEGER;
      AgreedCost : NUMBER;
      Currency : CURRENCY;
 INVERSE
      ProviderOrder : ProviderOrder FOR Items;
END_ENTITY;
```

Attribute definitions:
ItemOrdered: This is the item itself.
Selected: Date and time selected.
Status: current status of the order.
NumberOrdered: quantity ordered.
AgreedCost: price.
Currency: currency.
ProviderOrder: The larger order for which this is an item of.

ENTITY: KnownUser

Captures information about users of the CONNET system and its services. This information is captured fully for a known user, but partially filled for a user who isn't fully identified to the system.

EXPRESS specification:
```
ENTITY KnownUser
      SUPERTYPE OF (CommerceUser);
      Identity : Organisation;
      UserID : INTEGER;
      Profession : STRING;
      Joined : DATE;
      CertificateCookie : STRING;
      ServicesUsed : SET [1:?] OF ServiceInformation;
      Tracking : SET [0:?] OF Trace;
      Language : LANGUAGE;
      DeliveryFailures : INTEGER;
      Usercode : STRING;
      Password : STRING;
      MemberOf : SET [0:?] OF MembershipDetails;
 UNIQUE
      UR1:  UserID;
END_ENTITY;
```

Attribute definitions:

Identity: points to basic information about the user and their organisation to enable further contact with the user by any means.
UserID: the unique ID of the user over all CONNET services.
Profession: a specification of the user's profession, currently against basic titles such as architect, civil engineer, structural engineer, facility manager, etc.
Joined: date at which they first started using the system, or the date at which they fully registered with the system.
CertificateCookie: information on the security system used to validate this user when they come into the CONNET system or any of its services. This can include certificates, cookies, or usercode/password combinations.
ServicesUsed: identifies the set of CONNET related services that this user has previously worked with.
Tracking: points to collated information on what the user has done in the system over the total period that they have used it.
Language: language.
DeliveryFailures: number of past failed deliveries.
Usercode: username.
Password: password.
MemberOf: user may be use a corporate accounts or there may be special deals for members of particular ogganisations.

ENTITY: MembershipDetails

Details for corporate accounts or organisations which have special deals, discounts, etc.

EXPRESS specification:

```
ENTITY MembershipDetails;
      MembershipID : STRING;
      Usercode : STRING;
      Password : STRING;
      AccountNumber : STRING;
      Expiry : DATE;
      ProfessionalOrganisation : Organisation;
END_ENTITY;
```

Attribute definitions:

MembershipID: ID to identify organisation.
Usercode: corporate or group username
Password: corporate or group password.
AccountNumber: account number
Expiry: expiry date.
ProfessionalOrganisation: The organisation or company.

ENTITY: ObjectClass

Defines the accessible properties of any class which may be used by a CONNET service. In the first instance this defines the properties of products found through the manufactured product service in CONNET.

EXPRESS specification:
```
ENTITY ObjectClass;
      Properties : SET [1:?] OF PropertyDefinition;
      ExternalClassName : OPTIONAL STRING;
      ObjectClassName : STRING;
      Service : CONNETService;
 INVERSE
      Classifications : SET[1:?] OF ClassificationCode FOR
SearchableClasses;
 UNIQUE
      UR1:  ObjectClassName;
END_ENTITY;
```

Attribute definitions:

Properties: points to all the searchable properties of the CONNET class (product category).

ExternalClassName: optional name of class in external specification (e.g. IFC) corresponding to this CONNET class (product category).

ObjectClassName: name of the class (product category) in CONNET system.

Classifications: all classifications which may be applicable to this type class of product.

ENTITY: OrderableItem

A more intuitive alias for Dublin Core where all data is stored.

EXPRESS specification:
```
ENTITY OrderableItem
      ABSTRACT SUPERTYPE OF (DublinCore);
END_ENTITY;
```

ENTITY: Organisation

Defines the basic details of an organisation for use throughout the CONNET system and all its services.

EXPRESS specification:
```
ENTITY Organisation
      SUPERTYPE OF (ProviderOrganisation);
      Documents : SET [0:?] OF Document;
      Country : OPTIONAL COUNTRY;
      Name : STRING;
      Address : OPTIONAL STRING;
      Telephone : OPTIONAL STRING;
      Fax : OPTIONAL STRING;
      WebPage : OPTIONAL STRING;
      Email : OPTIONAL STRING;
      ContactName : OPTIONAL STRING;
      PostCode : OPTIONAL STRING;
      OrganisationType : OPTIONAL STRING;
      AssociatedOrganisations : SET [0:?] OF Organisation;
      Language : OPTIONAL LANGUAGE;
      Acronym : OPTIONAL STRING;
END_ENTITY;
```

Attribute definitions:

Documents: points to all documents associated with this organisation. This could include the logo, thumbnail of logo, brochure, ISO 9001 certificate, etc.
Country: specifies the country this organisation is based in.
Name: the trading name of the organisation.
Address: the mailing address of the organisation.
Telephone: main phone number for the organisation or service leader.
Fax: main fax number of the organisation or service leader.
WebPage: the URL of the organisation's web page.
Email: the email contact for the organisation or service leader.
ContactName: initial contact point for the organisation or service.
PostCode: mailing post code for the organisation.
OrganisationType: used to categorise the organisation. Currently captures information on university, membership based, research, standards, manufacturer, supplier, etc.
AssociatedOrganisations: provides a link to any associated organisations. Initially used in CONNET to define he suppliers for a particular manufacturer.
Language: main language.
Acronym: organisation acronym.

ENTITY: Profile

Captures individual user's profiles in each service for areas of interest and topics they wish to be notified about.

EXPRESS specification:

```
ENTITY Profile;
      NotifyFrequency : TimePeriod;
      NotifyQuery : PropertySet;
      Created : DATE;
      LastServiced : DATE;
      Delivery : DeliveryMode;
      Name : STRING;
      Query : STRING;
END_ENTITY;
```

Attribute definitions:

NotifyFrequency: specifies how often the user wishes to be notified about new items from this profile.
NotifyQuery: points to the query which matches the profile established by the user.
Created: date the profile was established.
LastServiced: date the profile was last checked and information sent back to the user.
Delivery: specifies the delivery mechanism to use to present information gathered against the profile back to the user.
Name: user specified name for this profile.

ENTITY: Property

Abstract class wrapping reference to definition of dynamic properties and their value (through appropriate subclasses).

EXPRESS specification:
```
ENTITY Property
      ABSTRACT SUPERTYPE OF
(ONEOF(StringProperty,RealProperty,IntegerProperty));
      Definition : PropertyDefinition;
END_ENTITY;
```

Attribute definitions:
Definition: reference to PropertyDefinition declaring the property name, datatype and unit (and optionally user group to view) for this dynamically assigned property.

ENTITY: PropertyDefinition

Definition of single property, which is one of the searchable properties related to a CONNET class or to extended properties of a product. This is mostly data dictionary information, though it also provides view information and units.

EXPRESS specification:
```
ENTITY PropertyDefinition;
      UserGroups : OPTIONAL STRING;
      Unit : OPTIONAL STRING;
      Static : BOOLEAN;
      DataType : STRING;
      PropertyName : STRING;
 INVERSE
      ObjectClass : ObjectClass FOR Properties;
END_ENTITY;
```

Attribute definitions:
UserGroups: optional identification of the group(s) of users to which the property is relevant and thus will be exposed.
Unit: unit of measure for the property, or the list of enumeration values.
Static: defines whether the property is part of the class definition or if it is defined dynamically through the Property class.
DataType: definition of the data-type of the property: string, real, integer, or enumeration.
PropertyName: name of the property.

ENTITY: PropertySet

Class capturing the properties of a product, containing both searchable properties as specified in Connet ObjectClass, and eventual extension properties as specified by manufacturer.

EXPRESS specification:
```
ENTITY PropertySet;
      PropertySetName : OPTIONAL STRING;
      Properties : SET [1:?] OF Property;
END_ENTITY;
```

Attribute definitions:
PropertySetName: optional name for the set of properties related to a product.

Properties: points to all properties associated with the named property set.

ENTITY: ProviderOrder

A user order may consist of items from many suppliers or providers. The ProviderOrder is the part of the overall order that goes to a single provider.

EXPRESS specification:
```
ENTITY ProviderOrder;
      Status : OrderStatus;
      Submitted : DATETIME;
      Completed : DATETIME;
      DelayedUntil : DATETIME;
      TotalCost : NUMBER;
      TotalProfit : NUMBER;
      TotalVAT : NUMBER;
      PPCost : NUMBER;
      OrderForProvider : ProviderOrganisation;
      Items : SET [0:?] OF ItemOrder;
      MemberPurchase : BOOLEAN;
      ResonDelayed : STRING;
      ProviderOrderID : STRING;
      Currency : CURRENCY;
 INVERSE
      Order : UserOrder FOR ProviderOrders;
END_ENTITY;
```

Attribute definitions:
Status : status of order;
Submitted : date and time submitted;
Completed : date and time completed;
DelayedUntil : date and time for any known delay;
TotalCost : cost;
TotalProfit : profit;
TotalVAT : VAT;
PPCost : charges for post and packing;
OrderForProvider : The provider organisation;
Items : The items within the order;
MemberPurchase : Is the buyer a member of any organisation for which special deals have been agreed. (This is an indication to the provider that the total cost will probably not reflect list price.);
ResonDelayed : reason for any delay;
ProviderOrderID : ID;
Currency : currency;
Order : The UserOrder that this order is a part of.

ENTITY: ProviderOrganisation

Subclass of organisation, captures information about organisations which periodically provide sets of information to the CONNET system.

EXPRESS specification:
```
ENTITY ProviderOrganisation
      SUBTYPE OF (Organisation);
      UpdateFrequency : TimePeriod;
      LastUpdate : DATE;
      UpdateMode : DeliveryMode;
      JoinedDate : DATE;
      Password : STRING;
      Usercode : STRING;
END_ENTITY;
```

Attribute definitions:

UpdateFrequency: the frequency that this organisation has agreed to supply information to the particular service it is working for.

LastUpdate: the last date at which a successful update was made of the information provided by this organisation.

UpdateMode: the method through which an update is made. This can include Email, HTTP, database, disk, etc.

ENTITY: RealProperty

Subclass of property, used when related PropertyDefiniton has data-type 'real'. Attribute Min holds the actual value, or minimum bound of range when the attribute Max is also populated.

EXPRESS specification:

```
ENTITY RealProperty
      SUBTYPE OF (Property);
      Min : REAL;
      Max : OPTIONAL REAL;
END_ENTITY;
```

Attribute definitions:

Min: the actual value of the property, or the minimum bound if Max is different.

Max: the actual value of the property, or the maximum bound if Min is different.

ENTITY: ServiceInformation

Captures user profile information against an individual service. This is defined on a service by service basis depending upon the needs of the service.

EXPRESS specification:

```
ENTITY ServiceInformation;
      ServiceProperties : PropertySet;
      Service : CONNETService;
      NotifyProfile : SET [0:?] OF Profile;
 INVERSE
      User : KnownUser FOR ServicesUsed;
 UNIQUE
      UR1:  Service;
      UR2:  User;
END_ENTITY;
```

Attribute definitions:

Service: points to the basic information on the service for which this profile is being captured.

NotifyProfile: defines the set of profiles used by the associated user for this service.

ServiceProperties: points to dynamically defined properties used to capture information required for the service.

User: points to the user for whom this service based information is being collected.

ENTITY: StringProperty

Subclass of property, used when related PropertyDefiniton has data-type 'string' 'list' or 'enum'.

EXPRESS specification:
```
ENTITY StringProperty
      SUBTYPE OF (Property);
      Content : STRING;
END_ENTITY;
```

Attribute definitions:
Content: the value of the string, list, or enumerated property.

ENTITY: TableOfCodes

For definition of many things amongst them emumerated types via property sets.

EXPRESS specification:
```
ENTITY TableOfCodes;
      TableName : STRING;
      IndexNumber : INTEGER;
      Rows : LIST [1:?] OF PropertySet;
      Language : LANGUAGE;
 INVERSE
      Services : SET[1:?] OF CONNETService FOR Tables;
END_ENTITY;
```

Attribute definitions:
TableName :name of enumeration;
Rows : pointer to list of enumerations;
Language : language;
Services : services used in;

ENTITY: Trace

Captures information about a known user’s every activity inside the CONNET system and all of its services.

EXPRESS specification:
```
ENTITY Trace;
      Action : STRING;
      ActionType : Action;
      Time : DATETIME;
      Service : CONNETService;
 INVERSE
      User : KnownUser FOR Tracking;
END_ENTITY;
```

Attribute definitions:
Service: points to the service which was being used by the user at the time they made some action.
Action: the action sting from the user. In most cases this would be a query string, though it could also be summarised results from a query.

ActionType: defines the type of action the user made in the service or CONNET.
Time: the time at which the action was initiated.

ENTITY: UserOrder

A user order may consist of items from many suppliers or providers.

EXPRESS specification:
```
ENTITY UserOrder;
      Submitted : DATETIME;
      Completed : DATETIME;
      Authorised : DATETIME;
      AuthorisationCode : STRING;
      UserOrderNumber : STRING;
      CardHolderName : STRING;
      CardLast4Digits : STRING;
      BankCardWarningsList : STRING;
      TotalCost : NUMBER;
      TotalProfit : NUMBER;
      TotalVAT : NUMBER;
      BankAuthorisedPayment : NUMBER;
      CardType : CreditCards;
      ProviderOrders : SET [0:?] OF ProviderOrder;
      Status : OrderStatus;
      Country : COUNTRY;
      DeliveryDetails : Organisation;
      BillingDetails : Organisation;
      ForUser : CommerceUser;
      Currency : CURRENCY;
END_ENTITY;
```

Attribute definitions:
Submitted : date and time;
Completed : date and time;
Authorised : date and time;
AuthorisationCode : any authorisation code required for credit card;
UserOrderNumber : order number;
CardHolderName : name on card;
CardLast4Digits : last 4 digits;
BankCardWarningsList : any info on card warnings list;
TotalCost : cost;
TotalProfit : profit;
TotalVAT : VAT;
BankAuthorisedPayment : code indicating whether payment has been authorised;
CardType : Type of credit card;
ProviderOrders : set of suppliers that parts of order will be sent to;
Status : Order Status;
Country : country ordered from;
DeliveryDetails : who and where to deliver to;
BillingDetails : who and where to bill;
ForUser : link to user details;
Currency : currency;

APPENDIX B: Schema CONNET_BPC

The CONNET best practice centre (BPC) schema provides the core information required by a best practice centre to manage its information as well as connections through to CONNET for its services. The scope of a best practice centre is to raise awareness of the benefits of best practice and provide guidance and advice to construction and client organisations so that they have the knowledge and skills required to implement change.

EXPRESS-G Diagram

Figure 7 Schema CONNET_BPC

TYPE: InformationSectorInitiativeType

Defines the types of sector initiatives.

EXPRESS specification:
```
TYPE InformationSectorInitiativeType = ENUMERATION OF
      ();
END_TYPE;
```

TYPE: InformationSubType

Defines the sub types information available in the best practice centre. The SubType is used to locate the information you want and to separate information into blocks so that it is easier to browse through.

EXPRESS specification:
```
TYPE InformationSubType = ENUMERATION OF
      ();
END_TYPE;
```

TYPE: InformationType

Defines the types information available in the best practice centre. The Type is used to denote what information a customer requires – a document requires ordering info, an activity requires booking/venue info and advisors require contact info

EXPRESS specification:
```
TYPE InformationType = ENUMERATION OF
      (Document,
       Activity,
       Advisor/Scheme);
END_TYPE;
```

TYPE: ThemeType

Defines the types of theme (classification) available in the best practice centre.

EXPRESS specification:
```
TYPE ThemeType = ENUMERATION OF
      ();
END_TYPE;
```

ENTITY: BestPracticeCentre

This is the top level descriptor of a best practice centre with pointers to different types of best practice information. This is a specialisation of CONNETService.

EXPRESS specification:
```
ENTITY BestPracticeCentre
      SUBTYPE OF (CONNETService);
      BestPracticeInformation : SET [1:?] OF InformationItem;
 UNIQUE
      UR1:  ServiceID;
END_ENTITY;
```

Attribute definitions:

ServiceID: the CONNET specified unique ID for this service. This is used in all conversations between this service and the CONNET system.

BestPracticeInformation: points to the set of all best practice information contained in the best practice centre.

ENTITY: InformationItem

Captures information about a piece of best practice information.

EXPRESS specification:
```
ENTITY InformationItem
      SUPERTYPE   OF   (BestPracticeDocument   ANDOR   ActivityInfo   ANDOR
AdvisorSchemeInfo);
      Type : InformationType;
      SubType : InformationSubType;
      MediaDescription : Media;
      Level : INTEGER;
      Entered : DATE;
      Modified : DATE;
      SectorInitiative : InformationSectorInitiativeType;
      Theme : ThemeType;
END_ENTITY;
```

Attribute definitions:

Type: classification of the information based on the set of information types.
SubType: classification of the information based on the set of information sub-types.
MediaDescription: defines how this information is made available e.g. book, video, web site, seminar or workshop.
Level: the level of domain knowledge that a user would need to have in order to make use of this information.
Entered: the date that the information was entered into the system.
Modified: the date that the information was last updated.
SectorInitiative: classification of the information based on sector initiatives.
Theme: classification of the information based on the set of themes.

ENTITY: BestPracticeDocument

Captures information about a document about best practice. It is a specialisation of InformationItem.

EXPRESS specification:
```
ENTITY BestPracticeDocument
      SUBTYPE OF (InformationItem);
      Formats : SET [0:?] OF Document;
      Price : NUMBER;
      Currency : Currency;
      InformationProvider : Organisation;
      DocumentInfo : DublinCore;
END_ENTITY;
```

Attribute definitions:
Formats: points to an electronic copy of the actual document, possibly available in more than one format e.g. PDF or HTML.
Price: the full price of the document.
Currency: the currency that the price was given in.
InformationProvider: the supplier of the information.
DocumentInfo: the dublin core description of the document.

ENTITY: ActivityInfo

Captures information about an activity. This is a specialisation of InformationItem.

EXPRESS specification:
```
ENTITY ActivityInfo
      SUBTYPE OF (InformationItem);
      Organiser : Organisation;
      Venue : Organisation;
      Host : Organisation;
      Price : NUMBER;
      Period : ActivityDate;
END_ENTITY;
```

Attribute definitions:
Organiser: the organiser of the activity.
Venue: the location where the activity will take place.
Host: the host for the activity.
Price: the cost of participating in the activity.
Period: the time period during which the activity is available.

ENTITY: AdvisorSchemeInfo

Captures information about an advisor or scheme. This is a specialisation of InformationItem.

EXPRESS specification:
```
ENTITY AdvisorSchemeInfo
      SUBTYPE OF (InformationItem);
      Availability : ActivityDate;
      Utilises : Company;
END_ENTITY;
```

Attribute definitions:
Availability: the time period during which this advisor/scheme is available.
Utilises: points to the company that is providing the expertise.

ENTITY: Company

Captures information about a company. This is a specialisation of Organisation.

EXPRESS specification:
```
ENTITY Company
      SUBTYPE OF (Organisation);
      Size : INTEGER;
      MarketRank : INTEGER;
      Source : STRING;
END_ENTITY;
```

Attribute definitions:
Size: the size of the company – number of employees.
MarketRank: indicates the companies position within their market.
Source: where the information for this record came from.

ENTITY: ActivityDate

Captures information about a company. This is a specialisation of Organisation.

EXPRESS specification:
```
ENTITY ActivityDate;
      Start : DATE;
      End : DATE;
      FullyBooked : BOOLEAN;
END_ENTITY;
```

Attribute definitions:
Start: the date that the activity starts, this is either the first day of a longer activity or the first of many dates that the activity is available.
End: the date that the activity ends or the last date that the activity is available.
FullyBooked: defines if the activity is fully booked.

APPENDIX C: Schema CONNET_CSC

The CONNET calculation and software centre (CSC) schema provides the core information required by a calculation and software centre to manage its information as well as connections through to CONNET for its services. The scope of a calculation and software centre is to be a gateway to all software available in a particular discipline (civil engineering for the first demonstrator in CONNET). This is international by default, though could be set up for a membership based organisation.

EXPRESS-G Diagram

Figure 8 Schema CONNET_CSC

TYPE: LicenseType

Defines the major categories of license applicable to software.

EXPRESS specification:
```
TYPE LicenseType = ENUMERATION OF   (PublicDomain,     Freeware,
Shareware,   Commercial);END_TYPE;
```

TYPE: SoftwareType

Defines the major categories of software being dealt with in this system (not shown for brevity) drawn from ISO/IEC TR 12182:1998.

EXPRESS specification:
```
TYPE SoftwareType = ENUMERATION OF  ();END_TYPE;
```

ENTITY: CalculationSoftwareCentre

This is the top level descriptor of a calculation and software centre with pointers to software providers. This is a specialisation of CONNETService.

EXPRESS specification:
```
ENTITY CalculationSoftwareCentre
      SUBTYPE OF (CONNETService);
      Vendors : SET [1:?] OF Vendor;
      DomainServed : STRING;
 UNIQUE
      UR1:  ServiceID;
END_ENTITY;
```

Attribute definitions:

ServiceID: the CONNET specified unique ID for this service. This is used in all conversations between this service and the CONNET system.

Vendors: points to the set of vendors who have provided software information into this calculation and software centre.

DomainServed: defines the domain which this software and calculation centre collates information for. This is likely to be architecture, civil engineering, etc.

ENTITY: Review

Captures information about a single review of a single piece of software available in this calculation and software centre.

EXPRESS specification:
```
ENTITY Review;
      At : DATETIME;
      Rating : INTEGER;
      From : STRING;
      Reviewer : Organisation;
      Review : STRING;
      Resource : STRING;
END_ENTITY;
```

Attribute definitions:
At: time of last update of record.
Rating: rating given to the resource ranging from 0-5.
From: host from which the update was made.
Reviewer: full contact details including name and email address of the reviewer.
Review: text of the review.
Resource: ID of the resource reviewed.

ENTITY: Software

Defines all information about a particular piece of software supplied by a particular vendor in this calculation and software centre.

EXPRESS specification:
```
ENTITY Software   SUBTYPE OF (DublinCore);      SoftwareVersion : STRING;
      ProductCoverage : STRING;     Reviews : SET [0:?] OF Review;
      ProcessCoverage : STRING;     Platforms : STRING;     Features :
STRING;     Availability : STRING;
      SoftwareType : SoftwareType;
      SoftwareLicense : LicenseType;
      At : DATETIME;
      Access : STRING;
      From : STRING;
      EnteredBy : ProviderOrganisation;
END_ENTITY;
```

Attribute definitions:
SoftwareVersion: specifies the version of the software recorded in the system.
ProductCoverage: keywords related to building product type, it is envisaged these will form the start of a controlled set drawn from a central classification scheme.
Reviews: points to a set of reviews about this particular piece of software.
ProcessCoverage: keywords related to building product, it is envisaged these will form the start of a controlled set.
Platforms: controlled set of platforms and operating systems that the software will operate on.
Features: where a demonstrator exists this defines how it is different from the actual software, initial categories are: Full; Crippled; TimeLimited.
Availability: defines how demonstrator software can be accessed, initial categories are: Download; Email; Mail.
SoftwareType: defines the type of software recorded in the system with keywords drawn from a controlled subset of ISO/IEC TR 12182:1998.
SoftwareLicense: defines the license type of the software, it is one of PublicDomain, Freeware, Shareware, Commercial.
At: time of last update of record.
Access: defines the cost category for access to the software, initial categories are: Free, $, $$, $$$.
From: host from which the update was made.
EnteredBy: full contact details including name and email of person that did the update.

ENTITY: Vendor

Provides information about the manufacturer/vendor of software accessible through this calculation and software centre.

EXPRESS specification:
```
ENTITY Vendor;
      SUBTYPE OF (Organisation);    Software : SET [1:?] OF Software;
      At : DATETIME;
      From : STRING;
      EnteredBy : ProviderOrganisation;
END_ENTITY;
```

Attribute definitions:

Software: points to all software available from this vendor.
At: time of last update of record
From: host from which the update was made
EnteredBy: full contact details including name and email of person that did the update.

APPENDIX D: Schema CONNET_ESC

The CONNET equipment supply centre (ESC) schema provides the core information required by an equipment supply centre to manage its information as well as connections through to CONNET for its services. The scope of an equipment supply centre is to be a directory of all specialist equipment and facilities that are available, either to buy or to lease, within a country.

EXPRESS-G Diagram

Figure 9 Schema CONNET_ESC

TYPE: BusinessTerms

Defines the business terms of an equipment supplier.

EXPRESS specification:
```
TYPE BusinessTerms = ENUMERATION OF
      (ForSale,
       ForLease,
       ForRentMonthlyRate,
       ForRentWeeklyRate,
       ForRentDailyRate,
       ForRentHourlyRate,
       PayPerUse);
END_TYPE;
```

ENTITY: EquipmentSupplyCentre

This is the top level descriptor of an equipment supply centre with pointers to equipment and facilities. This is a specialisation of CONNETCommercialService.

EXPRESS specification:

```
ENTITY EquipmentSupplyCentre
      SUBTYPE OF (CONNETCommercialService);
      EquipmentAndFacilities : SET [1:?] OF Item;
 UNIQUE
      UR1:  ServiceID;
END_ENTITY;
```

Attribute definitions:

ServiceID: the CONNET specified unique ID for this service. This is used in all conversations between this service and the CONNET system.

EquipmentAndFacilities: points to the set of equipment and facilities contained in the ESCs directory.

ENTITY: Equipment

Captures information about a piece of equipment available in this equipment supply centre. This is a specialisation of Item.

EXPRESS specification:

```
ENTITY Equipment
      SUBTYPE OF (Item);
      Suppliers : SET [1:?] OF Organisation;
      TradeName : STRING;
      EquipmentCode : STRING;
END_ENTITY;
```

Attribute definitions:

Suppliers: points to the set of organisations that can supply this piece of equipment.
TradeName: the trade name for this piece of equipment.
EquipmentCode: a unique ID for this piece of equipment.

ENTITY: Facility

Captures information about a facility available in this equipment supply centre. This is a specialisation of Item.

EXPRESS specification:

```
ENTITY Facility
      SUBTYPE OF (Item);
      Location : Organisation;
END_ENTITY;
```

Attribute definitions:

Location: The location of the facility.

ENTITY: Item

Defines all information about a particular item, either a piece of equipment or a facility, within this equipment supply centre. It is a specialisation of OrderableItem.

EXPRESS specification:

```
ENTITY Item
```

```
      SUPERTYPE OF (Equipment ANDOR Facility)
      SUBTYPE OF (OrderableItem);
      ItemSourceOwnerManufacturer : Organisation;
      Identifier : STRING;
      Description : OPTIONAL STRING;
      Synonyms : SET [0:?] OF STRING;
      UseTerms : BusinessTerms;
      Category : ObjectClass;
      Properties : OPTIONAL PropertySet;
      Documents : SET [0:?] OF Document;
      ModificationTime : DateTime;
      Classifications : SET [1:?] OF ClassificationCode;
      ExpirationTime : DateTime;
      InfoAuthor : ProviderOrganisation;
 UNIQUE
      UR1:  Identifier;
END_ENTITY;
```

Attribute definitions:

ItemSourceOwnerManufacturer: pointer to the organisation that is the source of this item. In the case of a piece of equipment this will be a manufacturer and for a facility this will be the owner.
Identifier: a unique identifier for this item.
Description: a brief description of the item.
Synonyms: points to the set of names, e.g. different trade names, that this item may be known as.
UseTerms: the business terms of this item i.e. whether it is available to buy or lease.
Category: points to the category that this item belongs to.
Properties: points to a set of properties that describe the item.
Documents: a set of documents that give further information about the item.
ModificationTime: the time that the information about this item was last updated.
Classifications: points to the set of classification codes, from one or more classification systems, that describe the item.
ExpirationTime: defines how long the information about this item remains valid.
InfoAuthor: points to the details of the author of the information about this item.

APPENDIX E Schema CONNET_NS

The CONNET news service (NS) schema provides the core information required by a news service to manage its information as well as connections through to CONNET for its services. The scope of a news service is to be a gateway to Internet based news from a range of controlled and selectable sources. This is international by default (as in the first demonstrator for CONNET) or could be set up within an organisation, or for a membership based organisation.

EXPRESS-G Diagram

Figure 10 Schema CONNET_NS

TYPE: SiteStatusType

Defines the status of a site in the index.

EXPRESS specification:

```
TYPE SiteStatusType = ENUMERATION OF
     (Available,
      TemporarilyUnavailable,
      PermanentlyUnavailable,
      NotSuitable);
END_TYPE;
```

TYPE: QualityType

Defines the estimate of the quality level of information at a particular site. This is a gross measure as the majority of sites are very variable.

EXPRESS specification:

```
TYPE QualityType = ENUMERATION OF
       (Unknown,
        IndustryCommitteeInformation,
        OrganisationAuthorisedInformation,
        IndividualResearch,
        Informative,
        Advertising);
END_TYPE;
```

ENTITY: NewsProfile

Specialises the core Profile class definition to provide further information about how the news should be gathered and also how it should be displayed.

EXPRESS specification:

```
ENTITY NewsProfile;
      SUBTYPE OF (Profile);
      UseThesaurus : BOOLEAN;
      NumberOfMatchingLines : INTEGER;
      NumberOfMatches : INTEGER;
      ShowTitle : BOOLEAN;
      ShowHeadings : BOOLEAN;
      FromQuality : INTEGER;
      FromURL : STRING;
      FromSource : STRING;
      FromCountry : COUNTRY;
END_ENTITY;
```

Attribute definitions:

UseThesaurus: defines whether a thesaurus should be used with the query to expand the words/terms being searched for.
NumberOfMatchingLines: defines how many descriptive lines should be displayed for each matching page from a site.
NumberOfMatches: defines how many matching pages should be displayed from a site, this would start at around 20 for a reasonable number of results.
ShowTitle: defines whether the title of a page is shown as part of the results.
ShowHeadings: defines whether the H1, H2, and H3 text in a web page is shown as part of the results.
FromQuality: limits the quality of sites which should be considered.
FromURL: limits the starting point of sites to be considered.

FromSource: limits the set of sites to be considered (e.g., only those gathered from EEVL).
FromCountry: limits the source country of sites to be considered.

ENTITY: NewsService

This is the top level descriptor of a news service with pointers to CONNET, its information sites. It is a specialisation of CONNETService.

EXPRESS specification:
```
ENTITY NewsService
      SUBTYPE OF (CONNETService);
      URLs : SET [1:?] OF NewsURL;
 UNIQUE
      UR1:  ServiceID;
END_ENTITY;
```

Attribute definitions:
URLs: points to the set of sites which are crawled to extract news for the service.
ServiceID: the CONNET specified unique ID for this service. This is used in all conversations between this service and the CONNET system.

ENTITY: NewsURL

Defines what meta-data is captured for each news site which is managed by the system. This comprises keywords and measures of the quality of the information which will be accessed.

EXPRESS specification:
```
ENTITY NewsURL;
      Quality : QualityType;
      GathererID : INTEGER;
      URL : STRING;
      Title : STRING;
      Keywords : STRING;
      Source : STRING;
      Country : COUNTRY;
      UpdateFrequency : TimePeriod;
      LastUpdate : DATE;
      Contact : Organisation;
      Description : STRING;
      RedirectedURL : STRING;
      ErrorCount : INTEGER;
      DefunctSite : ProblemType;
      PageGathered : BOOLEAN;
      Language : LANGUAGE;
      Email : STRING;
      PageText : STRING;
 UNIQUE
      UR1:  URL;
END_ENTITY;
```

Attribute definitions:
Quality: defines the predicted quality of information which will be obtained from this site.
ErrorCount: keeps track of the number of times the site has caused problems. Is an indicator that a site has closed.

URL: the top level location of a site. This represents the entry point for a particular organisation or type of service/information.
Title: the descriptive title used by the site or applied by an administrator adding a new site.
Keywords: a set of keywords defining the basic content of the site. This is uncontrolled.
Description: the description of the site as found in the sites meta data. This is uncontrolled.
Source: a set of identifiers which tracks where the pointer to this site was found.
Country: the country being served by the site.
Contact: points to information on who can be contacted regarding the site specified. This will often just be the email of the web-master as taken from the web site.
UpdateFrequency: the frequency which this web site should be crawled to ensure that the central news service remains up-to-date.
LastUpdate: the last time at which the site was successfully crawled to extract information on all pages currently available from the site.
SiteStatus: defines the status of the site at the time it was last gathered and indicated whether a site should be indexed in the future.
PageGathered: indicates that a site was successfully indexed.
Language: the language used on the site.
PageText: the information gathered from the site.

ENTITY: SourceURL

Defines the list of sources that are associated with the NewsURLs.

EXPRESS specification:
```
ENTITY SourceURL;
      SourceCode : STRING;
      SourceTitle : STRING;
      SourceURL : STRING;
      LastGathered : DATE;
      Country : COUNTRY;
 INVERSE
      GatherPages : SET[1:?] OF GathererURL FOR Source;
END_ENTITY;
```

Attribute definitions:
SourceCode: a unique code to represent this source
SourceTitle: the name of the source.
SourceURL: the URL of the source (if it is an internet resource).
Country: the country being served by the site.
LastGathered: the last time at which the site was successfully crawled to extract information on all pages currently listed.

ENTITY: GathererURL

This is an expansion of SourceURLs that gives a list of sites to crawl in search of new sites to add to the set of NewsURLs. Each of the URLs in this list point to lists of sites that may be of interest.

EXPRESS specification:
```
ENTITY GathererURL;
```

```
      Source : SourceURL;
      GatheringURL : STRING;
      Keywords : STRING;
END_ENTITY;
```

Attribute definitions:
Source: pointer to the SourceURL for this list.
GatheringURL: the URL of the pages to crawl.
Keywords: a list of keywords associated with this site.

APPENDIX F: Schema CONNET_TIC

The CONNET technical information centre (TIC) schema provides the core information required by a technical information centre to manage its information as well as connections through to CONNET for its services. The scope of a technical information centre is to be a gateway to the published technical information of a range of sources. This could be set up nationally (as in the first demonstrator for CONNET) or within an organisation, or for a membership based organisation, or even internationally.

EXPRESS-G Diagram

Figure 11 Schema CONNET_TIC

TYPE: OrderMode

Defines possible ways that an order can be passed to an information provider.

EXPRESS specification:
```
TYPE OrderMode = ENUMERATION OF
     (Email,
      FAX,
      Invoice,
      EDI);
END_TYPE;
```

ENTITY: InformationProvider

Describes a provider of technical information to the technical information centre. In the main these will be publishers within a country. This is a specialisation of ProviderOrganisation.

EXPRESS specification:
```
ENTITY InformationProvider
      SUBTYPE OF (ProviderOrganisation);
      Information : SET [1:?] OF TICResource;
      RawInfoSource : STRING;
      ReloadCall : STRING;
      PurchaseResolver : Organisation;
      PostPackageCalculation : STRING;
      PassOrderBy : OrderMode;
      SLAResponseTime : NUMBER;
      DiscountRate : NUMBER;
      MemberPrices : BOOLEAN;
      MinimumOrder : NUMBER;
      DiscountCalculation : STRING;
      LogoURL : STRING;
      TakesOrder : BOOLEAN;
      EnteredData : BOOLEAN;
END_ENTITY;
```

Attribute definitions:

Information: points to all of the resources/publications put out by this information provider.

RawInfoSource: defines what form the information provided is in. Currently this covers database, query interface, and web interface.

ReloadCall: captures the call which needs to be run to extract all information from the providers site in order to update the information in the system.

PurchaseResolver: points to the contact to whom orders can be sent if a user wants to make a purchase from this provider. This maybe a contact within the provider's organisation or it could be a wholesaler.

PostPackageCalculation: defines the calculation required to calculate the postage and packing charges that the provider imposes on orders.

PassOrderBy: an enumeration that describes the provider's preferred way of taking orders e.g. e-mail or fax.

SLAResponseTime: defines the number of days within which the publisher will fulfil an order as stated in the service level agreement between the provider and the service.

DiscountRate: the percentage discount that the provider offers.

MemberPrices: indicates whether or not the provider offers different prices to its members than to the general public.

MinimumOrder: the minimum value of order that the provider will accept.

DiscountCalculation: defines any discounts that the provider offers.

LogoURL: points to the provider's logo.

TakesOrders: indicates whether or not this provider accepts online orders.

EnteredData: indicates providers that have entered data into the system.

ENTITY: TechnicalInfoCentre

This is the top level descriptor of a technical information centre. It is a specialisation of CONNETComercialService with pointers to the information providers within the system.

EXPRESS specification:

```
ENTITY TechnicalInfoCentre
      SUBTYPE OF (CONNETCommercialService);
      Providers : SET [1:?] OF InformationProvider;
 UNIQUE
      UR1: ServiceID;
END_ENTITY;
```

Attribute definitions:

ServiceID: the CONNET specified unique ID for this service. This is used in all conversations between this service and the CONNET system.

Providers: points to the list of information providers who are active within this service.

ENTITY: TICResource

A specialisation of the Dublin Core resource description to include details of how to access the resource at the information providers site.

EXPRESS specification:

```
ENTITY TICResource
      SUBTYPE OF (DublinCore);
      InfoProviderInfoID : STRING;
      InfoProviderQuery : STRING;
      ElectronicCopy : OPTIONAL Document;
      Documents : SET [0:?] OF Document;
      DiscountStart : OPTIONAL DATE;
      DiscountEnd : OPTIONAL DATE;
      DiscountPercent : OPTIONAL NUMBER;
      MemberPrice : OPTIONAL NUMBER;
      InStock : BOOLEAN;
      InPrint : BOOLEAN;
      FullPrice : NUMBER;
      Currency : CURRENCY;
      CoverImageURL : STRING;
 UNIQUE
      UR1:  InfoProviderInfoID;
END_ENTITY;
```

Attribute definitions:

InfoProviderInfoID: this defines the unique ID for this resource at the information providers site.

InfoProviderQuery: defines the query that needs to be sent to the information providers site in order access all details of this resource.

ElectronicCopy: pointer to an electronic copy of the resource.

Documents: pointer to a set of documents describing or relating to the resource.

DiscountStart: the date from which any discount is valid

DiscountEnd: the date after which a discount is no longer valid

DiscountPercent: the discount, if any, that is available on the full price for this resource.

MemberPrice: the price, usually discounted, that members of the providers organisation are charged.

InStock: defines whether this resource is currently available.

InPrint: defines whether this resource is still in print.

FullPrice: the full price for the resource.

Currency: defines the currency used for MemberPrice and FullPrice according to the ISO list of currencies.

CoverImageURL: the location of an image associated with this resource.

APPENDIX G: Schema CONNET_WEC

The CONNET waste exchange centre (WEC) schema provides the core information required by a waste exchange centre to manage its information as well as connections through to CONNET for its services. The scope of a waste exchange centre is to be a trading post for a range of materials ranging from demolition through to excess products. The system is modifiable through its use of tables to operate outside the construction domain, in any domain where materials may be re-used. This could be set up nationally (as in the first demonstrator for CONNET) or within an organisation, or for a membership based organisation, or even internationally.

EXPRESS-G Diagram

Figure 12 Schema CONNET_WEC

TYPE: MaterialClass

Defines the major categories of material being dealt with in this exchange and if available or wanted.

EXPRESS specification:
```
TYPE MaterialClass = ENUMERATION OF
      (Wanted,
       Demolition,
       WasteMaterial,
       UnutilisedStock);
END_TYPE;
```

TYPE: NotificationType

Defines the major categories of material a user would want to be notified about against the material profile they would set up.

EXPRESS specification:
```
TYPE NotificationType = ENUMERATION OF
      (Never,
       IfWasteMaterial,
       IfUnutilisedStock,
       IfWasteOrUnutilised);
END_TYPE;
```

ENTITY: Location

Allows the definition of the location type table used inside this waste exchange centre.

EXPRESS specification:

```
ENTITY Location;
      Country : COUNTRY;
      Location : STRING;
      CentralPoint : STRING;
END_ENTITY;
```

Attribute definitions:

Country: defines the country the material resides in or is required in.

Location: defines the location in terms of postal code, town name, or county, etc. This is likely to be drawn from a table defined by each individual system, see the LocationTypeTable definition.

CentralPoint: defines a co-ordinate (latitude and longitude) for the site.

ENTITY: Materials

Describes a material which is available, or required, by a particular client of the system. It is a specialisation of OdererableItem.

EXPRESS specification:

```
ENTITY Materials
      SUBTYPE OF (OrderableItem);
      MaterialTypeID : MaterialType;
      LocationID : Location;
      ID : INTEGER;
      StartDate : DATE;
      EndDate : DATE;
      LatestDate : DATE;
      Cost : CURRENCYVALUE;
      Class : MaterialClass;
      RoughQuantity : STRING;
      RoughUnit : STRING;
      Description : STRING;
      TypeOfCost : STRING;
      Notify : NotificationType;
      DemolitionAddress : STRING;
      Company : Organisation;
 UNIQUE
      UR1:  ID;
END_ENTITY;
```

Attribute definitions:

MaterialTypeID: defines the name of the material (or category) which is available or required by the user. This will be defined in a table for each individual system, see the MaterialTypeTable definition.
LocationID: points to the location of the material available or required.
ID: a unique ID for the material request as specified by the waste exchange centre.
StartDate: specifies the date from which the material is available or can be utilised.
EndDate: specifies the date at which the material should be removed from view.
LatestDate: specifies the latest date at which the mentioned material would be of benefit for the client.
Cost: defines the approximate cost of the material.
Class: what type of material is being made available as defined in MaterialClass type. This also provides the distinction between wanted materials and available materials.
RoughQuantity: the approximate quantity which is available or required.
RoughUnit: the unit to be applied to the RoughQuantity. This specifies tonne, etc.
Description: the clients general description of the material available or required.
TypeOfCost: defines how the Cost field is interpreted in terms of per item, per tonne, the lot, etc.
Notify: defines against which Class of material the user wishes to be notified of the availablity or requirement for the material.
Company: points to contact details and contact name for the material being offered or searched for.
DemolitionAddress: the actual address of the site of the waste material, or the site at which the waste material is required.

ENTITY: MaterialType

Provides a definition of the whole MaterialTypeTable for this particular waste exchange centre.

EXPRESS specification:
```
ENTITY MaterialType;
      MaterialName : STRING;
      Type : MaterialTypes;
END_ENTITY;
```

Attribute definitions:
MaterialName: name of the material as specified for the MaterialTypeTable.
Type: defines the type of the material e.g. is it a waste material or unutilised stock.

ENTITY: WasteExchangeCentre

This is the top level descriptor of a waste exchange centre with pointers to waste product providers. It is a specialisation of CONNETCommercialService.

EXPRESS specification:
```
ENTITY WasteExchangeCentre
      SUBTYPE OF (CONNETCommercialService);
      MaterialTypeTable : INTEGER;
      LocationTypeTable : INTEGER;
      Materials : SET [1:?] OF Materials;
 UNIQUE
      UR1:  ServiceID;
END_ENTITY;
```

Attribute definitions:
MaterialTypeTable: CONNET ID for the material type table used in this waste exchange centre.
LocationTypeTable: CONNET ID for the location type table used in this waste exchange centre.
Materials: points to all materials required or available in the system at the current time.

APPENDIX H: Schema CONNET_WWC

The CONNET Who's Who (WWC) schema provides the core information required by a who's who centre to manage its information as well as connections through to CONNET for its services. The scope of a who's who centre is to be a gateway to the published technical information of a range of sources. This could be set up nationally or within an organisation, or for a membership based organisation, or even internationally.

EXPRESS-G Diagram

Figure 13 Schema CONNET_WWC

TYPE: MembershipTypes

Defines possible types of membership offered by associations.

EXPRESS specification:
```
TYPE MembershipTypes = ENUMERATION OF
     (FullMember,
      StudentMember);
END_TYPE;
```

ENTITY: Organisation

Describes organisations and self-employed individuals entered in the who's who directory. This is a specialisation of the CONNETCore Organisation.

EXPRESS specification:
```
ENTITY Organisation
        SUBTYPE OF (CONNETCore.Organisation);
        Staff : SET [1:?] OF Professional;
        MemberOf : SET [1:?] OF MembershipStatus;
        FieldsOfActivity : SET [1:?] OF FieldOfActivity;
        OrganisationCode : ClassificationCode;
END_ENTITY;
```

Attribute definitions:

Staff: the set of employees/members of an organisation.
MemberOf: the set of associations that this organisation is a member of.
FieldsOfActivity: the activities that the organisation is involved in.
OrganisationCode: the classification for the organisation as defined by the NACE classification system.

ENTITY: WhosWhoCentre

This is the top level descriptor of a who's who centre, it is a specialisation of CONNETService with pointers to the organisations in its directory.

EXPRESS specification:
```
ENTITY WhosWhoCentre;
     Organisations : SET [1:?] OF Organisation;
END_ENTITY;
```

Attribute definitions:

Organisations: pointer to the set of organisations that have entries in the directory.

ENTITY: Professional

Describes a person within an organisation. It is a specialisation of KnownUser.

EXPRESS specification:

```
ENTITY Professional
     SUBTYPE OF (KnownUser);
     MemberOf : SET [1:?] OF MembershipStatus;
     ProfesionalCode : ClassificationCode;
     Qualifications : FieldOfActivity;
END_ENTITY;
```

Attribute definitions:

MemberOf: the set of associations that this person is a member of.

ProfesionalCode: the ISCO 88, International Standard Classification of Occupation 1988, code for the persons profession.

Qualifications: is a link to this person's field of activity.

ENTITY: Association

EXPRESS specification:

```
ENTITY Association;
     AssociationDetails : Organisation;
END_ENTITY;
```

Attribute definitions:

AssociationDetails: Details of the association.

ENTITY: MembershipStatus

Describes a professional's or an organisation's membership of an association.

EXPRESS specification:

```
ENTITY MembershipStatus;
     MembershipType : MembershipTypes;
     Association : Association;
 INVERSE
     Contains : SET[1:?] OF Organisation FOR MemberOf;
END_ENTITY;
```

Attribute definitions:

MembershipType: an enumerated type describing the type of membership

Association: a pointer to the association that this person/organisation is a member of.

ENTITY: FieldOfActivity

Describes all the qualifications and services that a professional or an organisation have to offer.

EXPRESS specification:

```
ENTITY FieldOfActivity;
     ExtendedInfo : OPTIONAL ExtendedInformation;
     GeneralDescription : STRING;
     Services : OPTIONAL SET [1:?] OF STRING;
END_ENTITY;
```

Attribute definitions:

ExtendedInfo: a pointer to extra information supplied by users e.g. CVs, information sheets, project descriptions.
GeneralDescription: a description of the field.
Services: the set of services available in this field.

ENTITY: ExtendedInformation

Describes additional information supplied by users. This may be CVs, information sheets or project descriptions.

EXPRESS specification:

```
ENTITY ExtendedInformation;
     DetailedInformation : SET [0:?] OF Document;
     DetailedDescription : STRING;
END_ENTITY;
```

Attribute definitions:
DetailedDescription: a description of the information provided.
DetailedInformation: a pointer to the set of documents that the user provided.

APPENDIX I: API CONNET_Core

This appendix defines all the API functions which are offered by the CONNET thematic node for the benefit of the services which sit within its network. A Java API is currently available for all functions described in this appendix.

General policies on CONNET interfaces and API

All interfaces to the CONNET central node and the related services are implemented using HTTP. However, a language specific API is then provided to enable a more easily implementable interface to any of the services. This may not be as efficient as using some other protocols for the exchange of data on the Internet, such as the Z39.50, but it is robust, does not require an extra server and can be implemented with tools and software already in place to provide HTML pages for the end users. The API therefore contains a number of agreed-upon URLs. Since the location of the URLs are not defined and in the future will not be fixed, the location must be parameterised, for example after defining:

```
$LOC_CSC = 'www.fagg.uni-lj.si/connet/scs.cgi';
```

a function how to search library for software is defined like:

```
http://$LOC_CSC/Search?search=searchterm&format=formats
```

The API therefore defines the last part of the URL, the parameters passed to the service and the results returned.

Syntax for searchterm
Three types of searches are supported, simple search, field search and expression search. The first two are similar to AltaVista's simple syntax. Generally we use a full text search approach; all fields are searched for words written in the search

expression. Double quotes can be used to group words into phrases such as "New York". Several search words separated by blanks and grouped by quotes can be specified e.g.

```
"New York" sell painting
```

The database engines will try to find records that contain at least one of the terms defined above, but will sort results so that the records that match more terms are listed on top. The above search will also return records in that do not contain word 'sell'.

```
"New York" +sell painting
```

would limit the results to records which contain the word sell and then "New York" or "paining" or none of the other two. Similarly, pre-pending the term with a -, records that contain the term will not be listed:

```
-"New York" +sell +car
```

will list records which contain words sell and car and do not contain the phrase "New York". Search terms may also contain regular expressions that can help in making searches more precise. Searches are matched to word beginnings. Searching for "cat" will not match "tomcat" but will match "cathy". Searching for "cat\b" will not find "cathy" either.

Nicer search syntax allows fieldname:value type of syntax for example

```
+email:edu
```

Word starting in edu must be in email field.

```
-email:edu
```

Word starting edu must not be in email field.

If the search field includes character {, then everything to right (and including) this character is interpreted as a search expression. These search expressions are in fact Perl expressions with the following specialities: {fieldname} denotes value of a field; e.g.

```
{price} > 1000
```

will only find records where in the price field is larger than 1000.

While individual services may have their own advanced search syntax, the syntax used by the central core is kept simple to allow the query to be passed between all services, not just those of the same type.

The search term entered is used to search against all the text based fields in the database using a full text index.

Each word in the search term is used as a stemming word, that is, it will match to any word which starts with what has been entered. For example, if *build* is entered then the following words will match: build; builder; building; builders; buildings.

If two or more words are entered as part of he search term it is assumed that the user wants an *AND* condition between them. For example, if *build door* is entered then the system will find anything which contains words starting with both *build* and *door*.

To search for a phrase, the user must use a quoted string. For example, *"door frame"* will find anything which contains *door frame* together (e.g., 'Standard aluminum door frame, inside groove rubber gasket for efficiency.' would match).
The boolean search operators *AND* and *OR* can be used. By default any two words in a search term have an *AND* between them, but an *OR* operator must be put in explicitly. For example, if *build OR door* is entered then the system will find anything which contains words starting with *build* or *door* or both these words.

CONNET Core API

This section lists the API functions which provide information about CONNET as a whole and the top level services it offers.

ListCONNETProperties
Returns general information about CONNET in terms of the domain it operates in and the main components which comprise the system. This consists of a set of name-value pairs.

GetOwnerDetails
Returns all contact details for the owner/operator of the CONNET system. This allows email, phone, fax or normal mail contact to be established.

GetCertificate
Returns information on the certificate used to validate CONNET as the central node of a network of services and to establish a secure communication channel with the server.

GetHelpDeskURL
Returns the URL of the top level helpdesk associated with the CONNET system.

CONNET Service Based API

The following API functions provide information about the individual services which are part of the CONNET thematic network.

ListCONNETServiceTypes
Returns a list of all types of services currently offered by CONNET, currently this comprises the seven services, Classification, or UserProfile.

ListCONNETServices
Returns a list of all services known to CONNET of the specified service type (e.g., all WWC known to CONNET).

SubmitServiceQuery
Searches for all services which match the formed query string. Returns a list of matching services.

GetServiceDetails
Returns details of a specified service in terms of the properties of the CONNETService class described in another appendix.

SubmitFailureReport
Attaches a failure report to the named service in terms of the AvailabilityStatistics class described in Appendix A.

ListServiceClassificationSystems
Returns all classification systems used by the named service.

ListServiceTablesUsed
Returns all tables used by the named service (e.g., location codes, or material codes).

ListTableDetails
Returns the full contents of the named table used by a particular service (see above).

ListServiceParents
Returns all services which can answer a wider question than possible by the named service (e.g., for an organisation based service to reach a national service).

SubmitServiceQueryToParents
Requests CONNET to pass the specified query to the parents of the specified service and to collate the replies from each of these services.

SubmitServiceQueryToOtherService
Requests CONNET to pass the specified query to a different service type to attempt to answer the query for its domain (e.g., to take a WWC query and pass it to a BPC for info on what a person has done in Best Practice field).

GetServiceDataModel
Returns the CONNET maintained EXPRESS data model for the specified service type.

ListServiceAPIs
Returns all the possible API specifications for a particular service type (e.g., Java, C++, Perl).

GetServiceAPIDetails
Returns the API for the specific service type in the requested language.

Not sure what this means. Maybe it could return a URL for the service that shows the syntax needed to submit a search to the service via HTTP e.g. for the BPC this might be http://cig.bre.co.uk/servlet/BPC_search?query=searchterm&fromat=text

CONNET User Profile API

The following API functions relate to the management of user profiles by CONNET for the benefit of all services in the CONNET network.

SubmitUserIdentifyQuery
Searches for the unique ID of the nominated user. If the user is known then the existing ID will be returned, otherwise a new user record will be initiated and the ID for this new record passed back.

GetUserDetails
Returns the top level properties for a user in the CONNET system. This comprises the properties specified for the KnownUser class in Appendix A.

ModifyUser
Updates the details for the specified user to the values specified for each class property.

DeleteUser
Removes the specified user from active use in the CONNET user profile databases. Their information will however be retained for data mining purposes in the future.

GetUserServiceDetails
Returns the service specific details for the named user and named service. This comprises PropertySet based information as described in Appendix A.

AddUserService
Creates service specific information in the CONNET user profile databases.

ModifyUserService
Updates the details for the specified user in the specified service for each PropertySet property and value.

DeleteUserService
Removes the specified user's details from the named service from active use. Their information will however be retained for data mining purposes in the future.

ListUserServiceProfiles
Returns all profiles held by the named user for the named service.

GetProfileDetails
Returns the details for the specified profile for the specified user in the specified service. This comprises Profile class properties as specified in Appendix A.

AddProfile
Creates a new profile for the named user in the named service.

ModifyProfile
Updates the details for the specified profile for the specified user in the specified service.

DeleteProfile
Removes the specified profile from the named service for the named user from active use. This information will however be retained for data mining purposes in the future.

SubmitUserTrace
Creates a new trace record for the specified user recording an activity in a specific service in the CONNET system.

ListProfiles
Returns the set of profiles for the specified user.

CONNET Classification Tool API

The following API functions relate to the use, selection, and navigation through classifications systems known to the CONNET central node.

ListClassificationSystems
Returns the set of known classification inside the CONNET system

ListClassificationTables
Returns the set of tables which relate to the specified classification system. For a classification system with no tables this will be a single dummy entry permitting access through to the actual codes.

ListClassificationCodes
Returns the total set of classification codes which relate either to the classification system as a whole, or to a nominated table.

GetClassificationCodeDetails
Returns all details of a specific classification code.

ListParentClassificationCodes
Returns the list of all parent codes for a specified classification code in a hierarchical classification system.

ListChildClassificationCodes
Returns the list of all children codes for a specified classification code in a hierarchical classification system.

SubmitClassificationCodeQuery
Searches for all classification codes in the named system (or across multiple classification systems) which match the specified query. This function can request the use of a thesaurus to help broaden the query.

APPENDIX J: API CONNET_BPC

This appendix defines all the API functions, which should be offered by best practice centre operating within the CONNET thematic node.

Best Practice Centre Core API

ListLocalParameters
Returns the service specific details for this service. This comprises BestPracticeCentre class information as described in another appendix.

ListCONNETParameters
Returns information specific to the use of this service with CONNET. This comprises service ID and certificate information.

GetOwnerDetails
Returns all contact details for the owner/operator of this service. This allows email, phone, fax or normal mail contact to be established.

GetCertificate
Returns information on the certificate used to validate this service with CONNET as the central node of a network of services and to establish a secure communication channel with the server.

ListSectorInitiative
Returns a list of the sector initiatives available in the CONNET BPC database.

ListThemes
Returns a list of the themes available in the CONNET BPC database.

ListInformationTypes
Returns a list of the information types available in the CONNET BPC database.

ListInformationSubTypes
Returns a list of the information sub-types available in the CONNET BPC database.

SubmitQuery
Searches for all of the information available in the database, which matches the specified query. Returns a list of matching or partially matching entries.

GetInformationDetails
Returns full details of the specified information.

GetDocumnetDetails
Returns full details of the specified best practice document.

GetActivityDetails
Returns full details of the specified activity.

GetAdvisorSchemeDetails
Returns full details of the specified advisor/scheme.

FileDownload
Downloads the specified file from a web-server and saves it to a specified local location.

APPENDIX K: API CONNET_CSC

This appendix defines all the API functions which should be offered by a calculation and software centre operating within the CONNET thematic node. A Perl API is currently available for all functions described in this appendix.

Calculation and Software Centre Core API

ListLocalParameters
Returns the service specific details for this service. This comprises CalculationSoftwareCentre class information as described in Appendix E.

ListCONNETParameters
Returns information specific to the use of this service with CONNET. This comprises service ID and certificate information.

GetOwnerDetails
Returns all contact details for the owner/operator of this service. This allows email, phone, fax or normal mail contact to be established.

GetCertificate
Returns information on the certificate used to validate this service with CONNET as the central node of a network of services and to establish a secure communication channel with the server.

ListTablesUser
Returns the set of named tables used by this service to control the values for many of the properties of the software resources recorded in the system.

ListTableDetails
Returns the table of codes for the specified table used in the system.

SubmitQuery
Searches for all software available which match the specified query. Returns a list of matching or partially matching entries.

GetFullDetails
Returns details of the specified software in the system. This includes all properties of the Software class specified in Appendix E as well as Dublin Core properties as described in Appendix A.

SubmitVendorQuery
Searches for all vendors which match the specified query. Returns a list of matching or partially matching entries.

GetVendorDetails
Returns all details of the specified software vendor in the system. This comprises the properties of the Vendor class as described in Appendix E.

AddVendor
Creates a new vendor in the system, which is required before software can be added.

AddSoftware
Creates a new software record in the system for the specified vendor. The properties input are as described for the software class specified in Appendix E as well as Dublin Core properties as described in Appendix A.

ListReviews
Returns the set of reviews for the specified software item in the system.

GetReviewDetails
Returns all details of the specified review for the specified software in the system. This comprises the properties of the Review class as described in Appendix E.

AddReview
Creates a new review for the specified piece of software in the system. The review comprises the properties of the Review class as described in Appendix E.

APPENDIX L: API CONNET_ESC

This appendix defines the entire API functions, which should be offered by equipment, supply centre operating within the CONNET thematic node.

Equipment Supply Centre Core API

ListLocalParameters
Returns the service specific details for this service. This comprises EquipmentSupplyCentre class information as described in Appendix D.

ListCONNETParameters
Returns information specific to the use of this service with CONNET. This comprises service ID and certificate information.

GetOwnerDetails
Returns all contact details for the owner/operator of this service. This allows email, phone, fax or normal mail contact to be established.

GetCertificate
Returns information on the certificate used to validate this service with CONNET as the central node of a network of services and to establish a secure communication channel with the server.

ListClassificationSystems
Returns a list of classification systems available in the CONNET ESC database.

ListClassificationTables
Returns a list of classification tables available for the given classification system in the CONNET ESC database.

ListClassificationCodes
Returns a list of classification codes available for the given classification table in the CONNET ESC database.

ListObjectClasses
Returns a list of all ObjectClasses (equipment categories) for which there is a set of searchable properties available in the system. This can be based on a classification code or the name of an Equipment Item

ListSearchableProductProperties
Returns a list of the available searchable fields (PropertyDefinitions) associated with a given ObjectClass.

SubmitQuery
Searches for all items (equipment and facilities) available in the database which match the specified query. Returns a list of matching or partially matching entries.

GetFullItemDetails
Returns full details of the specified item in the database. This includes all properties values of the item class specified in Appendix D as well as dynamically defined properties stored in the PropertySet class described in Appendix A.

ListItemDocuments
Returns the set of documents, which are associated with the specified item.

GetInformationProviderDetails
Returns all contact details for the information provider or author for a specified item. This allows email, phone, fax or normal mail contact to be established.
GetOwnerManufacturerDetails
Returns all contact details for the owner of a facility or the manufacturer of equipment for a specified item. This allows email, phone, fax or normal mail contact to be established.

ListOwnerManufacturerDocuments
Returns the set of documents, which are associated with the specified manufacturer of equipment or an owner of a facility.

SubmitOwnerManufacturerQuery
Searches for all owners or manufacturers who match the specified query. Returns a list of matching or partially matching entries.

GetOwnerManufacturerDetails
Returns details of the specified owner or manufacturer. This comprises the properties of the Organisation class specified in Appendix A.

ListEquipmentSuppliers
Returns all suppliers of the specified equipment.

ListEquipmentSupplierDocuments
Returns the set of documents, which are associated with the specified supplier.

SubmitEquipmentSupplierQuery
Searches for all suppliers who match the specified query. Returns a list of matching or partially matching entries.

GetEquipmentSupplierDetails
Returns details of the specified supplier. This comprises the properties of the Organisation class specified in Appendix A.

GetFacilityDetails
Returns details of the specified facility. This comprises the properties of the Organisation class specified in Appendix A.

ListFacilityDocuments
Returns the set of documents, which are associated with the specified facility.

FileDownload
Downloads the specified file from the Owner/manufacturer, or supplier, web-server and saves it to a specified local location.

APPENDIX M: API CONNET_NS

This appendix defines all the API functions which should be offered by a news service operating within the CONNET thematic node. A Java API is currently available for all functions described in this appendix.

News Service Core API

ListLocalParameters
Returns the service specific details for this service. This comprises NewsService class information as described in Appendix F.

ListCONNETParameters
Returns information specific to the use of this service with CONNET. This comprises service ID and certificate information.

GetOwnerDetails
Returns all contact details for the owner/operator of this service. This allows email, phone, fax or normal mail contact to be established.

GetCertificate
Returns information on the certificate used to validate this service with CONNET as the central node of a network of services and to establish a secure communication channel with the server.

SubmitQuery
Searches for all news items (web pages) available which match the specified query. Returns a list of matching or partially matching entries. This does not provide enough

information to go to the source of the news item, this requires the function RetrieveURL described below.

RetrieveURL
Activates the call through to the news provider's web site for the specified news item to return the full web page through to the caller. This function allows the service to log when a user goes to a news provider's site for a found item.

SubmitHostURLQuery
Searches for all news provider sites which match the specified query. Returns a list of matching or partially matching entries.

GetNewsURLDetails
Returns the details of the news provider site. This comprises the properties defined for the NewsURL class in Appendix F.

AddNewsURL
Creates a new news provider site entry in the system with the parameters defined for the NewsURL class in Appendix F.

ListProfiles
Returns the set of profiles for the specified user.

GetProfileDetails
Returns the details for the specified profile for the specified user. This comprises NewsProfile class properties as specified in Appendix F.

AddProfile
Creates a new profile for the named user. This comprises NewsProfile class properties as specified in Appendix F.

ModifyProfile
Updates the details for the specified profile for the specified user. This comprises NewsProfile class properties as specified in Appendix F.

DeleteProfile
Removes the specified profile for the named user from active use. This information will however be retained for data mining purposes in the future.

APPENDIX N: API CONNET_TIC

This appendix defines all the API functions which should be offered by a technical information centre operating within the CONNET thematic node. A Java API is currently available for all functions described in this appendix.

Technical Information Centre Core API

ListLocalParameters
Returns the service specific details for this service. This comprises TechnicalInfoCentre class information as described in Appendix B.

ListCONNETParameters
Returns information specific to the use of this service with CONNET. This comprises service ID and certificate information.

GetOwnerDetails
Returns all contact details for the owner/operator of this service. This allows email, phone, fax or normal mail contact to be established.

GetCertificate
Returns information on the certificate used to validate this service with CONNET as the central node of a network of services and to establish a secure communication channel with the server.

ListClassificationSystems
Returns the set of known classification inside the service

ListClassificationTables
Returns the set of tables which relate to the specified classification system. For a classification system with no tables this will be a single dummy entry permitting access through to the actual codes.

ListClassificationCodes
Returns the total set of classification codes which relate either to the classification system as a whole, or to a nominated table.

SubmitQuery
Searches for all technical information which matches the specified query. Returns a list of matching or partially matching entries.

GetFullDetails
Returns details of the specified technical information resource in the system. This is somewhat less than the full Dublin Core set of information which the information providers would not wish to be given to any requesting system. The returned set of information also does not include the data required to retrieve the full information from a provider's site. This requires the RetrieveInformationProviderDocument function.

ListInformationProviders
Returns the full set of information providers who are utilised in this service.

RetrieveInformationProviderDocument
Activates the call through to the information provider's web site for the specified resource to return the full details through to the caller. This function allows the service to log when a user goes to an information provider's site for a found resource.

APPENDIX O: API CONNET_WEC

This appendix defines all the API functions which should be offered by a waste exchange centre operating within the CONNET thematic node. A Java API is currently available for all functions described in this appendix.

Waste Exchange Centre Core API

ListLocalParameters
Returns the service specific details for this service. This comprises WasteExchangeCentre class information as described in Appendix C.

ListCONNETParameters
Returns information specific to the use of this service with CONNET. This comprises service ID and certificate information.

GetOwnerDetails
Returns all contact details for the owner/operator of this service. This allows email, phone, fax or normal mail contact to be established.

GetCertificate
Returns information on the certificate used to validate this service with CONNET as the central node of a network of services and to establish a secure communication channel with the server.

GetMaterialTypeName
Returns the name of the table holding the material type codes used in this system.

ListMaterialTypes
Returns the table of material types used in the system.

GetLocationTypeName
Returns the name of the table holding the location codes used in this system.

ListLocationTypes
Returns the table of locations used in the system.

AddMaterialWanted
Creates a new material wanted record in the system with parameters as specified in the Materials class in Appendix C.

AddWasteMaterial
Creates a new waste material available record in the system with parameters as specified in the Materials class in Appendix C.

AddUnutilisedStock
Creates a new unutilised stock available record in the system with parameters as specified in the Materials class in Appendix C.

AddDemolition
Creates a new demolition record in the system with parameters as specified in the Materials class in Appendix C.

SubmitQuery
Searches for all materials available or wanted which match the specified query. Returns a list of matching or partially matching entries.

GetFullDetails
Returns details of the specified waste available, or wanted, resource in the system.

DeleteEntry
Removes the specified material available or wanted from the service.

APPENDIX P: API CONNET_WCC

This appendix defines all the API functions, which should be offered by whose who centre operating within the CONNET thematic node.

Who's Who Centre Core API

ListLocalParameters
Returns the service specific details for this service. This comprises Who'sWhoCentre class information as described in another appendix.

ListCONNETParameters
Returns information specific to the use of this service with CONNET. This comprises service ID and certificate information.

GetOwnerDetails
Returns all contact details for the owner/operator of this service. This allows email, phone, fax or normal mail contact to be established.

GetCertificate
Returns information on the certificate used to validate this service with CONNET as the central node of a network of services and to establish a secure communication channel with the server.

ListClassificationSystems
Returns a list of classification systems available in the CONNET WWC database.

ListClassificationTables
Returns a list of classification tables available for the given classification system in the CONNET WWC database.

ListClassificationCodes
Returns a list of classification codes available for the given classification table in the CONNET WWC database.

SubmitQuery
Searches for all organisations and/or professionals available in the database, which match the specified query. Returns a list of matching or partially matching entries.

SubmitOrgainsationQuery
Searches for all organisations available in the database, which match the specified query. Returns a list of matching or partially matching entries.

GetOrganisationDetails
Returns full details of the specified organisation. This includes all contact details as well as a list of field of activity, that includes a general description and services provided by this organisation

GetOrganisationStaff
Returns a list of staff members of the specified organisation. Extended information is available on each staff member as provided individually by staff member or organisation.

ListOrganisationDocuments
Returns the set of documents, which are associated with the specified organisation.

GetOrganisationAssociationMemberShip
Returns a list of associations in which the specified organisation is a member.

SubmitProfessionalQuery
Searches for all professionals available in the database, which match the specified query. Returns a list of matching or partially matching entries.

GetProfessionalDetails
Returns full details of the specified professional. This includes all contact details as well as a list of field of activity, that includes a general description and services provided by the professional

GetProfessionalAssociationMemberShip
Returns a list of associations in which the specified professional is a member.

GetExtendedInformation
Returns extended information on a field of activity as provided by an organisation or a professional listed in the database. This includes a detailed description and a set of documents.

ListOrganisationsInAssociation
Returns the set of all organisations that are registered members of this specified association and their membership status.

ListProfessionalsInAssociation
Returns the set of all professionals that are registered members of this specified association and their membership status.

SubmitAssociationQuery
Searches for all associations available in the database, which match the specified query. Returns a list of matching or partially matching entries.

GetAssociationDetails
Returns full details for the specified association. This includes all contact details as well as, if one is included, a list of field of activity, that includes a general description and services provided by this association

ListAssociatedAssociations
Returns the set of all associated associations for this specified association.

FileDownload
Downloads the specified file from a web-server and saves it to a specified local location.