Purpose

This document is a collaboration space to explore improvements to the working definition of “profile” previously adopted by the DXWG:

“A named set of constraints on one or more identified base *specifications*, including the identification of any implementing subclasses of datatypes, semantic interpretations, vocabularies, options and parameters of those base specifications necessary to accomplish a particular function.

This definition includes what are often called "application profiles", "metadata application profiles", or "metadata profiles".

Source: deliberations of the DXWG. See ProfileContext wiki page.

Editing guidelines

1) Edit in suggestion mode if proposing a change
2) Add comments using the commenting mechanisms to specific sections, and reply to comments directly where applicable
3) Comments and suggestions may be resolved during working group meetings
4) Feel free to add new sections detailing any specific concerns (leave Requirements at end as a read-only appendix)
5) Provide evidence for any assertions not already encompassed by UCR

Working draft definition

A named set of constraints on one or more identified base *specifications*, including the identification of any implementing subclasses of datatypes, semantic interpretations, vocabularies, options and parameters of those base specifications necessary to accomplish a particular function.

Makes proposed edits in “suggestion mode” or provide a new option below.

Meta-concerns

1. All documents should be using and consistent with a common definition since external stakeholders will expect this of a single Working Group.

2. As far as possible the definition should be consistent with usages of the word “profile” in other contexts, especially W3C and IETF usages.
3. It must be recognised that for specific communities usages of the term “profile” will naturally be narrower in scope, since it will relate to that community’s specific context - being profiles of things the community itself is concerned with (for example use of profile qualifier in MIME types, DCAT profiles, Dublin core application profiles etc.)

Specific concerns

Tabular summary

Name	***Nature***	***Proposed Resolution***
Definition boundaries	policy	
“Constraint”	Terminology choice	
“Specification” semantics	Definition	
Conformance	Expressivity	
<please add your favourite issues here>		

Details

Definition boundaries

At what stage do we rely on (OED?) definitions and common usage rather than trying to pin down every word in every sentence?

Suggested “one more step” candidates:
“Specification”,
“Constraint” (and./or “requirement”)
“conformance”

“Constraint”

The term constraint may be hard to understand.
OED definition is adequate: “*A limitation or restriction*.” The context is however that a constraint expresses a way to realise an underlying requirement the profile addresses.

possibly use the word “requirement” - and in the guidance document state that requirements SHOULD be testable and expressed in an platform-appropriate constraint language (eg SHACL or SHex for RDF, Schematron for XML)

“Specification” semantics

In the conneg work the functionality is mainly about declaring conformance with a specification, which may or may not be thought of as a profile.

This is technically not a problem, because any specification may be thought of as a profile of itself with an empty set of constraints, however these logical and programming concepts seem to be hard for non-technical audiences to immediately understand.

The problem arises from the lack of a formal, canonical, W3C definition and model for the general concept of a specification, so a response is required. dct:Standard is very vague and technically acceptable, but Profile (rdfs:subClassOf dct:Standard) is being asked to “do too much work:”

Options:

1) Explanatory text
2) Formalising a definition of specification and including it as text
3) Formalising a definition of specification in a separate new vocabulary (and importing into profiles)
4) Formalising a definition of specification and adding it the profiles ontology and possibly renaming it to cover the broader scope
5) ??

The current scope reflects the WG charter, and in practice formalising a model (the profiles ontology) is probably necessary to test and show our definition fits real world examples and requirements, as well as meeting most of the UCR requirements for profile description.

Conformance

The OED definition is fine conformance w.r.t to current definition.

“another term for conformity.”
E.g. *"conformance to international standards"*
“conformity”=> Compliance with standards, rules, or laws.
“Compliance” => The state or fact of according with or meeting rules or standards
“According”, “meeting” … etc etc etc etc

However, it may be worth making the concept of conformance a more explicit part of a general model of specifications (NB this is what the OGC “modular specification” (a specification for specifications) requires).

Won’t change anything substantive, but may help disentangle the confusion around “re-use” vs “profiling”.

There would probably be a constraint on a profile that its “conformance target” is a valid subclass of the conformance target of any base specification (remember “base” is a role, not a class). (another example where a thing is a valid subclass of itself by default)

Requirements

(from UCR process and document - this will need to be updated if new requirements identified and agreed)

6.10.1 Named collections of terms [RPFNCTERMS]

Profiles are "named collections of properties" or metadata terms (if not RDF).

§ 5.41 Vocabulary constraints [ID41]

github:issue/275
EDITOR'S NOTE
This requirement could also be in roles/functions if there is nothing there about defining terms.

6.10.2 Multiple base specifications [RPFMBSPEC]

A profile can have multiple base specifications.

§ 6.10.4 Profile inheritance [RPFINHER]

§ 5.37 Europeana profile ecosystem: representing, publishing and consuming application profiles of the Europeana Data Model (EDM) [ID37]

github:issue/268

6.10.3 Profile of profiles [RPFPP]

One can create a profile of profiles, with elements potentially inherited on several levels.

§ 5.37 Europeana profile ecosystem: representing, publishing and consuming application profiles of the Europeana Data Model (EDM) [ID37]

github:issue/270

6.10.5 Data publication according to different profiles-1 [RPFDAPUB-1]

Some data may conform to one or more profiles at once

§ 5.3 Responses can conform to multiple, modular profiles [ID3]

github:issue/608
NOTE

Group discussions raised this requirement as the result of a split of a earlier, wider requirement, with the aim of adding it to the the general 'profile' category. However it overlaps with the following requirement and there is a lot of uncertainty on where this requirement should be categorized.

6.10.6 Data publication according to different profiles-2 [RPFDAPUB-2]

Data publishers may publish data according to different profiles, either simultaneously (e.g. in one same data "distribution") or in parallel (e.g. via content negotiation)

§ 5.37 Europeana profile ecosystem: representing, publishing and consuming application profiles of the Europeana Data Model (EDM) [ID37]

github:issue/274
NOTE

There is a lot of uncertainty on where this requirement should be categorized. There is a big 'profile negotiation' flavour to it, but it is not only about negotiaton. We are keeping it here for now as we are not sure where it should be. Group discussions list it as a general requirement, but it could be categorized instead as a requirement for profile negotiation, or DCAT distribution, or both. In addition, it highly overlaps with the previous requirement (and we have edited the heading to reflect it).

6.11 Profiles — Profile functionality

This section is non-normative.

Requirements covering aspects of how profiles are being used, i.e. what functionality may they express or support, for example validation, or documentation of data.

6.11.1 Human-readable definitions [RPFHRDEF]

Profiles can have human-readable definitions of terms and input instructions.

§ 5.46 Profile support for input functions [ID46]

github:issue/283

6.11.2 Global rules for descriptive content [RPGRDC]

There needs to be a property in the profile where the rules for the descriptive content can be provided. This would apply to the entire profile.

§ 5.42 Metadata Guidance Rules [ID42]

github:issue/255

6.11.3 Data validation [RPFVALID]

A profile may be (partially) "implemented" by "schemas" (in [OWL-REF], [SHACL] [XMLSCHEMA11-1]...) that allow different levels of data validation

§ 5.37 Europeana profile ecosystem: representing, publishing and consuming application profiles of the Europeana Data Model (EDM) [ID37]

github:issue/273
NOTE

This requirement, which is a bit redundant with github:issue/279, has been shifted towards the function of data validation instead of focusing on the representations that enable it.

6.11.4 External specifications for individual properties [RPFESIP]

Profiles should be able to indicate which external specifications are expected to be applied/have been applied to values of individual properties.

§ 5.43 Description of dataset compliance with standards [ID43]

github:issue/280

6.11.5 Validity rules [RPFVALIDR]

Profiles may provide rules governing value validity.

§ 5.41 Vocabulary constraints [ID41]

github:issue/277

6.11.6 Value lists for data elements [RPFVLIST]

Profiles may provide lists of values to pick from in order to populate data elements.

§ 5.46 Profile support for input functions [ID46]

github:issue/282

6.11.7 Cardinality rules [RPFCARDR]

Profiles may provide rules on cardinality of terms (including "recommended").

§ 5.41 Vocabulary constraints [ID41]

github:issue/276

6.11.8 Dependency rules [RPFDEPR]

Profiles may express dependencies between elements of the vocabulary (if A then not B, etc.).

§ 5.41 Vocabulary constraints [ID41]

github:issue/278

6.11.9 User interface support [RPFUI]

Profiles can have what is needed to drive forms for data input or for user display.

§ 5.46 Profile support for input functions [ID46]

github:issue/281

6.12 Profiles — Profile distributions

This section is non-normative.

Requirements covering aspects of how (part of) profiles are concretely expressed/represented, using the languages that allow such expression.

6.12.1 Profile documentation [RPFDOCU]

A profile should have human-readable documentation that expresses for humans the main components of a profile, which can also be available as machine-readable resources (ontology or schema files, [SHACL] files, etc). This includes listing of elements in the profile, instructions

and recommendations on how to use them, constraints that determine what data is valid according to the profile, etc.

github:issue/272

6.12.2 Schema implementations [RPFSCHEMAS]

Profiles may be written in or may link to a document or schema in a validation language ([ShEx], [SHACL], [XMLSCHEMA11-1]).

§ 6.11.3 Data validation [RPFVALID]

§ 5.41 Vocabulary constraints [ID41]

github:issue/279
NOTE

This requirement, which is a bit redundant with github:issue/273, has been shifted towards the notion of distributions of schemas, instead of focusing on the general validation function.

6.13 Profiles — Profile metadata

This section is non-normative.

6.13.1 Profile metadata [RPFMD]

Profiles must be discoverable through a machine-readable metadata that describes what is offered and how to invoke the offered profiles.

§ 6.14.6 Profile metadata use [RPFMDUSE]

§ 5.5 Discover available content profiles [ID5]

github:issue/288

6.13.2 Documenting ecosystems of profiles [RPFDECOS]

From the perspective of management of profiles, and guidance to users and data experts, ecosystems of profiles should be properly described (e.g. in profile catalogues/repositories), especially documenting the relationships between profiles and what they are based on, and between profiles that are based on other profiles.

§ 5.37 Europeana profile ecosystem: representing, publishing and consuming application profiles of the Europeana Data Model (EDM) [ID37]

github:issue/271

6.14 Profile and content negotiation

This section is non-normative.

Requirements covering the provision, look-up and negotiation of profiles and compliant contents.

6.14.1 Profile negotiation [RPFNEG]

Enable the ability to negotiate the data profile via http, similar to the negotiation of metadata formats today.

§ 5.30 Standard APIs for metadata profile negotiation [ID30]

github:issue/265

6.14.2 Profile link relations [RPFLREL]

There needs to be a way to encode the necessary relations using an http link header.

§ 5.30 Standard APIs for metadata profile negotiation [ID30]

github:issue/266
EDITOR'S NOTE
There are other rewordings, more recent, in the issue and in the notes of Group discussion. Jaro and Lars will come up with one. More details on relations are also provided in github.

6.14.3 Profile discoverability support [RPFDISCO]

Profiles should support discoverability via search engines.

§ 5.40 Discoverability by mainstream search engines [ID40]

github:issue/222
NOTE
According to the notes of group discussions, this requirement has not yet been approved.

6.14.4 Server-side profile support [RPFSUP]

A client should be able to determine which profiles are supported by a server, and with which content types or other properties, in order to receive the one most appropriate for their use.

§ 5.2 Detailing and requesting additional constraints (profiles) beyond content types [ID2]

github:issue/285

6.14.5 Lookup of profile details [RPFLUP]

There should be a way to look up additional information about a profile - this may be machine readable for run-time mediation or used to develop or configure a client.

§ 5.2 Detailing and requesting additional constraints (profiles) beyond content types [ID2]

github:issue/286

6.14.6 Profile metadata use [RPFMDUSE]

NOTE

According to the notes of group discussions, this requirement has not yet been approved.

Metadata about server profile support can be used for discovery and mediated traversal via content negotiation.

§ 6.13.1 Profile metadata [RPFMD]

§ 5.5 Discover available content profiles [ID5]

github:issue/264

6.14.7 Profile selection on request [RPFSREQ]

When requesting a representation, a client must be able to specify which profile it prefers the representation to adhere to. This information about the requested profile is not a replacement for content type (e. g. application/xml), language (e. g. zh-Hant) nor any other negotiated dimension of the representation.

§ 5.5 Discover available content profiles [ID5]

github:issue/289

6.14.8 Response conformance to multiple profiles [RPFRCONF]

Some data may conform to one or more profiles at once. A server can indicate that a response conforms to multiple profiles.

§ 5.3 Responses can conform to multiple, modular profiles [ID3]

github:issue/287

6.14.9 Profile identifier [RPFID]

A profile must have an identifier that can be served with a response to an API or HTTP request.

§ 5.2 Detailing and requesting additional constraints (profiles) beyond content types [ID2]

github:issue/284

6.14.10 Profile alias [RPFALIAS]

A short token to specify a profile may be used as long as there is a discoverable mapping from it to the profile's identifying URI.

§ 5.5 Discover available content profiles [ID5]

github:issue/290