

# Specification



## OpenPeppol AISBL

---



## Peppol Transport Infrastructure ICT - Models

## Peppol Directory



**Version: 1.1.1**

**Status: In use**



**Editors:**

**Philip Helger, OpenPeppol Operating Office  
Ger Clancy, IBM**

## Revision History

Version	Date	Description of changes	Approved by
	2015-08-03	Updated details on the PD-SML connection Editorial changes in 5.2.2, 6.1.1 Changed link in chapter 7	PH
	2015-09-10	Moved chapter 4.1.5 to become 4.1.1 Fixed example requests in the PD Indexer section Fixed HTTP return codes in Indexer section Added chapter 4.1.7 on internal processing	PH
	2016-01-20	Updated to Peppol Directory Adopted to separate SMP interface Adopted to new XSD	PH
	2016-04-22	Introduction less technical Logical separation between Publisher and Indexer less important	PH
	2016-11-28	Updated to MC decisions Stripped down Business Card data	PH
1.0	2016-12-05	Updated contributor list	PH
	2017-04-03	Updated REST API description slightly	PH
1.1	2018-07-17	Updated REST API to match implementation Chapter 5.2: fixed example; added note on encoding Added chapter "Usage outside of Peppol" The BC data model now has a multilingual entity name Minor editorial corrections and clarifications added	PH
1.1.1	2020-10-15	Updated to the new Peppol branding Changed the spelling from "PEPPOL" to "Peppol" Added new Introduction chapter including References	PH

### Statement of originality

This deliverable contains original unpublished work except where clearly indicated otherwise. Acknowledgement of previously published material and of the work of others has been made through appropriate citation, quotation or both.

### Statement of copyright



*This deliverable is released under the terms of the Creative Commons Licence accessed through the following link: <http://creativecommons.org/licenses/by-nc-nd/4.0/>.*

*You are free to:*

**Share** — *copy and redistribute the material in any medium or format.*

*The licensor cannot revoke these freedoms as long as you follow the license terms.*

## 1 **Contributors**

### 2 **Organisations**

3 BRZ (Bundesrechenzentrum)<sup>1</sup>, Austria, <http://www.brz.gv.at/>

4 IBM, <http://www.ibm.com>

5 ESV, The Swedish National Financial Management Authority, <http://www.esv.se>

### 6 **Persons**

7 Philip Helger, OpenPeppol Operating Office (editor)

8 Ger Clancy, IBM

9 Martin Forsberg, ESV

10 Georg Birgisson, Midran Ltd.

---

<sup>1</sup> English: Austrian Federal Computing Centre

## 11 **1 Introduction**

### 12 **1.1 Audience**

13 This document describes a Peppol policy and guidelines for use of identifiers within the Peppol  
14 network. The intended audience for this document are organizations wishing to be Peppol enabled  
15 for exchanging electronic documents, and/or their ICT-suppliers. More specifically it is addressed  
16 towards the following roles:

- 17     ▶ ICT Architects
- 18     ▶ ICT Developers
- 19     ▶ Business Experts

### 20 **1.2 Terminology**

21 The keywords "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD  
22 NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as  
23 described in RFC 2119 [RFC2119].

### 24 **1.3 References**

- 25 [PEPPOLSMP] "Peppol Service Metadata Publishing (SMP) 1.2.0",  
26 [https://docs.peppol.eu/edelivery/smp/PEPPOL-EDN-Service-Metadata-Publishing-  
27 1.2.0-2021-02-24.pdf](https://docs.peppol.eu/edelivery/smp/PEPPOL-EDN-Service-Metadata-Publishing-1.2.0-2021-02-24.pdf)
- 28 [RFC2119] "Key words for use in RFCs to Indicate Requirement Levels",  
29 <http://www.ietf.org/rfc/rfc2119.txt>
- 30 [PFUOI4] "Peppol Policy for use of Identifiers 4.1.0",  
31 [https://docs.peppol.eu/edelivery/policies/PEPPOL-EDN-Policy-for-use-of-identifiers-  
32 4.1.0-2020-03-11.pdf](https://docs.peppol.eu/edelivery/policies/PEPPOL-EDN-Policy-for-use-of-identifiers-4.1.0-2020-03-11.pdf)

## 33 **2 Introduction to Peppol Directory (non-normative)**

34 The goal of this document is to describe the architecture and interfaces of the Peppol Directory (PD;  
35 formerly known as Peppol Yellow Pages) project. The goal of the PD project is to create a publicly  
36 available, searchable list of all Peppol participants with their respective metadata like company  
37 name, country code, etc. (for details see chapter 5.1). The PD is not meant to replace existing Peppol  
38 components but to be an aggregator for data that is contained in existing Peppol SMPs  
39 [PEPPOLSMP].

40 An additional singleton service is added to the Peppol infrastructure: the so-called **PD Server**. It is  
41 filled with electronic **Business Cards** of the Peppol participants on a voluntary basis meaning that  
42 SMP providers can (but are not forced to) publish their client's metadata in the PD. The data is stored  
43 in correlation with the SMP entry of the respective participant (aka service group). Details are  
44 described in chapter 5.2. SMP providers **MUST** provide the technical interface and **MAY** publish  
45 client's metadata.

---

46 This document describes the architecture of the PD server, the interfaces to and from it as well as the  
47 data format for the Business Cards (see chapter 5) within the SMP. This document concludes with a  
48 high-level technical description on how the PD Server is implemented.

### 49 **3 Why Peppol Directory? (non-normative)**

50 Due to variations between countries and markets, there are no shared models on how to know the  
51 Peppol Participant ID (PPID) of the sender, further enforced by the lack of open national business  
52 registries. Knowing each other in domains of limited size, for example e-CODEX project in e-Justice, is  
53 easy, however in domains like Peppol having potentially millions of organizations it is impossible.

54 Trying to solve the problem of finding each other, Peppol Directory (PD) is introduced, a central  
55 service to query based on given metadata. Querying may be part of a manual or automated process  
56 before performing lookup in SML (Service Metadata Locator) and SMP (Service Metadata Publisher).  
57 PD contains indexed Peppol Directory Business Cards (BC) containing metadata related to a given  
58 PPID. The lack of a Peppol Directory is a constraint to wider scale adoption of Peppol by small and  
59 medium sized enterprises.

#### 60 **3.1 Use Cases**

61 The Peppol Directory is intended to support business cases that are concerned with finding Peppol  
62 participants registered on the Peppol network, in order to start exchanging business documents with  
63 them. Some of the possible business cases are identified below.

##### 64 **3.1.1 New Peppol BIS support - Matching**

65 An organization that has recently become a Peppol participant, wishing to exchange a particular  
66 Peppol BIS, as a Customer or a Supplier, will want to find who of their trading partners are capable of  
67 exchanging the same BIS documents in the opposing role.

68 As example an organization that is starting to send invoices may want to know which of their  
69 customers can receive them and an organization that is starting to receive invoices will want to know  
70 which of their suppliers can send them.

##### 71 **3.1.2 Monitoring new Peppol users - Alerting**

72 An organization that is using Peppol to exchange one or more Peppol BIS may want to monitor when  
73 more of their trading partners become Peppol participants and consequently to automate their  
74 trading relations with them by using Peppol.

#### 75 **3.2 Planned key functions of Peppol Directory**

76 The following key functions are planned for the Peppol Directory and will be implemented through  
77 different releases of the Directory. These features are intended to support the business use cases  
78 described in the previous chapter.

### 79 **3.2.1 Free text search**

80 A free text search allows the Directory user to enter a text string into an online form and get a list of  
81 result for all listings in the Directory where that string appears. As an example, if the user enters the  
82 word "Acme" he will get a list of all participants whose name contains the word "Acme" as well as  
83 participants where the word "Acme" appears in other elements of the Business Card.

84 The user can browse the list to find the Peppol participant he is looking for and then click on his  
85 choice to see the full details.

### 86 **3.2.2 Identifier search**

87 The Directory supports the use of qualified identifiers for the search. The objective is to enable single  
88 match searches where the user submits a query on whether there exists a user, with a particular  
89 identifier and BIS capabilities. This enables searching by VAT, legal identifiers and other parameters  
90 that are commonly known but may differ from the Peppol end point identifiers. As an example, a  
91 user may want to find the end point identifier for a customer who has a particular VAT identifier. By  
92 restricting the search to a particular capability, he can use the query to monitor when that customer  
93 starts to support the given documents.

### 94 **3.2.3 API connection**

95 The Peppol Directory will also enable Directory users to let their systems connect automatically  
96 instead of manually browsing through a web interface. This supports automated searches that can be  
97 integrated into the sending process.

98 A drawback to be considered is that the publication of the Business Cards in the Peppol Directory  
99 happens on a voluntary basis.

## 100 **3.3 Considerations**

101 The following considerations influencing the development of the Peppol Directory have been  
102 identified but require additional analysis.

### 103 **3.3.1 Searching for senders**

104 The current architecture of the Peppol network does not require Peppol Participants who are only  
105 sending documents to be registered in the SMP's and consequently they are not in the SML. This  
106 limits the capability of the Peppol Directory to include these Peppol participants in search results.  
107 This relates to other issues that are currently being addressed in other Peppol initiatives. A potential  
108 change in the Peppol policy that requires registration of senders would benefit the Peppol Directory  
109 without requiring additional changes to the PD.

110 Alternatively sending only participants may be registered to an SMP with an empty service group  
111 which allows them to publish Business Cards for the Peppol Directory as well.

## 112 **3.4 Usage outside of Peppol**

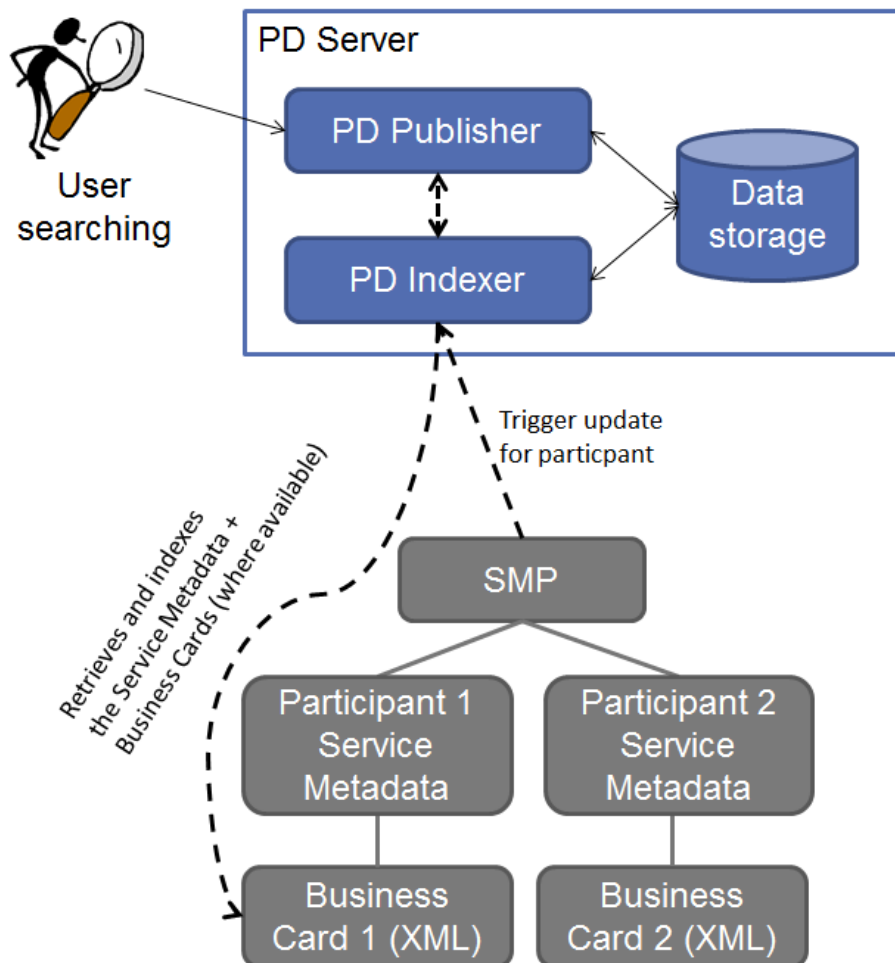
113 This specification and the software components were originally created for the usage within Peppol.  
114 As other projects also showed interest in reusing these artefacts it can be clearly stated, that the  
115 components described herein can be reused in different scenarios unrelated to Peppol. E.g. the

---

116 TOOP project ([www.toop.eu](http://www.toop.eu)) uses Peppol Directory as “TOOP Directory” inside their dynamic  
 117 discovery component to find multiple receivers using the REST query API.

#### 118 4 PD Server architecture (non-normative)

119 This section describes the overall architecture of the PD Server. It logically consists of two major  
 120 parts: a *PD Indexer* which is responsible for creating, updating, deleting and indexing the Business  
 121 Card data and the *PD Publisher* which is the public web frontend to the PD for both humans and  
 122 machines.



123

124

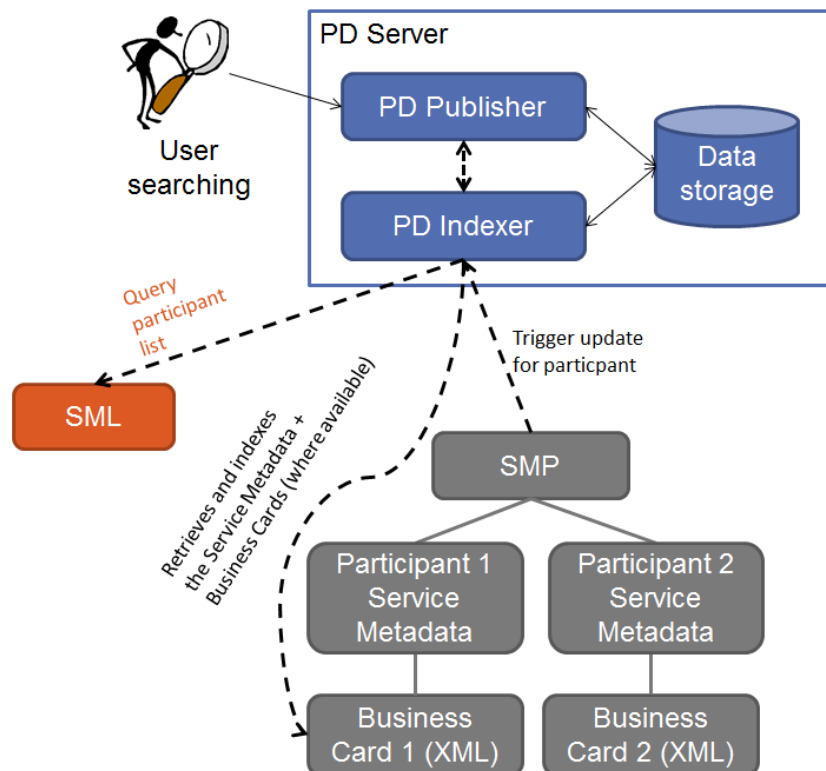
Figure 1: PD big picture without SML

125 The above big picture outlines the information flow. If a participant’s business card is added to,  
 126 updated to or deleted from an SMP, the SMP SHOULD trigger an update to the *PD Indexer* (see arrow  
 127 from SMP to the *PD Indexer* in the figure) even if the Business Card contained in the SMP is empty. If  
 128 data is to be added or updated on the PD, the *PD Indexer* will retrieve the complete Business Card  
 129 from the respective SMP and index it for searchability (see arrow from *PD Indexer* to *Business card*  
 130 in the figure).



131 If a user wants to know whether a certain company is registered in the Peppol network he opens the  
 132 web site of the *PD Publisher*, types the search term (e.g. the company name) and a list of potential  
 133 hits (including the Peppol participant identifier and the supported Peppol document types) shows up.  
 134 In addition to the human interface, a REST interface for automatic searching is offered. The *PD*  
 135 *Publisher* retrieves all relevant information directly from the *PD Indexer* so that no interaction with  
 136 the concerned SMPs is necessary.

137 An extension to the *PD Indexer* is the direct connection to the SML to retrieve a list of **all** registered  
 138 Peppol participants. In this case the PD Indexer will query the SML regularly (e.g. once a week) for a  
 139 complete participant list and queries the respective SMPs independent of the SMP provided update  
 140 status.



141

142

Figure 2: PD big picture with SML

143 As shown in the previous figure the overall architecture is only extended to interconnect with the  
 144 SML and no other changes are necessary. The SML already offers an interface to retrieve a list of all  
 145 registered Peppol participants and is therefore prepared to be interconnected with the PD.

146 Early benchmarks on the SML test machine (being slower than the production machine) showed that  
 147 a list with 100.000 entries can be created in 16 seconds and 150.000 entries took 34 seconds. By  
 148 middle of 2018 approx. 112.000 entries were in the production database, by October 2020 the  
 149 number increased to approx. 248.000 entries.

## 150 5 Business card (normative)

### 151 5.1 Data format

152 This section describes the layout of the business card data that is stored in an SMP. Because the  
153 scope of a single Peppol participant within an SMP can be very broad, the data format must be  
154 capable of handling information for more than one business entity in a structured way. Sometimes a  
155 Peppol participant may even link to different entities in different countries.

156 Existing formats like vCard, xCard or the UBL 2.1 Party type were not considered because they are  
157 either not XML or too complex to interpret fully. Instead a new minimal XML-based format is created  
158 because Peppol participant identifiers are used very differently it was decided to use a very flexible  
159 scheme that can represent multiple business entities at once.

160 The format defines a single business card consisting of the following fields:

- 161 • Peppol participant ID
  - 162 ○ Description: Peppol participant identifier corresponding to a service group hosted on
  - 163 the same SMP. The constraints for participant identifiers are described in [PFUO14].
  - 164 ○ Multiplicity: 1..1 (mandatory)
- 165 • Peppol document type ID
  - 166 ○ Descriptions: all Peppol document type identifiers as indicated by the default SMP
  - 167 service group query. The constraints for document type identifiers are described in
  - 168 [PFUO14].
  - 169 ○ Multiplicity: 0..n (optional but potentially many)
- 170 • Business entity
  - 171 ○ Description: a business entity that can be reached via the provided Peppol
  - 172 participant ID (see details below)
  - 173 ○ Multiplicity: 0..n (optional but potentially many)

174 Each Business Entity consists of the following fields:

- 175 • Entity name
  - 176 ○ Description: the company name or the name of the governmental entity. The name
  - 177 MUST NOT be empty and MAY include an optional language per entity name. If the
  - 178 language is used it MUST be in ISO 639-1 format (e.g. “de” for German).
  - 179 ○ Multiplicity: 1..n (mandatory but potentially many)
- 180 • Country code
  - 181 ○ Description: the country code to which the entity belongs. The format MUST be ISO
  - 182 3166-2 (e.g. “AT” for Austria)
  - 183 ○ Multiplicity: 1..1 (mandatory)
- 184 • Geographic information
  - 185 ○ Description: describes the location or region of the entity that is usually used to
  - 186 identify the entity. This may be an address, a state name etc.

- 187 ○ Multiplicity: 0..1 (optional)
- 188 ● Identifier
  - 189 ○ Description: additional (non-Peppol) identifiers of the entity that are not part of the
  - 190 Peppol participant identifier. It consists of a type and a value. This can e.g. be a
  - 191 national VAT identification number; a national company register number etc. The
  - 192 following identifier types (case insensitive) must at least be supported by the
  - 193 Directory:
    - 194 ■ "vat" – VAT identification number including the national prefix
    - 195 ■ "orgnr" – the national organisation number
    - 196 ■ "gln" – Global Location Number (GLN)
    - 197 ■ "duns" – DUNS number
  - 198 ○ Multiplicity: 0..n (optional but potentially many)
- 199 ● Registration date
  - 200 ○ Description: the date when the participant joined the Peppol network
  - 201 ○ Multiplicity: 0..1 (optional)

202 The link to the XML Schemas describing the layout of the Business Card can be found in chapter 8 of  
 203 this document. To support future updates of this Business Card scheme the XML root element  
 204 (`BusinessCard`) has an XML namespace URI that allows for easy versioning of the contained data:

- 205 ● Version 1 of the XML schema for the business card uses the XML namespace URI
- 206 `http://www.peppol.eu/schema/pd/businesscard/20160112/`
- 207 ● Version 2 of the XML schema uses the XML namespace URI:
- 208 `http://www.peppol.eu/schema/pd/businesscard/20161123/`
- 209 ● Version 3 of the XML schema uses the XML namespace URI:
- 210 `http://www.peppol.eu/schema/pd/businesscard/20180621/`

211 A non-normative example Business Card with a single entity looks like this:

```

212 <BusinessCard
213   xmlns="http://www.peppol.eu/schema/pd/businesscard/20161123/">
214   <ParticipantIdentifier
215     scheme="iso6523-actorid-upis">0088:example</ParticipantIdentifier>
216   <BusinessEntity registrationDate="2010-07-06">
217     <Name>ACME Inc.</Name>
218     <CountryCode>AT</CountryCode>
219     <GeographicalInformation>ACME street 123</GeographicalInformation>
220     <Identifier scheme="VAT">ATU12345678</Identifier>
221     <Identifier scheme="OrgNr">hjd7as9ds</Identifier>
222   </BusinessEntity>
223 </BusinessCard>
  
```

## 224 5.2 SMP impacts

225 This chapter describes the constraints for storing Business Cards in an SMP and how to access the  
226 Business Cards from the outside world.<sup>2</sup>

### 227 5.2.1 Storage (non-normative)

228 This section describes how and where Business Cards are to be stored in an SMP (see [PEPPOLSMP]).  
229 The SMP differentiates between Service Groups and Service Registrations. A Service Group is  
230 basically the Peppol participant identifier whereas a Service Registration is the combination of a  
231 participant identifier, a document type identifier, a process identifier, a transport protocol and an AP  
232 endpoint URL (plus some additional information).

233 Each Business Card MUST be stored in relation to a single SMP Service Group. There are no  
234 predefined rules how this is to be achieved as the data storage mechanisms of an SMP server are  
235 quite different in practice. The only binding rules are:

- 236 1. An SMP MUST NOT provide Business Cards for service groups not owned by this SMP.
- 237 2. Each service group MUST have zero or one associated Business Card.
- 238 3. The link between the Service Group and the Business Card MUST be the Peppol participant  
239 ID.

240 Originally it was considered to store the Business Card information in the `Extension` element of an  
241 SMP Service Group. The positive aspects of this solution are that the data model of existing SMPs  
242 does not need to be altered and that no new APIs for the SMP need to be provided. The negative  
243 aspects of this solution are that the network traffic for non-PD queries would heavily increase and  
244 the general performance of SMPs might be downgraded and that non-relevant information would be  
245 returned in regular Service Group queries. An additional problem with this solution is that the Peppol  
246 SMP specification is lacking support for multiple extensions in a single service group which in turn  
247 would require an additional non-standard “extension container” to maintain extensibility. OASIS  
248 BDXR SMP adds supported for multiple extensions.

### 249 5.2.2 Public REST interface

250 To retrieve the Business Cards from an SMP server a new REST interface is introduced. This interface  
251 MUST be provided by all Peppol SMP servers. REST was chosen because the existing SMP interfaces  
252 are already REST based and therefore no new technology is introduced.

#### 253 5.2.2.1 Retrieve Business Card interface

254 REST request: `GET /businesscard/{participantID}`

255 Note: `{participantID}` is the placeholder for the effective Peppol participant identifier in the  
256 URL encoded form

---

<sup>2</sup> phoss SMP and IBM SMP have already implemented support for the BusinessCard API in their solutions.

257 REST response: the XML representation of the Business Card (according to an XSD specified in  
258 chapter 8) preferably in UTF-8 encoding using MIME type `application/xml`.

259 REST response code:

- 260 ● HTTP 200 (OK) – everything was ok. A response body MUST be send back.
- 261 ● HTTP 404 (Not found) – no Business Card was found for the provided participant ID.
- 262 ● HTTP 500 (Internal server error) – something internally went wrong. Response body SHOULD  
263 contain the details in plain text.

264 Non-normative example to query the business card for Peppol participant `9915:test` on the SMP  
265 server running at `http://smp.example.org`:

```
266 http://smp.example.org/businesscard/iso6523-actorid-upis%3A%3A9915%3Atest
```

267 The response may look like the example provided in section 5.1.

268 Note: using Peppol participant identifiers directly in URLs may impose problems. It must be ensured  
269 that appropriate URL escaping is performed (e.g. `:` replaced with `%3A`).

270

271 Note: this interface must also work with the computed “B-....edelivery.tech.ec.europa.eu” URLs.

272

273 Note: as a future extension, the response of the SMP may be signed with the respective SMP  
274 certificate.

## 275 6 PD Indexer (normative)

276 This chapter describes the technical details of the *PD Indexer*. It describes the data elements that  
277 must be passed to the *PD Indexer* so that Business Cards can be created, updated, deleted or  
278 retrieved. This is a REST interface, because the SMP server (that will trigger this interface) is also a  
279 REST server and therefore the technology is well known and supported.

280 All REST interface URLs contain a version number so that it will be easy to provide updated interfaces  
281 in the future without breaking the existing ones.

### 282 6.1 Authentication and authorization

283 Note: this section is only applicable, if the *PD Indexer* runs on a server that offers secure HTTP  
284 connections (https via TLS).

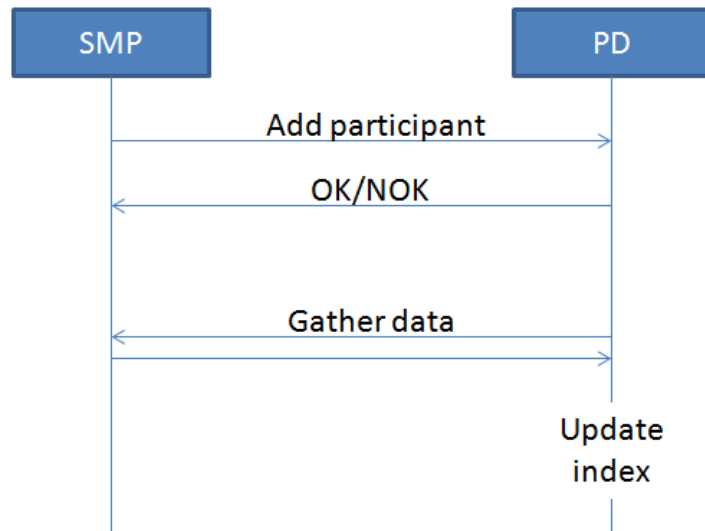
285 For security reasons, only legitimate SMPs are allowed to request modifications in the *PD Indexer*. To  
286 ensure this *all* HTTP calls to the *PD Indexer* interface must provide a client X.509 certificate. This is  
287 the same technology that is already used in the SMP to SML communication and should therefore be  
288 implementable in a quick and easy way. Requests to the *PD Indexer* without a client certificate will  
289 result in an error.

---

290 The provided client certificate must be the SMP certificate as used for the communication with the  
 291 SML.

## 292 6.2 Adding a participant

293 For adding a participant, only the participant identifier must be passed to the *PD Indexer*. The  
 294 Business Card is read directly from the respective SMP (determined via DNS lookup) and is not  
 295 passed in this call. This allows the *PD Indexer* to build a queue of items to be updated in an optimized  
 296 way, and also avoids overwriting data of participants that are owned by different SMPs.



297

298 **Figure 3: Add participant workflow**

299 REST request: `PUT /indexer/1.0/`

300 Request body: `{participantID}`

301 **Note:** {participantID} is the placeholder for the effective Peppol participant identifier

302 Example request:

- 303 • URL: `PUT /indexer/1.0/`
- 304 • Body: `iso6523-actorid-upis::0088:gln1234`

305 The participant identifier **MUST NOT** be URL encoded.

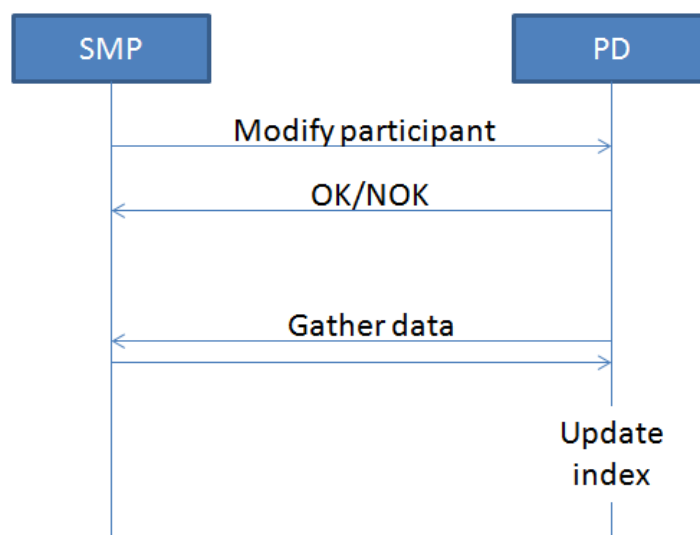
306 REST response code:

- 307 • HTTP 204 (OK, No content) – everything was ok. No response body is send back.
- 308 • HTTP 403 (Forbidden) – no client certificate or an invalid client certificate provided
- 309 • HTTP 500 (Internal server error) – something internally went wrong. Response body **SHOULD**
- 310 contain the details in plain text.

311 Note: This requires the DNS entry of the added Peppol participant already being available publicly to  
 312 resolve the owning SMP. Therefore an SMP MUST call the PD after the registration at the SML. The  
 313 *PD Indexer* will handle added participants gracefully if the respective DNS entry is not yet present  
 314 and will retry at a later point in time. If a new participant DNS entry is not present within a  
 315 configurable duration related to the original indexing request, this particular request is discarded  
 316 and therefore no indexing takes place. If previous indexed information of that participant is  
 317 present (if it is an updating call) they are left unchanged.

### 318 6.3 Modifying an existing participant

319 If the Business Card of an existing participant is modified the *PD Indexer* must be informed about the  
 320 change. The API and the constraints are identical to “Adding a participant” (see chapter 6.2).



321

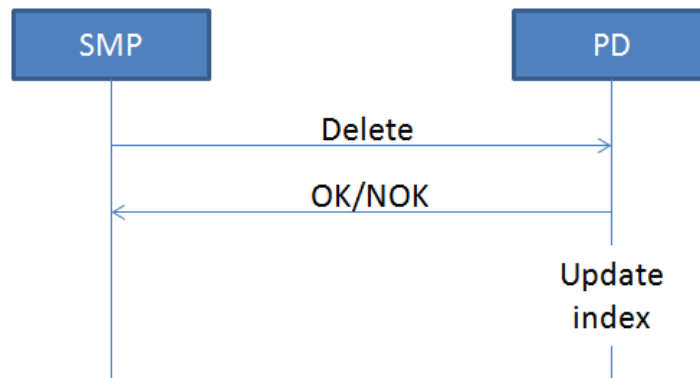
322

Figure 4: Modify participant workflow

323 Note: there is no possibility to identify whether the participant was added or updated by the  
 324 response. To check for existence, use the GET operation defined below.

### 325 6.4 Deletion of a participant

326 When a service group in the SMP is about to be deleted (either because the participant leaves the  
 327 Peppol network or because an SMP migration takes place), the *PD Indexer* must be notified. To  
 328 delete participant information in the *PD Indexer* it is suitable to provide only the respective Peppol  
 329 identifier.



330

331

Figure 5: Delete participant workflow

332 REST request: `DELETE /indexer/1.0/{participantID}`

333 Note: {participantID} is the placeholder for the effective Peppol participant identifier in URL encoded  
334 form

335 Example request:

336 • `DELETE /indexer/1.0/iso6523-actorid-upis%3A%3A0088%3AglN1234`

337 Note: using Peppol participant identifiers directly in URLs may impose problems. It must be ensured  
338 that appropriate URL escaping is performed (e.g. `:` replaced with `%3A`).

339 REST response code:

- 340 • HTTP 204 (OK, No content) – everything was ok. No response body is send back.
- 341 • HTTP 403 (Forbidden) – no client certificate or an invalid client certificate provided
- 342 • HTTP 500 (Internal server error) – something internally went wrong. Response body SHOULD  
343 contain the details in plain text.

344 Note: if a participant is moved (migrated) from one SMP to another it must first be deleted by the old  
345 SMP and then re-created by the new SMP.

346

347 Note: the delete operation may impose a security problem because one SMP can delete the  
348 information of a participant created by a different SMP. Therefore the deletion does not directly  
349 delete the information in the index but only marks the respective records internally as “deleted”  
350 so that the data can be restored in case of a misuse.

## 351 6.5 Existence check of a participant

352 Checking whether a Business Card of a participant is present in the *PD Indexer* can be performed via  
353 the following interface:

354 REST request: `GET /indexer/1.0/{participantID}`



355 Note: {participantID} is the placeholder for the effective Peppol participant identifier in URL encoded  
356 form

357 Example request:

358 • `GET /indexer/1.0/iso6523-actorid-upis%3A%3A0088%3Agln1234`

359 Note: using Peppol participant identifiers directly in URLs may impose problems. It must be ensured  
360 that appropriate URL escaping is performed (e.g. `:` replaced with `%3A`).

361 REST response code:

- 362 • HTTP 204 (OK, No content) – Yes, the participant is already in the *PD Indexer*.
- 363 • HTTP 403 (Forbidden) – no client certificate or an invalid client certificate provided
- 364 • HTTP 404 (Not found) – the participant is **not** in the *PD Indexer*.
- 365 • HTTP 500 (Internal server error) – something internally went wrong. Response body SHOULD  
366 contain the details in plain text.

367 Note: because of the internal asynchronous processing, it might take some time after an index  
368 request until the participant is available in search results. See chapter 6.7 for more details.

369

370 Note: calling this API also requires the presence of a client certificate.

## 371 6.6 Auditing and Logging

372 All successful calls to the *PD Indexer* (create/update/delete/get) are logged together with the  
373 timestamp, the source IP address and some information from the provided certificate (country,  
374 subject name and serial number) to ensure traceability of the performed actions.

## 375 6.7 Internal processing of the data (non-normative)

376 Internally the Indexer keeps a FIFO work queue that is processed asynchronously. All new indexing  
377 requests (create/update/delete) are put into that queue and wait for their serial processing to avoid  
378 overloading a single SMP with queries. That's why deletion (see chapter 6.4) may not trigger an  
379 immediate return code like "not found" because the result is not known synchronously.

380 If the data retrieval from the SMP fails (for whatever reason) the work item is put into a special "retry  
381 queue" and the data retrieval is retried sometime later (suggested duration until retry is 5 minutes –  
382 must be configurable). If an entry cannot be indexed after a certain period of time (suggested period  
383 is 24 hours – must also be configurable), it is moved to a "dead work item queue". In case of a  
384 permanent failure manual intervention is necessary. E.g. the PD administrator may re-trigger the  
385 work item manually or choose to drop it completely.

386 The asynchronous processing may impose problems when trying to check for the existence of a  
387 certain participant identifier in the index. This check will only return success if the item was already  
388 processed and stored in the index but not if it is still in the work queue.

---

## 389 6.8 Internal data structure (non-normative)

390 The internal data structure of the *PD Indexer* is slightly different from the Business Card entities  
391 defined in chapter 5.1. Besides the Business Card content, the following data elements should also be  
392 stored:

- 393 • All supported Peppol document type identifiers as listed by the SMP service group interface.  
394 Therefore, a separate SMP query on the ServiceGroup must be performed and the document  
395 types must be extracted.
- 396 • The unique identifier taken from the client certificate that triggered the indexing of the  
397 document (the “requestor”). This can e.g. consist of the certificates subject name, serial  
398 number and country code.
- 399 • The date and time when the Business Card was last indexed.

## 400 7 PD Publisher (normative)

401 This section describes the components of the *PD Publisher*. It consists of a machine-to-machine  
402 search interface as well as a search interface for humans as well as a list of registered Peppol  
403 participants for download. Additional features can be integrated into the Publisher after the initial  
404 version.

405 Currently two implementations are available:

406 <https://directory.peppol.eu> (production server)

407 <https://test-directory.peppol.eu> (test server)

### 408 7.1 Search interface

409 This section only describes the machine-to-machine search interface. It uses REST as the protocol and  
410 responds with XML or JSON data.

#### 411 7.1.1 Request

412 The relative base URL of the REST search service is `/search/1.0/[format]` which is then  
413 followed by a list of query parameters as outlined below. The `[format]` placeholder in the request  
414 API denotes the desired response format. Initially `xml` (for XML output) and `json` (for JSON output)  
415 are supported but other formats might be added as future extensions. All search REST requests are  
416 HTTP GET requests. Other HTTP methods like POST, PUT etc. are not supported.

417 The search routines use the following text matching algorithms:

- 418 • *Exact match (case sensitive)*: the search term and the indexed values must be completely  
419 equal, including case sensitivity.
- 420 • *Exact match (case insensitive)*: the search term and the indexed values must be completely  
421 equal, excluding case sensitivity.

- 422
- 423
- 424
- 425
- 426
- 427
- *Partial match*: the search term must be equal or fully contained in the indexed value in a case insensitive way (e.g. searching for “tici” or “TICI” in the indexed value “participant” will be a match)
  - *Starts with match*: a special version of the partial match that requires the indexed value to begin with the search term in a case insensitive way (e.g. search for “part” or “PART” will match “participant” but “art” won’t match “participant”)

Parameter name	Explanation
<b>q</b>	General purpose query term. This term is searched in all fields with the matching rules of the respective fields. Multiple search terms can be provided separated by a whitespace character. If multiple search terms are provided, they are interpreted as “AND” operators, so only results with all query terms are returned.
<b>participant</b>	Searches for <i>exact matches (case insensitive)</i> in the <i>participant identifier</i> field (the identifier scheme must be part of the value).
<b>name</b>	Searches for <i>partial matches</i> in the <i>entity name</i> field. Only search terms consisting of at least 3 characters are used for search. This parameter can occur more than once. Tokens are not split when using this parameter.
<b>country</b>	Searches for <i>exact matches (case insensitive)</i> in the <i>country code</i> field. This parameter can theoretically occur more than once but it does not make sense, because a business card cannot have more than one country.
<b>geoinfo</b>	Searches for <i>partial matches</i> in the <i>geographic information</i> field. Only search terms consisting of at least 3 characters are used for search. This parameter can occur more than once. Tokens are not split when using this parameter.
<b>identifierScheme</b>	Searches for <i>exact matches (case insensitive)</i> in the <i>additional identifier type</i> field (only the type, not the value). Tokens are not split when using this parameter. Combine this field with <b>identifierValue</b> field for fine-grained searching.
<b>identifierValue</b>	Searches for <i>exact matches (case insensitive)</i> in the <i>additional identifier value</i> field (only the value, not the type). Tokens are not split when using this parameter. Combine this field with <b>identifierScheme</b> field for fine-grained searching.
<b>website</b>	Searches for <i>partial matches</i> in the <i>website</i> field. Only search terms consisting of at least 3 characters are used for search. This parameter can occur more than once. Tokens are not split when using this parameter.
<b>contact</b>	Searches for <i>partial matches</i> in the <i>contact</i> fields (type, name, phone number and email address). Only search terms consisting of at least 3 characters are used for search. This parameter can occur more than once. Tokens are not split when using this parameter.
<b>addinfo</b>	Searches for <i>partial matches</i> in the <i>additional information</i> field. Only search terms consisting of at least 3 characters are used for search.

	This parameter can occur more than once. Tokens are not split when using this parameter.
<b>regdate</b>	Searches for exact matches in the <i>registration date</i> field. The value of the date to search must be provided in the format 'YYYY-MM-DD' (ISO 8601/XML Schema based date format). The parameter supports neither a time nor a time zone.
<b>doctype</b>	Searches for <i>exact matches (case sensitive)</i> in the <i>document type identifier</i> field (the identifier scheme must be part of the value).

428 If multiple of the query parameters are used together only the results matching ALL query terms are  
429 returned (like a boolean AND operation).

430 The following table contains the additional parameters that can be used to control the result subset:

Parameter name	Explanation
<b>resultPageIndex</b>	The result page to be shown. If this parameter is not present the first page is returned. The result page index is 0-based meaning that the first page has index 0. The index of the first search result returned is calculated by $resultPageIndex * resultPageCount$
<b>resultPageCount</b>	The number of results to be returned on a single page. If this parameter is not present 20 results are returned by default.
<b>beautify</b>	Format the results so that they are more human readable? This should only be used for debugging purposes as it increases the transferred data volume. By default the returned code is minified.

431

### 432 7.1.2 Response

433 If no query term parameter (see table in chapter 7.1.1) is provided the return value is HTTP 400 (Bad  
434 Request).

435 In addition to the result Business Cards, each response contains the following fields (in a syntax-  
436 specific way):

Field name	Explanation
<b>version</b>	The version of the response layout, defining the contained fields. This is only present to handle future modifications. The current version is <b>1.0</b> .
<b>total-result-count</b>	The total number of matching documents.
<b>used-result-count</b>	The number of results contained in the response. This is always $\leq$ <b>total-result-count</b> as it is based on the paging parameters used.
<b>result-page-index</b>	The 0-based index of the result page.
<b>result-page-count</b>	The number of entities to show on a single page.
<b>first-result-index</b>	The effective 0-based index of the first result item returned (inclusive). This is the result of $result-page-index * result-page-count$ .
<b>last-result-index</b>	The effective 0-based index of the last result item returned (inclusive). This is the result of $min((result-page-index + 1) * result-page-count - 1, total-result-count - 1)$ .

<b>query-terms</b>	The combined query string that was used to perform a search. This is mainly for debugging purposes to cross-check which parameters took effect in searching.
<b>creation-dt</b>	The UTC date and time when this response was created. If possible it is formatted according to XML Schema (XSD) rules.

437

438 Note: the PD Publisher will deliver at most the top 1.000 results. If the combination of  
 439 `resultPageIndex` and `resultPageCount` results in too small (< 0) or too large values (>  
 440 1000) the return value is HTTP 400 (Bad Request). The index of the first search result returned is  
 441 `resultPageIndex * resultPageCount`. The index of the last search result returned is  
 442 `(resultPageIndex + 1) * resultPageCount - 1`.

## 443 7.2 User interface (non-normative)

### 444 7.2.1 Use case Search

445 The PD Publisher must offer a publicly available web page where the user can enter search terms to  
 446 search for one or more Peppol participants. It should provide a simple search form where only a set  
 447 of terms can be entered and the *PD Publisher* will search for the best possible matches. Additionally,  
 448 an extended search form with all fields (as outlined in chapter 7.1.1) should be available.

449 The search results will be shown on the website and will also be made available for download.

### 450 7.2.2 Use case Browse

451 The *PD Publisher* should offer a list of all registered business entities so that the information is  
 452 browsable or even downloadable as e.g. an Excel document. This implies that the full data must be  
 453 stored in the *PD Indexer*.

## 454 8 Annex A - Business Card XSD

455 The Peppol Directory implementation supports multiple Business Card formats. The official Business  
 456 Card XML Schemas can be found in the following folder:

457 [https://github.com/phax/phoss-directory/tree/master/phoss-directory-](https://github.com/phax/phoss-directory/tree/master/phoss-directory-businesscard/src/main/resources/schemas)  
 458 [businesscard/src/main/resources/schemas](https://github.com/phax/phoss-directory/tree/master/phoss-directory-businesscard/src/main/resources/schemas)

459 The details in the versions are as follows:

- 460 • `peppol-directory-business-card-20160112.xsd`
  - 461 ○ This is the original proposal of the data format
  - 462 ○ It contains a single entity name without a language
- 463 • `peppol-directory-business-card-20161123.xsd`
  - 464 ○ This is the official data format version 1
  - 465 ○ Compared to 20160112 it does not contain the fields “AdditionalInformation”,
  - 466 “Contact” and “WebsiteURL” but for the rest it is identical.
  - 467 ○ It contains a single entity name without a language

- 468 • peppol-directory-business-card-20180621.xsd
- 469 ○ An extension to the 20160112 format
- 470 ○ It allows to specify more than one name in different languages
- 471 ○ This is the **preferred version** to implement

## 472 9 Annex B - Implementation proposal (non-normative)

473 This section roughly describes, how the *PD Server* could be implemented and how existing SMP  
474 servers could be modified to interact with the PD server.

475 All data described in this document must be stored and/or transmitted in UTF-8 character encoding  
476 set. Using other character encodings is prohibited.

477 The rest of this chapter assumes that the development is done with Java.

### 478 9.1 PD Server

479 For simplicity the *PD Server* should be implemented as a regular Java web application that is runnable  
480 on a regular servlet container like Apache Tomcat or Jetty. It internally consists of two main parts:  
481 the *PD Indexer* and the *PD Publisher*. Both components have to expose a component to the outside  
482 world but need to fulfil different tasks.

#### 483 9.1.1 PD Indexer

484 The *PD Indexer* is responsible for gathering the business cards from the different SMPs and storing it  
485 into a searchable index. It is also responsible for periodically grabbing all participants from the SML.

486 The basic components are:

- 487 • A “work queue” that handles the requested actions for certain participants with a certain  
488 priority handling (requests from SMPs have a higher priority than SML crawling results). The  
489 work queue must be able to filter out duplicate requests and leave only the ones with the  
490 highest priority.
- 491 • A “fetcher” that grabs action items from the work queue and queries the SMP for the  
492 corresponding data of a participant
- 493 • An “indexer” that takes the fetch results and stores them into a searchable index
- 494 • A scheduled “SML retriever” that retrieves the participant list from the SML and stores all  
495 entries for updating in the work queue.
- 496 • A REST server implementing the interfaces as defined in chapter 6 and accordingly filling the  
497 work queue. Only HTTP requests providing a valid Peppol SMP client certificate are accepted.
- 498 • An “auditor” that keeps track of all indexing actions together with some meta information

499 The *PD Indexer* is based on Apache Lucene (<https://lucene.apache.org/core/> - Apache 2 License) for  
500 the indexing. The REST interface is to be done with Jersey (<https://jersey.java.net/> - CDDL 1.1 or GPL  
501 2 with Classpath exception) like with the SMP.

### 502 **9.1.2 PD Publisher**

503 A simple *PD Publisher* can be built with the ph-oton library (<https://github.com/phax/ph-oton> -  
504 Apache 2 License) which offers capabilities to create state of the art (responsive, fast, nice looking)  
505 web applications quickly. For the main searching Apache Lucene will be used (must be identical to  
506 the version used for indexing).

507 The basic components of the *PD Publisher* are:

- 508 • A REST based search interface as described in chapter 7.1
- 509 • A public web page for the simple search
- 510 • A public web page for the extended search
- 511 • A public web page with the most recently added participants
- 512 • A secure web site to see the log and audit entries

## 513 **9.2 SMP-PD interface**

514 The PD software suite should ship with a library that can be used to trigger the indexing in the *PD*  
515 *Indexer*. SMP software providers can use this library to simplify the process of integrating their  
516 software with the PD as they just need to call this when relevant information changes (new  
517 participant, Business Card update, participant deletion).

518 The Open Source phoss SMP (<https://github.com/phax/phoss-smp>) and other commercial SMP  
519 implementations already support the Business Card API.

520 The CEF SMP server (<https://ec.europa.eu/cefdigital/wiki/display/CEFDIGITAL/SMP>) cannot be used  
521 for Peppol because it only supports the OASIS BDXR SMP 1.0 interface and it does not support the  
522 required Business Card interface.