

## SDMX GUIDELINES

# **GUIDELINES FOR THE DESIGN OF DATA STRUCTURE DEFINITIONS**

VERSION 1.0

10 June 2013



## Contents

<b>1</b>	<b>Aim of this document</b> .....	<b>1</b>
<b>2</b>	<b>General Design Principles</b> .....	<b>2</b>
2.1	Reuse of existing DSDs and code lists .....	2
2.1.1	Identify existing DSDs and code lists .....	3
2.1.2	Priority ranking of existing DSDs and code lists .....	3
2.1.3	Suitability of available DSDs and code lists.....	4
2.2	Flexibility and future needs .....	7
2.3	Structural principles .....	7
2.3.1	Parsimony.....	7
2.3.2	Simplicity.....	7
2.3.3	Purity .....	8
2.3.4	Density and sparseness.....	8
2.3.5	Unambiguousness .....	8
2.3.6	Exhaustiveness.....	9
2.3.7	Orthogonality .....	9
2.3.8	User-friendliness .....	9
2.3.9	Fitness for use throughout the statistical business process.....	10
<b>3</b>	<b>Usage contexts</b> .....	<b>10</b>
3.1	Type of data .....	10
3.2	Domain.....	10
3.3	Purpose.....	10
3.4	Type of data exchange and recipient.....	12
3.5	Role in data exchange.....	12
3.6	Process pattern .....	12
3.7	Phase in statistical business process.....	12

<b>4</b>	<b>Data structuring approaches .....</b>	<b>13</b>
4.1	Number and content of concepts .....	14
4.2	Number and relations of DSDs .....	18
<b>5</b>	<b>Minimum structural and semantic requirements .....</b>	<b>21</b>
5.1	Required and recommended dimensions and attributes .....	22
5.2	Attribute attachment levels and definition of groups .....	23
5.3	Header elements .....	24
<b>6</b>	<b>Step-by-step guide .....</b>	<b>25</b>
6.1	High-level overview of the process .....	25
6.2	Defining modified DSDs.....	28
6.3	Defining new DSDs .....	29
6.4	Checklist for DSD Designers .....	34
<b>7</b>	<b>Annex 1. Glossary of Terms .....</b>	<b>36</b>
<b>8</b>	<b>Annex 2. What is a DSD? .....</b>	<b>36</b>
<b>9</b>	<b>Annex 3. References .....</b>	<b>37</b>
9.1	SDMX Documents.....	37
9.1.1	Existing documents.....	37
9.1.2	Not yet available documents .....	38
9.2	Non-SDMX Documents .....	38

## 1 AIM OF THIS DOCUMENT

The development of global Data Structure Definitions (DSDs) by the SDMX consortium and similar efforts by individual SDMX sponsor organizations and other international organizations to enable the broader adoption of the SDMX standard for data collection, exchange, and dissemination raise a need for standards or at least common guiding principles and recommendations. This document provides such guidelines based on conceptual considerations and first hand experiences with global DSD development. It does so by taking into account the specific requirements of different usage contexts. For example, DSDs may target:

- different types of data, e.g., micro data and macro data (cross-sectional and time series);
- different data exchange scenarios such as exchange at the national level, collection of international organizations from national member organizations, exchange between international organizations, dissemination to the general public; and/or
- different types of intended recipients, for instance in machine-to-machine or machine-to-user communication.

Different approaches to structuring data serve the varying needs of these different usage contexts to different extents. Therefore, this document presents different data structuring approaches and discusses their pros and cons in different situations instead of prescribing “the best” one-size-fits-all approach. It concentrates on the exchange of macro data; micro data are not covered.

Target audiences for these guidelines include domain experts and official statisticians involved in DSD development. Thus focusing on the business/content side of DSD development, the document tries to avoid technical jargon when explaining underlying concepts and ideas, but tries to still be useful for IT experts supporting SDMX implementations. Ideally, the document can bridge the gap between IT and statistical experts. The scope of the guidelines is restricted to conceptual aspects. Organizational and technical aspects are treated in separate documents.

- Code lists are the crucial building blocks of data structure definitions. Especially in the case of SDMX recommended code lists (particularly for cross-domain concepts; see SDMX Content Oriented Guidelines under “Guidelines” at <http://sdmx.org/>), list development and maintenance as well as DSD development and maintenance are carried out by different organizations at different points in time. For example SDMX recommended code lists for frequency and observation status already exist and should be used by reference in DSDs. While “SDMX” is responsible for the maintenance of these code lists, the DSD developing organization will be responsible for the maintenance of the DSD, that is, for the structure at a higher level. (Of course, a global DSD may also have “SDMX” as maintenance agency.) In any case, there is a strong interrelationship between DSD and code list development and maintenance (see SDMX Guidelines for the creation and maintenance of code lists under “Guidelines” at <http://sdmx.org/>).
- Maintenance and governance rules for DSDs including issues of updating, versioning, retiring as well as questions of responsibilities, especially relevant in the context of global DSDs jointly developed by multiple organizations and maintained by “SDMX” (or multiple organizations), will be covered by separate guidelines (see “Guidelines” at <http://sdmx.org/>).
- Issues related to SDMX registries (in general, and the global SDMX registry in particular) such as storage, federation, and registration of, as well as search for,

49 retrieval and download of, code lists and DSDs are not in the scope of this document.  
50 For more information on the registry see the “Standards” page at <http://sdmx.org/>.  
51 - Guidelines for the development, maintenance, and governance of metadata structure  
52 definitions (MSDs) will be made available separately under “Guidelines” at  
53 <http://sdmx.org/>.  
54 - Documentation on more IT-related issues is available at the SDMX IT tools and  
55 SDMX tutorials site at [http://sdmx.org/?page\\_id=13](http://sdmx.org/?page_id=13). The SDMX Tools Repository can  
56 be accessed at <http://www.sdmxtools.org/>. Many of the SDMX tools listed and  
57 described there are available free of charge.

58 This document is structured as follows. Section 2 outlines general design principles of DSDs.  
59 Section 3 discusses different usage contexts of DSDs in more detail. Section 4 gives an  
60 overview of different data structuring approaches including benefits, drawbacks, and context-  
61 specific recommendations. General minimum structural and semantic requirements are  
62 discussed in section 5. Section 6 provides a step-by-step guide to designing DSDs including  
63 a checklist for DSD designers. The three annexes include a glossary in Annex 1, a definition  
64 and brief introduction of the core components of a DSD in Annex 2, and a list of references  
65 in Annex 3.

## 66 **2 GENERAL DESIGN PRINCIPLES**

67 Besides the evident requirement of *standard compliance*, a couple of general design  
68 principles apply to SDMX DSD development independently of the domain and the particular  
69 usage context the DSD is embedded in. Examples include *flexibility in changing*  
70 *requirements*; *stability*; *usage of existing* code lists or even DSDs; and *parsimony*, *simplicity*,  
71 *unambiguousness*, and *density* of the dimensional model. Please note that the SDMX-ML  
72 Standards do not impose an order on concepts (i.e., dimensions and attributes). Strictly  
73 speaking, standard compliance of a DSD only entails technical compliance with the SDMX  
74 technical standard. However, *adherence to SDMX* content recommendations, principles, and  
75 best practices as provided in the *SDMX Content-Oriented Guidelines* (see  
76 [http://sdmx.org/?page\\_id=11](http://sdmx.org/?page_id=11)) is strongly recommended. It should be kept in mind that one  
77 major aim of SDMX is to have transparency and agreement on the meaning of statistical  
78 concepts in order to allow their flawless communication.

### 79 **2.1 Reuse of existing DSDs and code lists**

80 Whenever a DSD is required to exchange data according to the SDMX standard, the *reuse*  
81 *of existing DSDs and code lists* should be the first guiding principle. As far as possible, this  
82 reuse should be accomplished by referring to the existing artefacts, not by creating  
83 independent copies. What needs to be considered, though, is the handling of updates of the  
84 reused DSD or code lists in the new DSD (or data flow or data provision agreement). This  
85 heavily depends on the guidelines for the maintenance of code lists (provided as a separate  
86 document) and to what extent the maintenance agency follows these guidelines.

87 In case of artefacts with maintenance agency “SDMX” or one of the sponsor organizations,  
88 reasonable versioning of artefacts and availability of old versions can be expected, as these  
89 organizations have a genuine interest in fostering the usage, and thus maintenance, of the  
90 SDMX standard and its artefacts. This means that by referring to a certain version of a code  
91 list or DSD, the new structure will not change automatically when a new version of the code  
92 list or DSD that was included by reference becomes available. Rather, re-users of the (now  
93 modified) code list or DSD have full control whether they want to modify their artefacts by  
94 pointing to the new version. In case of more local maintenance agencies (with potentially  
95 less compliance to the SDMX Content-Oriented and other Guidelines) it may make sense to  
96 maintain a separate copy of the artefacts to be reused, this way circumventing issues with

97 less dependable sources. As stated in the introductory section, questions of the  
98 maintenance of DSDs and notification mechanisms for SDMX artefacts are discussed in  
99 separate documents.

### 100 **2.1.1 Identify existing DSDs and code lists**

101 The Global SDMX Registry (currently under development) is the primary location to search  
102 for global SDMX artefacts, especially DSDs, MSDs, SDMX cross-domain concepts and code  
103 lists, and domain specific concept schemes and code lists used by global DSDs. It includes  
104 artefacts with maintenance agency “SDMX” as well as artefacts maintained by sponsor  
105 organizations. Usage of the Global SDMX Registry is explained in a separate document. In  
106 addition, sponsor organizations, other international and national organizations may have  
107 their own SDMX registries or other ways of distributing their code lists, DSDs, and MSDs on  
108 their websites.

### 109 **2.1.2 Priority ranking of existing DSDs and code lists**

110 Regarding reuse of existing DSDs, global DSDs with “SDMX” or SDMX sponsor  
111 organization(s) as maintenance agency have priority. If a suitable global DSD does not exist,  
112 the usage of other already available DSDs is to be considered. For example, in case of the  
113 development of a new global DSD, a DSD already in use by a number of international  
114 organizations may work well. This is not a recommendation for having an automatism for de  
115 facto standards becoming SDMX standard; the internationally agreed DSD could be  
116 considered as a starting point for the working group that develops the global DSD.  
117 Departmental DSDs are considered the lowest priority; their usage is merely adequate for  
118 data exchange within an institution or as a basis for developing a harmonized DSD for inter-  
119 organizational exchange. Overall, priority should be given to existing DSDs in the following  
120 order:

- 121 - global DSDs with maintenance agency “SDMX”;
- 122 - global DSDs with SDMX sponsor organization(s) as maintenance agency;
- 123 - other internationally agreed DSDs;
- 124 - nationally agreed DSDs;
- 125 - DSDs used by the organization;
- 126 - DSDs used by the department.

127 If none of the available DSDs is appropriate, it is still possible that existing concepts and/or  
128 code lists may be reused. (Only if the required concepts and code lists do not exist at all, a  
129 completely new DSD has to be developed with new concepts and new code lists.) Priority  
130 should be given to existing code lists in the following order:

- 131 - code lists recommended by the SDMX COG;
- 132 - other ISO code lists;
- 133 - code lists used by many SDMX sponsor organizations;
- 134 - other internationally agreed code lists;
- 135 - nationally agreed code lists;
- 136 - organization-wide code lists;
- 137 - departmental code lists.

138 The same disclaimers hold for code lists as for DSDs. Code lists used by many sponsor  
139 organizations or other internationally agreed code lists are a great basis for developing  
140 SDMX recommended code lists, and they can be used for data exchange if agreed by all  
141 parties. It is not suggested that they are accepted as SDMX recommended code lists  
142 automatically. Departmental code lists are considered the lowest priority. Their usage should

143 be avoided wherever possible, but is acceptable for data exchange within an institution or as  
144 a basis for developing a harmonized code list for inter-organizational exchange.

### 145 **2.1.3 Suitability of available DSDs and code lists**

146 In case an existing DSD is close to but differs from what is needed, it may: (i) contain  
147 irrelevant concepts, (ii) lack some required concepts, (iii) use the concepts in different roles  
148 than required, (iv) deviate with respect to some of the code lists, or (v) contain pure  
149 dimensions when mixed dimensions would make more sense or vice versa. More complex  
150 situations that are combinations of several (or even all) of these five cases may occur as  
151 well. For example, an existing DSD could contain unnecessary concepts and lack other  
152 concepts at the same time.

#### 153 **2.1.3.1 Irrelevant concepts**

154 Two options exist to deal with the situation of only a subset of dimensions being relevant<sup>1</sup>:

- 155 1. define a new, reduced concept scheme that includes only the relevant concepts and  
156 code lists by reference and a new DSD that uses the reduced concept scheme;
- 157 2. reuse concept scheme, code lists, and DSD, but add constraints to the data flow  
158 definition (or to the DSD, but this would also make it a new, derived DSD) that set the  
159 irrelevant dimensions to whatever applies from the following:
  - 160 a. If a concept is irrelevant because all observations take a particular value in  
161 that dimension or attribute, the concept should be restricted to that value via  
162 constraints in the data flow. For example “Unit” may be a dimension in a DSD  
163 because the data are disseminated in national currency, US Dollars, and  
164 percent change, but the new data exchange only allows US Dollars. Then the  
165 concept could be assigned to the attribute role (instead of the dimension role)  
166 which would entail defining a new DSD. This is not desirable if it can be  
167 avoided. Instead, the dimension can be kept and a constraint for “Unit” = US  
168 Dollars added.
  - 169 b. If a concept is obsolete because only total values aggregated over the  
170 corresponding dimension are relevant, the dimension (or code list) should be  
171 restricted to a “total” item. For instance, an existing DSD on bilateral trade  
172 contains “Partner Country” as dimension, since data are collected with a  
173 breakdown by country of trade counterpart. The new data exchange  
174 disseminates similar data, but only the trade totals “vis-à-vis all countries”.
  - 175 c. If a concept is not needed because it cannot even be relevant for the data at  
176 all or because an additional breakdown is just not available, the concept  
177 should be restricted to “not applicable” or “unknown” via constraints. For  
178 example, a financial instrument breakdown was not collected and it is unclear  
179 whether data for all or only for some financial instruments were included, that  
180 is whether the “total” value can be used. In this case, the dimension would be  
181 restricted to “unknown”. Consider another simple example of a DSD that  
182 contains, amongst others, the two dimensions “Unit of Measure” and “Base

---

<sup>1</sup> Technically speaking, a third possibility exists. A structure map can be used to define the reduced DSD. The structure map establishes a mapping between a source structure and a target structure. In this special case, the aim of the structure map is simply to get rid of irrelevant dimensions. To this end the DSD is mapped to itself, and any unmapped dimensions will not be part of the target structure. The original DSD is not affected by the structure map. The reduced DSD can be derived from the structure map, but from a technical point of view there is no need to actually create the reduced DSD as an artefact. It can exist as a “virtual” DSD that is merely defined by the Structure Map.



183 Period". The code list for "Unit of Measure" consists of percent per annum,  
184 percentage, and index with base year=100. "Base Period" can contain dates,  
185 years, months, etc. If the new data exchange restricts "Unit of Measure" to  
186 percent per annum, "Base Period" becomes obsolete and should be  
187 constrained to "not applicable".

### 188 2.1.3.2 Missing concepts

189 In this case additional concepts (and possibly code lists) are required, for example to  
190 accommodate an additional cross-classification. One option is to adapt the existing DSD to  
191 satisfy the new needs, i.e., create a new version of the DSD by adding the concepts,  
192 dimensions/attributes, and code lists. The feasibility of this solution depends on the relation  
193 between the organization requiring the new data structure and the organization maintaining  
194 the existing DSD, and the relation between the two usage contexts. The original, restricted  
195 model needed for the existing data flows can be specified by means of constraints, as  
196 described above for irrelevant concepts, or by referring to the original version of the DSD.

197 If a modification of the existing DSD does not make sense or is not possible, the relevant  
198 concepts and code lists should be reused, but an extended concept scheme, an extended  
199 (new) DSD, and maybe also additional code lists need to be defined. If a code list already  
200 used by the DSD applies to the new dimension/attribute, it can be reused. An example is the  
201 inclusion of a "Partner Country" breakdown for which the already defined "Reference Area"  
202 code list can be reused. If additional code lists are necessary, again different scenarios are  
203 feasible:

- 204 1. The required code list is available somewhere else. In this case the priority ranking  
205 provided above should be applied. For instance if an additional sector breakdown is  
206 required, the sector code list defined in the global DSD for National Accounts can be  
207 referred to.
- 208 2. A code list similar to what is needed is available somewhere else.
  - 209 a. If only a subset of the existing code list is relevant, the code list can be reused  
210 with a constraint imposed either on the code list, or in the DSD, or in the data  
211 flow definition (or in the data provisioning agreement). It is also possible to  
212 use the entire code list but only report data for the subset.
  - 213 b. In case a (different) hierarchy is needed, the underlying flat code list can be  
214 referenced and a new hierarchical code list introduced. This means that a flat  
215 code list (i.e. without an explicitly defined hierarchy) is available that meets  
216 the coverage requirements, but that the existing hierarchy defined on top of  
217 the flat code list deviates from the required hierarchy. Hence, the suitable flat  
218 code list can be reused, but a new hierarchical code list needs to be defined.  
219 Consider for instance the "Reference Area" code list as recommended by the  
220 SDMX Content-Oriented Guidelines (COG), i.e. containing ISO-2-character  
221 codes for countries. Different groupings of these countries are relevant in  
222 different contexts, for example, regional aggregates by continent, by income  
223 level, or by membership in certain international groups (e.g. monetary  
224 unions). A flat code list can be defined that contains all these country groups  
225 in addition to the individual countries. This list does not specify parent-child  
226 relationships between groups and countries, as this would entail repeating  
227 countries for each group they belong to. It basically provides the value  
228 domain for a geographic dimension, but not the semantics of the values in  
229 terms of the group composition.
- 230 On top of this flat code list, different hierarchical code lists can be defined that  
231 may use the complete set of codes or just a subset thereof. The flat code list

- 232 can be referenced by any DSD with a geographical reference, and different  
233 DSDs can build their own hierarchical code lists based on the flat list.
- 234 c. If additional items are needed, a derived code list can be specified by  
235 including each element from the existing code list by reference and adding  
236 the new elements as required. The current versions of the SDMX Technical  
237 Standard do not allow combining existing code lists into one or referencing an  
238 entire code list and adding a few elements to be managed in the new code  
239 list. Often, simply a copy of the existing list is introduced as new code list with  
240 the new items included. This is not optimal, as conceptually identical items  
241 have to be managed in multiple code lists. At least in theory it is also possible  
242 to just create a new version of the existing code list with the additional items.  
243 Existing data flows would then either use the original version of the code list  
244 or the new version with constraints, whereas the new version of the code list  
245 would be used in the new data flow. Again, this option depends on the  
246 organizational background.
- 247 Consider as an example the inclusion of “Currency” into a DSD with a need  
248 for codes for “Domestic currency” and “Foreign currency” in addition to the  
249 codes specified in the code list recommended by the SDMX COG. In the first  
250 option, the currencies from the recommended code list are included by  
251 reference and the two new items added to a new code list. This is superior to  
252 the common practice of including copies of the existing codes (the currencies)  
253 instead of references. This makes the new code list more independent of the  
254 existing one, but it increases the maintenance cost and the risk of  
255 inconsistencies. Another option is to extend the existing code list by creating  
256 a new code list version. In the currency example, the SDMX consortium as  
257 the owner of the recommended code list would need to decide whether this  
258 new version should be created or not.
- 259 3. No appropriate code lists are available. New code lists have to be defined based on  
260 the guidelines for the development of code lists. This may often be the case for  
261 domain-specific code lists, especially in new areas of investigation.

### 262 **2.1.3.3 Concepts in different roles**

263 In this case concepts are available in other roles than required, for example what needs to  
264 be a dimension is merely an attribute or vice versa. This case is already briefly discussed  
265 above as part of the first case (“irrelevant concepts”). Basically, a new DSD has to be  
266 defined. It can reuse the concept scheme and code lists, but specifies the concepts in the  
267 new DSD as dimensions or attributes as required. In case an attribute needs to become a  
268 dimension, it may be necessary to define a new code list for that dimension in case it did not  
269 exist previously.

270 An example for an attribute having to be redefined as a dimension may be the “Unit of  
271 measure” that is frequently just specified as an attribute. If certain indicators are presented in  
272 different units in the same data flow, the corresponding DSD must contain “Unit of measure”  
273 as dimension, though.

### 274 **2.1.3.4 Different code lists**

275 In this case the new requirements differ from the existing DSD with respect to some of the  
276 code lists, either by only a subset of codes being relevant, by a deviating hierarchical  
277 structure, or by necessitating additional codes. These three scenarios are discussed above  
278 as special cases of the “missing concepts” case. In theory, just defining a new code list  
279 whenever an existing one is not completely appropriate is also possible (but not desirable).  
280 However, this means that the overlapping items have to be managed in multiple code lists  
281 unless they are included by reference. Also, different DSDs have to be maintained. If the

282 constraints are neither imposed at the code list nor at the DSD level, but at the data flow  
283 level, the DSD is simply reused. This is highly recommended. The cost of maintaining  
284 multiple DSDs or multiple, largely overlapping code lists can be high. The lack of  
285 harmonization has one advantage, though: the increased maintenance and versioning  
286 flexibility. For global data exchange, this is not regarded as a reasonable solution.

#### 287 **2.1.3.5 Pure vs. mixed dimensions**

288 The design principle of pure dimensions is explained in more detail in subsections 3.3 and  
289 section 4 on data structuring approaches. If an existing DSD does not have the desired  
290 degree of dimension purity, it is necessary to further decompose and/or combine dimensions  
291 of that DSD. This will lead to a new derived DSD and also requires the definition of new  
292 (combined or split) code lists, unless they are available from elsewhere.

### 293 **2.2 Flexibility and future needs**

294 As already mentioned in the initial statement of this section, DSD design should take into  
295 account *potential future needs*. A DSD should be flexible enough to accommodate changing  
296 requirements and still remain as stable as possible for a reasonable time period (e.g. five  
297 years). Given the possibly high development and implementation costs, users should be  
298 able to rely on a stable DSD as a data exchange standard for a certain data flow. Changes  
299 in DSDs may have implications for data providers' and consumers' processes and may incur  
300 adjustment costs.

301 This future-orientation may require the introduction of a dimension that is not relevant at the  
302 time of DSD design but known (or suspected) to become relevant despite the (at least)  
303 temporary redundancy of the additional dimension. For example, it may be likely that certain  
304 additional variables will be introduced in a data collection instrument in the future. Even if it is  
305 unknown whether this will really happen and if so, when, it is reasonable to include those  
306 additional concepts in the DSD from the beginning and use a “total” or “not applicable” value  
307 for that concept until it gets implemented in the data collection exercise.

### 308 **2.3 Structural principles**

309 In terms of the data structure itself, *parsimony*, *simplicity*, *exhaustiveness*,  
310 *unambiguousness*, *orthogonality*, and *density* of the dimensional model should be taken into  
311 account.

#### 312 **2.3.1 Parsimony**

313 A *parsimonious* DSD does not contain any redundant dimensions that are not needed to  
314 uniquely identify a data point. It may contain concepts that are not needed for data  
315 identification, but those take the role of attributes that further describe observations. It  
316 attaches those attributes at the highest possible level, i.e. to groups of observations that  
317 share the same value of an attribute. For example, if all data for Country “Canada” are  
318 provided in “Canadian Dollars”, for “US” in “US Dollars”, etc., the Unit may be defined as an  
319 attribute at Country level. This means it only has to be specified once for each value of  
320 Country.

#### 321 **2.3.2 Simplicity**

322 A *simple* DSD is often considered as one that keeps the observation keys (or identifiers) as  
323 short as possible by keeping the number of dimensions to the absolute minimum. This is  
324 related to the parsimony of DSDs, but typically goes beyond that by using what is often  
325 called “mixed dimensions”, i.e., dimensions that combine different concepts. If this idea were  
326 taken to an extreme, there would be only one dimension containing an observation key.

### 327 2.3.3 Purity

328 The *purity* of concepts, especially dimensions, is a principle that is in conflict with the aim of  
 329 DSD simplicity. Pure dimensions only relate to one pure concept, not to a combination of  
 330 concepts. They usually have shorter and less complex code lists than “mixed dimensions”.  
 331 Balancing these two antagonistic principles can be difficult; it is discussed in more detail and  
 332 with a few examples in section 4 on data structuring approaches.

### 333 2.3.4 Density and sparseness

334 The *density* of a DSD is closely related to its simplicity whereas *sparseness* often comes  
 335 along with purity. For a dense DSD, a data flow provides data for all (or the large majority of)  
 336 cells defined by the Cartesian product<sup>2</sup> of the DSD dimensions. This is typically the case for  
 337 simple DSDs. For pure DSDs with many dimensions, it is usually not feasible to share data  
 338 for the entire data space created by the combination of all dimensions.

339 For example, a breakdown by “Institutional Sector” or “Gender” may only make sense for a  
 340 subset of the “Indicators” provided. The sparseness may be measured in terms of the  
 341 number of dimensions requiring a “not applicable” value or the number of observations that  
 342 take at least one “not applicable” or “total” value (both as shares of the total number of  
 343 dimension or the total number of observations, respectively)<sup>3</sup>. An even more precise  
 344 measure of sparseness is the proportion of theoretically possible key combinations that are  
 345 irrelevant or not feasible or do not carry data.

### 346 2.3.5 Unambiguousness

347 Another important DSD design principle is *unambiguousness*. It should be avoided that one  
 348 observation can be expressed by multiple combinations of dimension values (keys). This  
 349 may occur when multiple dimensions are used to express similar or even overlapping  
 350 concepts. To illustrate the principle of unambiguousness, consider the following example  
 351 with four dimensions (apart from country and time) and value domains as depicted in  
 352 Table<sup>o</sup>1.

353 **Table 1. Unambiguousness example – dimensions**

Indicator	Measurement	Unit	Scale
GDP	Current prices	National currency	Units
GDP nominal	Millions of national currency, current prices	US \$	Thousands
GDP real	Millions of US \$, current prices	Index	Millions
GDP deflator	Constant prices	Euro	Billions
Consumer prices	Millions of national currency, constant prices	Euro, 2005	...
...	Millions of national currency, 2005 prices	US \$, 2005	
	Millions of US \$, constant prices	US \$, 2010	
	Millions of US \$, 2005 prices		

354

<sup>2</sup> A Cartesian product (or product set) is a mathematical construct that builds a new set out of a number of given sets. Each member of the Cartesian product corresponds to the selection of one element each in every one of the original sets.

<sup>3</sup> In case a structure map is used to define reduced versions of the DSD, the number of unmapped dimensions is the equivalent measure of sparseness.

355 How would an observation of “Gross domestic product, volume, US dollars, reference year =  
356 2005, millions” for the United States be represented with these dimensions? Table 2  
357 provides three different possible representations (there may be even more).

358 **Table 2. Unambiguousness example – ambiguous representations**

Indicator	Measurement	Unit	Scale
GDP	Constant prices	US \$, 2005	Millions
GDP real	Millions of national currency, 2005 prices	US \$	Units
GDP	Millions of US \$, constant prices	US \$, 2005	Millions

359

360 In this simplified example, there are several ways of resolving the ambiguity. In practice,  
361 overlaps and ambiguities are often less obvious and finding an unambiguous solution may  
362 be less straightforward.

### 363 **2.3.6 Exhaustiveness**

364 An *exhaustive* DSD includes every piece of information that is required to unambiguously  
365 represent a data point and to correctly interpret it outside its usual context. It may not be  
366 necessary to specify the respective concepts as dimensions, but if they are attributes they  
367 should be made mandatory. For instance it may be absolutely clear that all data in a certain  
368 database are measured in millions of Euros, but once the data are shared and thus available  
369 outside the context of the original database, how would a consumer of those data know -  
370 unless s/he is told so?

### 371 **2.3.7 Orthogonality**

372 *Orthogonality* of DSD dimensions corresponds to the independence of the meaning of a  
373 value of one dimension from the values of any other dimensions. Orthogonality helps to  
374 avoid ambiguity. In the example for lacking unambiguousness above, the dimensions are not  
375 orthogonal but show a semantic overlap that leads to dependencies between the  
376 dimensions.

377 For instance, dimensions “Indicator” and “Measure” are dependent; indicator “GDP real”  
378 cannot be combined with any of the “Current prices” measures. Another example from the  
379 tables above is “Scale” and its dependence on “Measurement” and “Unit of measure”. “Unit”  
380 combined with “Current prices” and “US \$” really means “Unit”, i.e. the indicator is presented  
381 in (units of) US \$, current prices; but if “Unit” is combined with “Millions of US \$, current  
382 prices” and some “Unit of measure”, the indicator is presented in millions of (units of) US \$,  
383 current prices. The meaning of “Scale” equal to “Unit” changes in dependence of the values  
384 of the other dimensions.

### 385 **2.3.8 User-friendliness**

386 The *user-friendliness* of a DSD may also be regarded as a general design principle. It is  
387 often said to increase with the simplicity of a DSD, but this is not necessarily the case. User-  
388 friendliness of a DSD mainly depends on the data sharing context, on the tools used to deal  
389 with the DSD and the data, and on the role of the user (e.g. the requirements of a DSD  
390 manager may be different from those of a researcher looking for certain time series in a  
391 disseminated dataset). While a simple DSD consisting of a few dimensions only may be  
392 easier to understand by a human data consumer, a more complex, but purer DSD is typically  
393 more flexible in terms of further usage in automated processes. These aspects are  
394 discussed in more detail in the following sections.

### 395 **2.3.9 Fitness for use throughout the statistical business process**

396 Another DSD requirement is its *fitness for use throughout the entire statistical business*  
397 *process*, that is, at least from data collection through processing to exchange and  
398 dissemination. This means that data producers', consumers', and metadata managers' needs  
399 should be taken into account in the design process. The requirements may diverge as more  
400 detailed data may often be collected than disseminated. Similarly, data sharing at the  
401 national level may be more granular or require somewhat different code lists than data  
402 sharing between national and international organizations or data sharing on the web for the  
403 general public. This divergence can be addressed by means of a “master DSD” and related  
404 “satellite DSDs”. The master DSD has all concepts and code lists that are required  
405 throughout the process. The satellite or sub-DSDs are derived from the master DSD and  
406 refer to the same concepts and code lists, but specify constraints (and possibly structure  
407 maps) to restrict the DSD to what is needed at a certain stage in the process. This helps  
408 maximize the extent to which artefacts are shared between the DSDs, and hence  
409 harmonized.

## 410 **3 USAGE CONTEXTS**

411 Different DSD usage contexts have specific requirements and different data structuring  
412 approaches suit these requirements to varying extents.

### 413 **3.1 Type of data**

414 For example, *time series* data require Time to be a dimension in the data structure definition,  
415 while it may just be a (mandatory) attribute for *cross-sectional* data. Similarly, *micro data* (not  
416 covered by this document) need a dimension that uniquely identifies each observation unit,  
417 whereas aggregated data do not have this requirement.

### 418 **3.2 Domain**

419 A related distinction is the one between *single-* and *cross-domain* (or *multi-domain*) data  
420 structures. For cross-domain data it may be difficult to define a single DSD with “pure”  
421 concepts. Consider for instance a data structure that is supposed to cover selected labor  
422 market and trade indicators. Cross-domain concepts such as Reporting Country, Frequency,  
423 and Unit of Measure, obviously apply to both domains. Besides, the two domains may share  
424 additional classification concepts, e.g., the corresponding type of economic activity/product  
425 (agriculture, manufacturing, health, etc.).

426 Other relevant concepts differ between the domains, though. Labor market indicators may  
427 include breakdowns by gender or age, whereas trade statistics may contain additional cross-  
428 classifications by terms of trade or destination country. This raises a couple of questions:  
429 Should all concepts be put into one DSD, despite the applicability of some concepts to only  
430 one of the two domains? Should this be done by combining the relevant concepts into one  
431 dimension with a longer (and maybe hierarchical) code list? Or is it preferable to split the  
432 data structure into one DSD for each domain covered?

### 433 **3.3 Purpose**

434 Questions like these also apply to *multi-purpose* (as opposed to *single-purpose*) data  
435 structures. Multi-purpose data structures are typically used in different, related data  
436 exchange exercises (that may be represented by different data flows). They are used to  
437 collect and/or disseminate related data, typically in the same domain(s), by different  
438 organizations or by one organization.

439 An example for a multi-purpose scenario is a supra-national organization such as Eurostat or  
 440 the ECB acting as a “data hub” for its member countries in terms of data exchange with  
 441 international organizations like the IMF or the UN. In this scenario, for instance the ECB may  
 442 collect data for its own purposes, but also for its member countries’ reporting duties to the  
 443 IMF, the OECD, and the BIS. The data would (partially) be redistributed to the international  
 444 organizations so national banks and statistics offices would not have to report the same (or  
 445 very similar) data many times.

446 The global BOP DSD that is currently being developed may serve as a more specific  
 447 example for a multi-purpose DSD. It is supposed to support, amongst others, exchange of  
 448 the ECB's Balance of Payments (BOP) and International Reserves Template (IRT) data,  
 449 Eurostat's International Investment Position (IIP) and Trade in Services (TS) data, the  
 450 OECD's BOP data, and the IMF's Coordinated Portfolio Investment (CPIS) and Coordinated  
 451 Direct Investment (CDIS) data.

452 Table 3 below shows some of the concepts considered relevant for some or all of these  
 453 related data exchange exercises.<sup>4</sup> Reporting Country and Unit of Measure are required by all  
 454 data exchanges; the other concepts listed are only necessary (marked by an “X”) for a  
 455 subset of the data exchanges. For instance, Eurostat's TS and IMF's CDIS data do not  
 456 require the distinction of flows and stocks, different maturities, or valuations (indicated by an  
 457 “O”). Still, there is value in defining one master DSD that covers all concepts required for all  
 458 of the data exchanges.

459 If that approach is pursued, satellite DSDs for the individual purposes (or exchange  
 460 exercises) can be created via constraints (and/or structure maps). Each exchange exercise  
 461 may also be represented as a data flow (the constraints may also be defined in the data flow  
 462 instead of the DSD). So there would be one data flow defined for each column in the table  
 463 below. For instance, the IMF CPIS data flow would restrict “Flows and stocks indicator” and  
 464 “Valuation” to certain values from the respective code lists. Data provision agreements may  
 465 then be set up for each data flow with each reporting country. Constraints can be used to  
 466 restrict the contribution of each country to its own data, so “Reporting country” would be set  
 467 to the respective value. If the constraints are defined in the data flow and/or structure maps  
 468 are used to exclude irrelevant dimensions, the satellite DSDs do not materialize; they are  
 469 “virtual” DSDs.

470 **Table 3. Excerpt of concepts and data exchange exercises relevant for the global BOP DSD**  
 471 **(X=Yes)**

Concept	ECB		Eurostat		OECD	IMF	
	IRT	BOP	IIP	TS	BOP	CPIS	CDIS
Reporting country or area	X	X	X	X	X	X	X
Unit of measure	X	X	X	X	X	X	X
Flows and stocks indicator	X	O	O	O	O	O	O
Reporting sector	X	X	X	O	X	X	X
Financial instrument	X	X	X	O	X	X	X
Maturity	X	X	X	O	X	X	O
Valuation	X	O	X	O	O	O	O

<sup>4</sup> Please note that the example is taken from the development status of the BOP DSD at the time of writing this document. The concepts and their relevance for certain data exchanges (represented as data flows or derived DSDs) may be different in the final version of the DSD.

### 472 **3.4 Type of data exchange and recipient**

473 The type or *level of data exchange* also plays an important role. In terms of required  
474 concepts, data exchange *within an organization* may necessitate less context information  
475 (that is, less (mandatory) attributes) than data exchange *between organizations*. Referring to  
476 official standards may provide this context information as well, even for exchanges between  
477 organizations. International data exchanges, no matter if *among international organizations*  
478 or *between international organizations and national member organizations*, typically aim at  
479 cross-country comparisons of (highly) aggregated indicators. National data exchanges often  
480 require more detailed data structures (e.g., longer code lists or further concepts for additional  
481 breakdowns), alternative code labels (in national languages), or additional concepts that  
482 explain national methodologies that may differ from standard or recommended  
483 methodologies underlying standard code lists.

484 Data *dissemination to the general public* usually involves interaction with *human users* and  
485 hence requires less complex data structures and easier-to-grasp data discovery and retrieval  
486 mechanisms than *machine-to-machine* communication that is often used within and between  
487 organizations. As demonstrated by the recent emergence of Open Data initiatives, there is a  
488 growing demand to make data publicly available and to enable automated reading of data  
489 from the web via application programming interfaces (APIs).

### 490 **3.5 Role in data exchange**

491 In addition to the type of data exchange and the type of data recipient (machine or human),  
492 an actor's *role* determines whether certain features of data structuring approaches are  
493 regarded as pros or cons. A very complex DSD with many dimensions may be beneficial  
494 from a *data collection and processing* point of view because of its flexibility, but less  
495 attractive from the perspective of the *data provider* in the same data exchange. A data  
496 provider may find it easier to set up a mapping from the data production system to simple  
497 observation keys. However, this is merely a perceptual issue, as it is always possible to  
498 specify a list of admissible observation (or time series) keys as combination of dimension  
499 values that can be used for the mapping. Similarly, fewer dimensions may be better suited  
500 for human consumption of disseminated data, although a high complexity of the resulting  
501 "composite" dimensions may outweigh this initial advantage. Also, end users may appreciate  
502 increased flexibility in creating their queries provided by a higher-dimensional data structure.

### 503 **3.6 Process pattern**

504 The *process pattern* also contributes to the data exchange scenario. *Bilateral* exchange,  
505 *gateway* exchange, and *data-sharing* exchange are discerned. Gateway exchange  
506 corresponds to an organized set of bilateral exchanges with a single known format using a  
507 single known process. Data-sharing exchange means that there are no bilateral data  
508 exchange agreements. Instead, open, freely available data formats (at best: standards) that  
509 anyone can consume are adhered to. The major differences of these process patterns in  
510 terms of data structuring requirements are the relevance of standards and the level of  
511 generality. The DSD for a bilateral data exchange may be more specifically designed to  
512 meet the particular needs of the two data exchanging parties, standards are less important,  
513 and the DSD is more likely to be set up on an ad-hoc basis than for a more generic process  
514 pattern such as the gateway or data sharing exchange.

### 515 **3.7 Phase in statistical business process**

516 SDMX was developed primarily for data exchange and, hence, is often regarded as relevant  
517 mainly for the collection and dissemination phases of the statistical business process (as



518 defined by the *GSBPM*). Still, considerations concerning different data structuring  
519 approaches may also be made with respect to *all process phases*. For example, a more  
520 granular data structure is more flexible in terms of further processing and analysis.

## 521 **4 DATA STRUCTURING APPROACHES**

522 Two major challenges in DSD development are the specification of (i) the number and  
523 content of the dimensions required to identify an observation, and (ii) the number of DSDs  
524 needed. The former is due to the tradeoff between vertical and horizontal data structure  
525 complexity. High *horizontal* or *between-dimension complexity* refers to a very granular  
526 decomposition of the observation key or identifier into many dimensions with shorter code  
527 lists. In contrast, high *vertical* or *within-dimension complexity* is characterized by fewer but  
528 more complex dimensions with longer code lists (that are typically more complex with more  
529 hierarchy levels).

530 These “*composite*” or “*mixed*” *dimensions* are usually easier to understand by end users, but  
531 less flexible in terms of re-usage by other systems and adaptation to future requirements.  
532 Moreover, shorter and less complex code lists are easier to maintain, even if the number of  
533 code lists is higher. However, the specification of the subset of observation keys valid and/or  
534 relevant in a data flow by means of constraints is more intricate for a DSD with many  
535 dimensions. The theoretically possible set of observation keys defined by the Cartesian  
536 product of the code lists involved may be only sparsely covered by actual (or observable)  
537 data.

538 In a horizontally complex DSD with many dimensions, some dimensions may need a value  
539 “not applicable” or “total” so that different parts of a data flow that may be provided at  
540 different levels of granularity can be represented. This can be regarded as an indication that  
541 the DSD should be split into multiple DSDs, as not all parts of the data flow make use of the  
542 full set of dimensions. However, a multi-DSD approach typically entails higher maintenance  
543 costs and requires more processing resources in data production as compared to a single  
544 master DSD approach.

545 Another means of avoiding the heavy usage of “not applicable” values is increasing the  
546 vertical complexity of the DSD by creating composite dimensions. These mixed dimensions  
547 then have code lists with composite values; the “not applicable” values of the individual code  
548 lists are simply omitted when concatenating the values. The composite code list only  
549 requires a “not applicable” value for the case of all “component” values being “not applicable”  
550 (that is, none of the dimensions combined in the mixed dimension applies).

551 It is not obvious how to define the optimal DSD(s) for a domain that balances these pros and  
552 cons; it largely depends on whether the focus is on ease of DSD and code list maintenance  
553 (incl. flexibility, re-usability, and adaptability) or end user friendliness and whether only  
554 certain stages of or the entire statistical business process (e.g. collection, exchange,  
555 dissemination) should be covered.

556 Currently, the SDMX Standard (V2.1) does not specify any mandatory requirements with  
557 regard to the number of dimensions, the purity of dimensions, or the number of DSDs to be  
558 used to represent a domain. The SDMX Technical Notes (Section 6 of the Standards  
559 documentation) provide some recommendations in section 3.4.1.2 “Defining Data Structure  
560 Definitions (DSDs)”, but those are explicitly defined as being not normative. The  
561 recommendations include “Avoid dimensions that are not appropriate for all the series in the  
562 data structure definition”, “Devise DSDs with a small number of Dimensions for public  
563 viewing of data”, and “Avoid composite dimensions”. As discussed it is neither possible nor

564 does it seem necessary to satisfactorily implement these reasonable, but partly conflicting  
565 suggestions at the same time.

#### 566 **4.1 Number and content of concepts**

567 The decision on content and number of concepts in a DSD usually leads to the question of  
568 how far the “*indicator*” dimension should be decomposed. There are some (cross-domain)  
569 concepts, such as geographical and temporal reference and unit of measure, that are  
570 relevant in most DSDs. Once those are defined (the usage of the SDMX COG is highly  
571 recommended!) the actual “*subject-matter*” or “*domain*” concepts remain. One option is to  
572 combine all those concepts into one “*indicator*” dimension which may make sense in certain  
573 scenarios, for example for smaller single-domain, single-purpose DSDs with few or no cross-  
574 classifications or for display in an end-user dissemination tool. The other extreme strategy is  
575 to decompose into as many components as possible by splitting any breakdown concepts  
576 from the core indicator concept.

577 The range of options between the “*just one*” (mixed) and “*all component*” subject-matter  
578 dimensions approaches is subject to the comprehensiveness (i.e. size, coverage) of the data  
579 exchange that the DSD is being developed for. If using a “mixed dimensions” approach,  
580 rules for the composition of the mixed dimension(s) may be specified (e.g. concatenate  
581 concepts A, B, and C to get mixed dimension X), allowing their easy re-decomposition. In  
582 general composite dimensions should be avoided as previously recommended by the SDMX  
583 Technical Notes, but there are cases that suggest the usage of composite dimensions. Table  
584 4 juxtaposes general pros and cons of the “*many pure concepts*” and “*fewer composite*  
585 *concepts*” approaches.

586 **Table 4. General comparison of data structuring approaches**

<b>Many pure concepts</b>	<b>Few composite concepts</b>						
cleaner data structure	Mixed dimensions may be composed inconsistently making the decomposition into purer concepts and code lists difficult (requiring complex mapping etc.). Information that corresponds to the same concept may be included in different dimensions, e.g. reference year is contained in the indicator dimension in the first example but in the unit in the second example below. The optimal common data structure would consist of Economic Indicator, Unit, and Base period. <table border="1" data-bbox="774 1523 1347 1702"> <thead> <tr> <th data-bbox="774 1523 1069 1568"><b>Economic Indicator</b></th> <th data-bbox="1069 1523 1347 1568"><b>Unit</b></th> </tr> </thead> <tbody> <tr> <td data-bbox="774 1568 1069 1635">Industrial production (2000=100)</td> <td data-bbox="1069 1568 1347 1635">Index</td> </tr> <tr> <td data-bbox="774 1635 1069 1702">GDP real</td> <td data-bbox="1069 1635 1347 1702">US Dollars at 2005 prices</td> </tr> </tbody> </table>	<b>Economic Indicator</b>	<b>Unit</b>	Industrial production (2000=100)	Index	GDP real	US Dollars at 2005 prices
<b>Economic Indicator</b>	<b>Unit</b>						
Industrial production (2000=100)	Index						
GDP real	US Dollars at 2005 prices						
shorter and simpler code lists	code lists longer and more complex, may require hierarchy to be “readable”						

<b>Many pure concepts</b>	<b>Few composite concepts</b>
more flexible in terms of defining constraints, but constraints more complex	simpler constraints, but some constraints may be difficult to be represented because of mixed dimensions. Consider for instance a constraint “Base period = 1995” in the above example, where some observations include the base period in the Economic Indicator dimension, others in the Unit dimension. Instead of specifying a constraint on a pure Base Period dimension, the constraints may have to be specified at observation (or time series) level
more flexible in terms of mapping to other data structures (used by other systems), further processing and analysis (e.g. tabulation, dissemination format), and future needs	“mixed” dimensions make data structure less flexible in these respects
longer (i.e. more complex) observation keys	shorter keys
special values of code lists such as “not applicable”, “total” may be rather heavily used	less usage of these special values
creates sparse data if many observations use “not applicable”	way to avoid sparseness
many constraints may be necessary due to sparseness	typically fewer constraints required because data are less sparse
many dimensions are tantamount to many attachment levels for attributes (i.e. DSD more flexible in terms of attribute attachment)	less dimensions = less possible attribute attachment levels
more difficult to handle by an end user	presumably more easily comprehensible and manageable by an end user
more flexible in terms of defining queries; can be mapped to any “mixed” representation	less flexible in terms of search and retrieval

587

588 The latter two aspects mentioned in the table could be summarized as the “many pure  
589 dimensions” approach being more difficult to handle for a “basic” user, but providing fewer  
590 options for an “advanced” user. When it comes to dissemination to end users, a purer data  
591 structure is the appropriate format for consumption by applications and advanced users. For  
592 less advanced user groups it makes sense to hide the (for them: unnecessary) complexity by  
593 means of concatenating dimensions, for instance to create a time series view.

594 Comparing single-purpose and single-domain exchange scenarios with multi-domain and/or  
595 multi-purpose scenarios, pure concepts are typically easier to achieve in the former,  
596 whereas composite concepts/dimensions may make life easier in the latter, especially  
597 because certain cross-classification concepts may only apply to some domains and/or  
598 purposes covered. “Purpose” means either a certain data exchange exercise or data flow,  
599 for instance in the BOP DSD endeavor mentioned above each column represents one  
600 “purpose”, e.g. ECB IRT or OECD BOP. In multi-domain or –purpose scenarios, pure  
601 concepts are more easily obtained by a “many DSDs” approach, no matter if those are  
602 independent from each other or linked by a “master DSD”. Although it does not rule out the  
603 specification of pure concepts, a “one DSD” approach typically leads to using fewer,  
604 composite concepts (dimensions) in those scenarios.

605 Table 5 provides an overview of the pros and cons of the “many pure concepts” and “fewer  
606 composite concepts” approaches in different data exchange settings with respect to the type  
607 of organizations involved. In any of these settings it is always possible to use one of the data  
608 structures that may already exist at one of the involved parties as DSD for the data

609 exchange. The benefits and drawbacks discussed in the table assume that a new DSD is to  
 610 be defined. A distinction between two different types of intended recipients is implicitly made.  
 611 Inter-organizational data exchange is mostly machine-to-machine, whereas dissemination of  
 612 data to end-users is often machine-to-user.

613 **Table 5. Data structuring approaches by level of data exchange**

<b>Level of data exchange</b>	<b>Pure vs. composite concepts approach</b>
<b>within an organization</b>	<p>Depends on diversity of systems involved in data exchange.</p> <p>The approach that requires the least mapping (and similar processing) steps between the two communicating data structures is preferable in terms of a “quick win” solution.</p> <p>In general, a more granular model is preferable due to its flexibility that helps support potential future needs (with respect to processing, analysis, exchange, dissemination, etc.).</p> <p>However, an internal exchange should not be made more complex than necessary. If the structures of the communicating systems are comparable, it may not make sense to create an artificial intermediary structure that is more pure, but also more complex than both underlying structures.</p> <p>Still, as a longer-term strategy it seems reasonable to define a set of internal “standard” code lists that all systems can map to. This allows bilateral communication via the shared concepts and code lists meaning that every data structure only has to be mapped once – to the internal standard – to be able to communicate with all other participating (i.e. mapped) systems.</p>
<b>between organizations at national level</b>	<p>The pros and cons at this level of exchange are comparable to those at the “within organization” level. If the data structures of the communicating systems are comparable, there is no need to introduce complexity by a conceptually optimal, pure data structure. However, if the data structures deviate to a greater extent (and they often do), they should both be decomposed to find a “common denominator”, a more granular “exchange vocabulary” which they can be mapped to.</p> <p>If related international or national standards exist, they should be used, even though national labels and/or additional levels of detail may be required in the code lists.</p>
<b>between international organization and national organizations of member countries</b>	<p>International organizations should collect data at a level of granularity and purity that is most suitable for the intended (and potential future) analyses. The tradeoff with the higher complexity of constraints required to check structural validity of collected data needs to be taken into account as well. Also it is recommended to consider the burden that a more complex data structure may put on national data providers. However, once a DSD is defined, its lifetime is expected to be a number of years. The main effort of the data provider is to specify the mapping from the production data structure to the DSD. Once this is done the data exchange can be automated and the complexity of the DSD does not matter that much.</p>

Level of data exchange	Pure vs. composite concepts approach
<b>between international organizations</b>	Ideally, international organizations agree on code lists and DSDs to collect their data from member countries AND exchange data among them. Such a data structure should be as granular and pure as required for the intended uses of the data; one may even say, as pure as possible, with the constraint that it should not become too sparse. (The sparseness may be dealt with by constraints, though.) It would be great to provide a concrete numerical threshold for sparseness, but there is not yet enough experience and empirical evidence. Hence, for the time being, this question is, to a certain extent, a matter of preference and left to the use of one's common sense.
<b>between organizations and the public</b>	Mixed dimensions are often easier to handle by end users. They can be easily defined from a pure data structure in the background. Multiple presentation data structures with hierarchies may be required, as the needs typically differ by type of end user to be addressed. Tables and charts (visualizations) for "basic" users often contain highly compressed information (i.e. mixed dimensions), whereas more advanced users require more flexibility, detail, and granularity. These dissemination or presentation data structures allow the removal of "not applicable" dimensions as well as the usage of attributes in table/chart titles or footnotes. To improve the ease of data discovery, dissemination data structures should only contain concepts and codes for which data are available. This may be achieved by means of content constraints and/or structure maps or by creating the DSDs "on the fly".

614

615 In addition to the different levels of data exchange, the type of exchange as defined by the  
 616 process pattern (bilateral, gateway, or data-sharing) plays a role in the decision of pure vs.  
 617 composite concepts. The purity of the model is less important for bilateral exchanges and  
 618 ad-hoc or short-term scenarios as opposed to gateway or data-sharing exchanges. Still, the  
 619 general advantages of purer data structures apply. Data structures with fewer, mixed  
 620 concepts have their merits only when they are very close to the existing data structures in  
 621 the communicating systems.

622 Finally, the perspectives of different actors/roles in the discussed exchange scenarios are  
 623 briefly described in Table 6.

624

625 **Table 6. Data structuring approaches by role in data exchange**

<b>Role in data exchange</b>	<b>Pure vs. composite concepts approach</b>
<b>Data provider</b>	<p>If the composition of the concepts in the data provider's production system largely differs from the one in the DSD, mapping it to a few composite concepts may be more complex than mapping it to many pure concepts. (Mapping to just one mixed concept is straightforward, though.) This is due to the need to decompose and recombine concepts in case of a "mixed concepts" DSD. If the data provider's internal data structure is very granular or very similar to the DSD, it does not make a huge difference if the concepts in that DSD are pure or not.</p> <p>For a "final" data provider disseminating data to the public, the flexibility offered by a pure data structure in terms of defining different output formats may be beneficial.</p>
<b>Data collector</b>	<p>Defining constraints for data validation is more complex for a high-dimensional, pure DSD. But such a DSD provides more flexibility in terms of consumption and reuse, i.e. mapping to the data collector's internal data model mapping easier.</p>
<b>DSD maintenance</b>	<p>Pure concepts usually have shorter, less complex code lists and are thus easier to maintain. In contrast, the maintenance of constraints, hierarchical code lists, and derived, composite concepts (e.g. for dissemination) requires more effort.</p>
<b>End user ("the public")</b>	<p>Consumption and reuse are more flexible in a pure data structure, but it is more difficult to identify observation keys that actually have data because of the created sparseness. (Constraints may help in this respect.) Frequent occurrences of "non applicable" values may also make data usage cumbersome.</p>

626

## 627 **4.2 Number and relations of DSDs**

628 There may not be a generic solution for the simple observation keys vs. pure concepts issue,  
 629 but there is a way of dealing with the one or many DSDs question. SDMX 2.1 allows the  
 630 specification of constraints in DSDs, data flow definitions, and data provision agreements.  
 631 This enables the specification of master or "umbrella" artefacts on the one hand and of  
 632 "satellite" or subset artefacts derived from those master structures via constraints on the  
 633 other hand. This applies to concept schemes, code lists, and DSDs. As mentioned before,  
 634 structure maps can be used to define (virtual) satellite DSDs by leaving the irrelevant  
 635 dimensions unmapped (instead of constraining them to a "not applicable" value).

636 If the constraints are specified at the data flow definition or data provision agreement level,  
 637 satellite DSDs are not even needed; i.e. they also are "virtual" in this case. The different data  
 638 flow definitions and/or data provision agreements all refer to the same master DSD but with  
 639 different sets of constraints. Another possibility is the definition of satellite DSDs that all refer  
 640 to the same master concept scheme and master code lists but differ in terms of constraints.

641 In general, this specification of multiple, interconnected DSDs (and/or data flows and/or  
 642 provision agreements) is recommended over the definition of (more or less) independent  
 643 DSDs, although there are a few cases where more loosely coupled or even independent  
 644 DSDs make more sense. Whether the constraints should be defined at DSD, data flow, or  
 645 data provision agreement level needs to be decided case by case depending on the  
 646 requirements of the parties involved.

647 The “one DSD” approach works best for single-domain and/or single-purpose scenarios. In  
 648 more complex scenarios, more complex approaches are more suitable. Usage of the “one  
 649 DSD” approach in a multi-domain or multi-purpose scenario actually means that one master  
 650 DSD containing all concepts, code lists, and codes relevant in any (but most likely not all)  
 651 domains and/or purposes is used by all domains and/or purposes without constraints. If a  
 652 “many pure concepts” approach is used, the DSD will be sparse and require many “not  
 653 applicable” values or structure maps.

654 In those more complex scenarios, multi-DSD approaches have more potential. The “master  
 655 DSD + satellite DSDs” approach imposes more restrictions and aims at a higher degree of  
 656 content harmonization than the more loosely coupled (or even independent) multi-DSD  
 657 approach. While the former specifies the concepts and code lists to be used by all derived  
 658 DSDs, the latter is more flexible. Therefore, the master + satellites approach is suggested for  
 659 data exchange scenarios with a high degree of harmonization / standardization required  
 660 such as at the international level or between national and international organizations. Please  
 661 note that what is termed “master DSD + satellite DSDs” approach here may also be  
 662 implemented as master DSD plus constrained data flows with or without using structure  
 663 maps.

664 Even in the multiple independent DSDs approach, sharing of concepts and code lists by  
 665 reference is recommended. This may be problematic if additional codes are needed by  
 666 certain DSDs, as neither the addition of codes to a code list used by reference nor the  
 667 concatenation of multiple code lists included by reference is supported by the current SDMX  
 668 Technical Standards. The only way of implementing “combined” code lists by reference is to  
 669 reference each single code from each relevant partial code list.

670 Independent DSDs are better suitable for exchange scenarios with less harmonization  
 671 required, e.g. bilateral exchange at the national level. This approach also works well for data  
 672 dissemination to end-users. DSDs may be created at the time of retrieval and only contain  
 673 concepts, code lists, and codes for which data are actually available (and which are not “not  
 674 applicable”).

675 Advantages and disadvantages of the three different structuring approaches also differ  
 676 depending on the level of data exchange. Table 7 gives a brief summary.

677 **Table 7. Data structuring approaches by level of data exchange**

Level of data exchange	Data structuring approach		
	one DSD	master + satellite DSDs	multiple, indep. DSDs
<b>within organization</b>	best for single-domain, single-purpose  can be created on the fly from structured databases	use if harmonization is important in covered domains or purposes  or if such a set of DSDs is already available at international level	easier to do than master + satellite approach  each domain/purpose can maintain DSDs independently  can be created on the fly from structured databases
<b>between national organizations</b>	the same applies as to the “within organization” scenario		

Level of data exchange	Data structuring approach		
	one DSD	master + satellite DSDs	multiple, indep. DSDs
<b>between int. organization and national organizations</b>	best for single domain, single purpose scenarios that are usually rather restricted with very clear specification of what needs to be exchanged	preferable over multi-DSD approach in case of multi-domain and/or multi-purpose scenarios with highly correlated data flows for maintenance reasons	for multi-domain and/or multi-purpose scenarios; only recommended if overlap of domains/purposes is minor (e.g. just w.r.t. cross-domain concepts)  equivalent to multiple "one DSD" solutions, one for each domain / purpose
<b>between international organizations</b>	comparable to "national to international" scenario		
<b>dissemination to public</b>	for single-domain, single-purpose cases  in more complex cases this may be the preferable approach for data discovery tools (one data structure to find and access all data)	in multi-purpose or –domain scenarios:  if it is relevant for the public to see the relationship between the data structures: use master + satellites approach  otherwise the multi-DSD option is preferable, although with the highest possible degree of re-use of code lists and concepts  in both cases: important to include only concepts, code lists, and codes actually available / used by the data	

678

679 In general, finding the "perfect" data structure is less important for bilateral data exchange.  
 680 Independent, custom-tailored DSDs may do the job quite well, as harmonization and  
 681 standardization are typically not of high importance. If the data exchange is just a part of a  
 682 more comprehensive scenario (e.g. multi-purpose, multi-domain, gateway, or data-sharing  
 683 scenarios), a master DSD with satellite DSDs is preferable.

684 Table 8 outlines the pros and cons of the three approaches from the point of view of different  
 685 roles in the data exchange.

686 **Table 8. Data structuring approaches by role in data exchange**

Role in data exchange	One DSD vs. master + satellite DSDs vs. multiple, indep. DSDs
<b>Data provider</b>	It is easier to set up a data submission process against a single DSD (= less initial costs) than against multiple DSDs.
<b>Data collector</b>	Data validation is easier with DSDs that only cover what needs to be collected. This is achieved via constraints in the master + satellites approach or via tailor-made independent DSDs. If a single DSD is used in a multi-domain or –purpose scenario, necessary constraints can be specified in the data flow definition or data provision agreement.  Further processing of collected data is more flexible and easier if relations are transparent and code lists are shared as in the one DSD or master + satellite DSDs approaches. The "shared context" created through the master DSD increases harmonization and standardization and this way facilitates combined usage of data.



Role in data exchange	One DSD vs. master + satellite DSDs vs. multiple, indep. DSDs
<b>DSD maintenance</b>	<p>The complexity and initial costs for developing and maintaining master + satellite DSDs are higher than for independent DSDs as this involves managing constraints and managing impacts of changes in shared code lists to all DSDs.</p> <p>In the multiple independent DSDs approach, development and maintenance efforts may be distributed. This can be seen as an advantage, but on the other hand requires coordination in case the DSDs are only partially independent (i.e. share some code lists).</p>
<b>End user (“the public”)</b>	<p>For data discovery and retrieval the user needs to know what data is actually available (instead of what might be collected/disseminated with a certain data structure). This means that the potential sparseness should be hidden from the user. A reduced DSD derived from the data structure used in the background is more useful in most cases. Whether this is done via one DSD and constraints, master + satellite DSDs, or independent DSDs does not matter that much for the user.</p>

687

## 688 5 MINIMUM STRUCTURAL AND SEMANTIC 689 REQUIREMENTS

690 Although each data exchange scenario has specific requirements, especially on whether a  
691 concept needs to be a dimension, a mandatory or conditional attribute, on the attachment  
692 level of attributes, and on the attributes provided in the header of a DSD, a small set of  
693 minimum structural and semantic requirements can be defined for all scenarios.<sup>5</sup>

694 Certain concepts can be broadly agreed upon as being relevant in any data exchange,  
695 although their roles may differ between scenarios. The SDMX Content-Oriented Guidelines  
696 define many of these cross-domain concepts and, thus, should be referred to for further  
697 details on their specification.

698 In general, multi-purpose and multi-domain scenarios may require more concepts than  
699 single-purpose and/or –domain scenarios. This mainly applies to subject-matter (or domain-  
700 specific) concepts and concepts that inform about the data source, provider, or process.

701 Exchanges between organizations, especially on an international level, typically require  
702 more concepts to cover context information, as data are transferred out of their usual  
703 context, meaning that users in the new context do not have the same knowledge of the data  
704 and may need additional background information. For exchanges of data within an  
705 organization, some context information may be common (implicit) knowledge so that it does  
706 not need to be made explicit in the data structure.

707 For example, it may be obvious within the ECB that the data source of certain data is the  
708 national bank of the reporting country, or that certain data are always presented in Euros. An  
709 analogous argument can be brought forward for the exchange of data that comply with a  
710 certain (international) standard. In order to specify particular methodological aspects, it may  
711 be sufficient to refer to that standard (e.g., the BPM6, SNA2008) for a user familiar with the  
712 standard. But even in the two examples given it is preferable to adhere to the  
713 recommendations for (international) data exchange between organizations and include each

---

<sup>5</sup> For other more technical requirements such as the admissible characters in a code or label see the SDMX technical documents.

714 concept that is required for proper interpretation by someone without prior knowledge of the  
715 data.

716 Similarly, although bilateral exchanges may be more informal than gateway or data-sharing  
717 exchanges and require less context information in the DSD, making that information explicit  
718 in the DSD ensures higher transparency and sufficiency of the exchanged structural  
719 information. This means that the proposed minimum requirements for a DSD should be  
720 fulfilled regardless of the type and level of data exchange.

721 From a data provider perspective, certain pieces of information (especially the high-level  
722 attributes) may be obvious, meaning they would not need to be included in the DSD. As this  
723 information is typically of relevance to the data consumer though, it becomes a requirement  
724 to add the respective concepts to the data structure for the exchange. For example, the  
725 reference area of data provided from a national institution to an international organization is  
726 clear in the context of the data exchange, but needs to be added explicitly to the data when  
727 combined with data from other countries. It could be used as a mandatory attribute at the  
728 data flow level in the national to international data exchange DSD and as a dimension in the  
729 DSD for cross-country data dissemination at the international level. If one master DSD is  
730 used for the entire statistical business process, it will contain reference area as a dimension  
731 with constraints on that dimension in the data provision agreements from the national to the  
732 international level.

733 Depending on their more specific roles and tasks, different types of end users may require  
734 more or less attributes. A detailed DSD in the background provides the flexibility to fulfill  
735 those different needs. Finally, for an end user it is rather a matter of how the data structure is  
736 visualized and what functionality is offered than how the data are structured in the  
737 background.

738 One issue concerning functionality is that some existing SDMX web services do not support  
739 queries by attributes; only dimensions can be used for data retrieval. However, this is not a  
740 restriction of the SDMX standard, rather a decision that was made on the implementation  
741 side that should be corrected. If the query functionality offered to the user allows queries on  
742 dimensions only, concepts that are required to be available to the query mechanism need to  
743 be specified as dimensions even if they do not contribute to the identification of an  
744 observation and would hence rather be attributes. This is not conceptually clean and extends  
745 the DSD in terms of more complex observation keys and sparseness. It is not recommended  
746 to go that route and let the technology drive the design of the data structure; an adaptation of  
747 the query functionality is the preferred solution.

## 748 ***5.1 Required and recommended dimensions and attributes***

749 Table 9 lists the concepts that are considered as required at a minimum in any DSD for  
750 macro data (with two of these concepts only relevant for time series data). Table 10  
751 suggests a number of additional concepts that are considered of high relevance in certain  
752 scenarios but not as minimum requirements for all scenarios. Both tables show what kinds of  
753 questions about the data each concept helps to answer, if the concept is defined in the  
754 SDMX COG, whether a code list is recommended in the COG, and what role the concept  
755 plays in a DSD for time series (TS) or cross-sectional (CS) data, respectively.

756 Reference area and unit of measure are required concepts in DSDs for time series and  
757 cross-sectional macro data that may be represented as a dimension or a mandatory attribute  
758 depending on whether or not they are required to uniquely identify an observation or not. In  
759 terms of re-usability of DSDs and fitness for future needs it may make sense though to  
760 specify them as dimensions. Frequency is only relevant for time series and may also be

761 specified as dimension or mandatory attribute at the appropriate attachment level. Further  
 762 dimensions are time period (only for time series though; for cross-sectional data it will  
 763 typically be a mandatory attribute at the DSD level) and all domain-specific “indicator”  
 764 dimensions. Further mandatory attributes are unit multiplier, decimals, time format, and date  
 765 of last data update for both types of macro data DSDs, and adjustment and time period –  
 766 collection just for time series.

767 **Table 9. Minimum requirements for DSDs\*\***

Question	Concept	COG	Code list	Time series	Cross-section
Where?	reference area	X	revision	mand. attribute or dimension	
What?	“indicator”	-	domain	one or multiple dimensions	
How?	unit of measure	X	development	mand. attribute or dimension	
How?	unit multiplier	X	available	mandatory attribute	
How?	decimals	X	available	mandatory attribute	
How?	<i>adjustment</i>	X	development	mand. att.	not relevant
When?	time period	X	format	dimension	mand. att.
When?	time format	X	available	mandatory attribute	
When?	time period – collection	X	development	mand. att.	cond. att.
When?	data update – last update	X	time stamp	mandatory attribute	
How often?	<i>frequency</i>	X	available	mand. att. or	not relevant
How much?	observation value	-	numeric	measure	

768 \*\*Concepts in *italics* are only relevant for time series DSDs. An “X” in the COG column means the concept is  
 769 defined in the COG. Code list “development” means that the SWG will develop a code list to be recommended in  
 770 the COG; “revision” means that the code list is recommended by the COG and under revision by the SWG;  
 771 “format” means that a format is defined by another concept; “text”, “time stamp”, and “numeric” provide data types  
 772 used for uncoded concepts.

773 Suggested additional attributes for certain scenarios are observation status, confidentiality  
 774 status, and compiling agency (both types of data) as well as time series title and observation  
 775 pre-break value (time series only).

776 **Table 10. Suggested additional concepts for certain scenarios\*\***

Question	Concept	COG	Code list	TS	CS	Scenario
Who?	compiling agency	X	development	conditional (sibling)	conditional (obs. level)	data provider different from data compiler
Who?	confidentiality status – observation	X	available	mandatory (obs. level)		except dissemination
How?	observation status	X	available	conditional (obs. level)		except orig. collection
How much?	<i>observation pre- break value</i>	-	numeric	cond. (obs.)	not relevant	except orig. collection
What and how?	<i>time series title</i>	X	text	cond. (TS)	not relevant	dissemination

777 \*\* The legend of Table 9 applies to Table 10 as well. The suggested attachment level of attributes (if any) is  
 778 provided in parentheses in the TS (time series) or CS (cross-section) columns. In case an attribute does not vary  
 779 at that level in a certain use case, it should be attached at the highest possible level.

## 780 **5.2 Attribute attachment levels and definition of groups**

781 Each concept can only be used once as a dimension or an attribute in one DSD. Each  
 782 attribute must be explicitly attached to an observation, series, or group. The attachment level

783 depends on whether the value of the attribute changes by observation, observation group, or  
784 time series, or is the same for all observations. In the latter case, the attribute has to be  
785 specified at the *data flow* or *dataset* level. For some attributes described in the previous  
786 section, a certain attachment level applies, for others the attachment level depends on the  
787 data. For example, the time series title has to be attached at the time series level and the  
788 observation status at the observation level.

789 Series and groups are useful groupings of data that allow the specification of attributes for a  
790 set of observations instead of having to declare those attributes for every data point thereby.  
791 This increases the readability of an SDMX data file, reduces the size of the data file, and (in  
792 some cases) even increases the processing efficiency.

793 Series is relevant for time series data only. It refers to a group of observations that differ only  
794 with respect to the time dimension, i.e. all dimensions except time define the series  
795 attachment level. The best-known example of a group definition is the sibling group that  
796 combines time series with different frequencies. Observations in a sibling group differ with  
797 respect to frequency and time; all other dimensions are used to define the sibling group. A  
798 sibling group can be regarded as a time series group with the frequency excluded from the  
799 group definition. Any other combination of dimensions (or a single dimension) can also be  
800 used to define an observation group. An example for a group defined by a single dimension  
801 is reporting country. For instance, attributes related to methodology are often the same for all  
802 data of a country. In order to attach attributes to a group, a name for that group has to be  
803 specified.

804 The attachment levels are organized in a hierarchy with dataset as the top (most coarse)  
805 level, followed by groups and series, and observation as the lowest (most detailed) level.  
806 Attributes attached at a more detailed level can override the attribute declarations of higher  
807 level attribute declarations. For example, values specified for an attribute at the sibling level  
808 can be overridden at the series level. Attribute declarations at any group level can be  
809 overridden at the observation level.

810 When defining groups, a common-sense and trial-and-error approach may be used to work  
811 on the reduction of the file size and the increase of processing efficiency without making the  
812 data file too complex to parse and process. The use of groups is not mandatory but  
813 recommended in case of attributes that do not vary by observation to leverage the  
814 advantages described above.

### 815 **5.3 Header elements**

816 In order to exchange data using SDMX, a *message* must be created. The message includes,  
817 among other things, the data and a reference to the DSD which describes the data. The  
818 message must provide some additional administrative and descriptive information as part of  
819 the exchange. The mandatory information follows a common construct, i.e. the basic  
820 elements are standardized across different types of SDMX-ML messages (e.g. queries,  
821 structure definitions, and data). From a technical point of view, the following elements are  
822 required for an SDMX message that contains a DSD or a dataset:

- 823 - *ID*: a unique identifier of the message
- 824 - *Test*: a Boolean attribute that indicates whether the message is for test purposes or  
825 not
- 826 - *Prepared*: the date the message was prepared
- 827 - *Sender*: the identification of the organization that is transmitting the message  
828 (recommended: code from the agency code list in the SDMX COG)

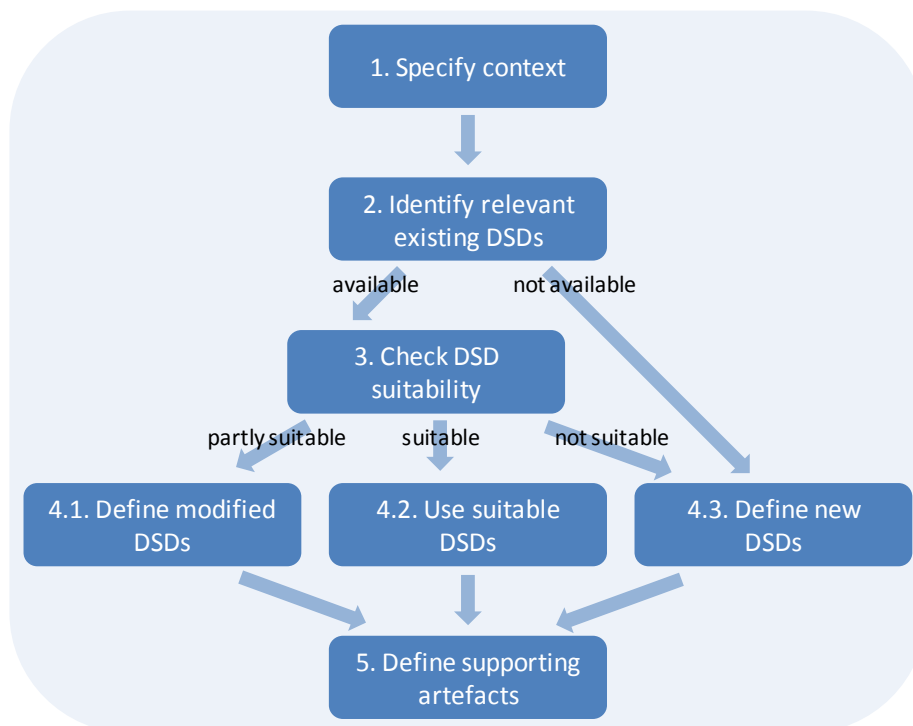
829 From a business perspective, the inclusion of the *Name* element is highly recommended, as  
 830 it can help to understand the purpose of the exchange message. Other header elements  
 831 such as *Receiver* are optional.

## 832 6 STEP-BY-STEP GUIDE

833 As a more practical guide to the design of SDMX Data Structure Definitions, this section  
 834 presents a summary of the DSD design process and the aspects to be considered at each  
 835 process step.

### 836 6.1 High-level overview of the process

837 Figure 1 provides an overview of the overall process. As a first step, the context of the data  
 838 exchange(s) that should be covered by the DSD(s) is defined in terms of purpose, domains,  
 839 level of exchange, type of data, type of recipient, role of in data exchange, process pattern,  
 840 and GSBPM phase (see Figure 2). Since reusing existing artefacts is one of the guiding  
 841 principles, the second step identifies existing DSDs that may be reused (see Figure 3). In  
 842 case relevant DSDs are available, their suitability in the present context is evaluated in step  
 843 3. Aspects to be taken into account are concept coverage, concept roles, attribute  
 844 attachment levels, and code lists (see Figure 4). Step 4 is subject to the outcome of step 3.  
 845 In case of a favorable assessment, the DSDs are simply reused. If the DSDs are partly  
 846 suitable, modified versions can be derived. See section 2. for a summary of possible DSD  
 847 modification scenarios. If the DSDs are not suitable or if no relevant DSDs are available at  
 848 all, new DSDs will be defined as described in section 3. Finally, supporting artefacts such as  
 849 data flow definitions and data provision agreements are defined (see Figure 5).



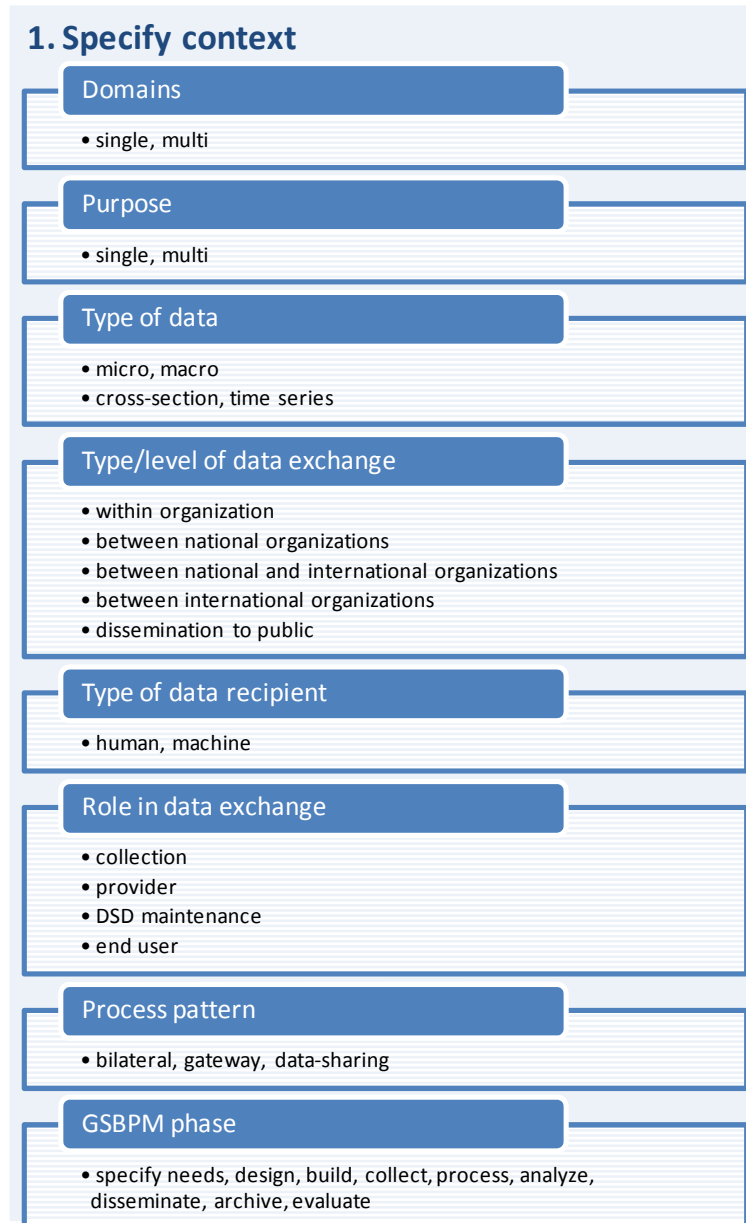
850

851

852 **Figure 1. Overview of the DSD design process**

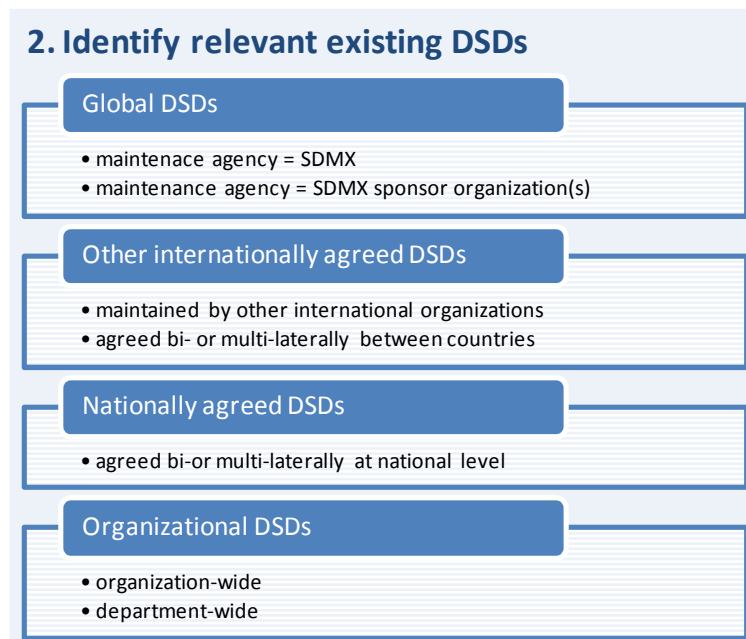
852

853 Figure 2 summarizes the characteristics of the data exchange context that is defined in step  
 854 1. These characteristics affect the decision on the data structuring approach that is part of  
 855 the process of defining the concepts of a new DSD (step 4.3. in Figure 1; see Figure 7 in  
 856 section 2.).



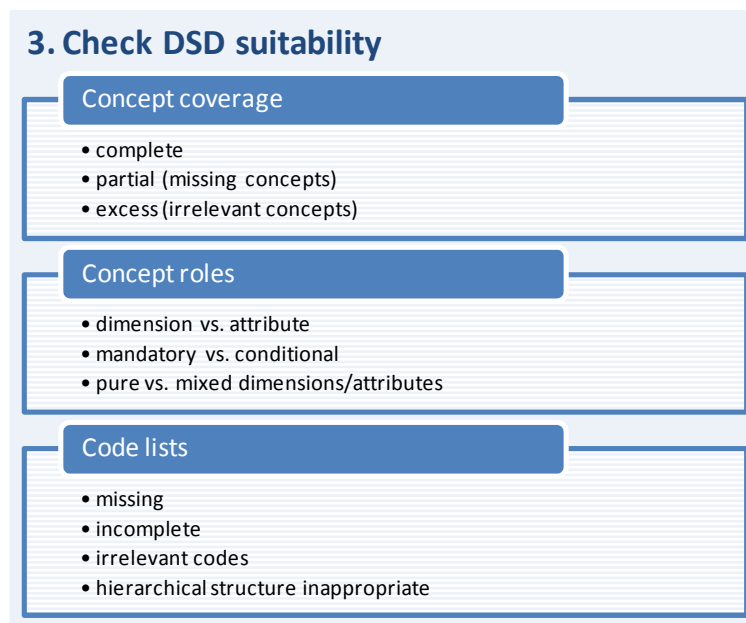
857  
 858 **Figure 2. Characteristics of data exchange context**  
 859

860 Figure 3 recaps the priorities given to different types of existing DSDs when searching for  
 861 candidates for reuse in step 2. Global DSDs maintained by the SDMX consortium are ranked  
 862 the highest. They can be found via the Global SDMX Registry.



863  
 864 **Figure 3. Priority ranking of existing DSDs for reuse**

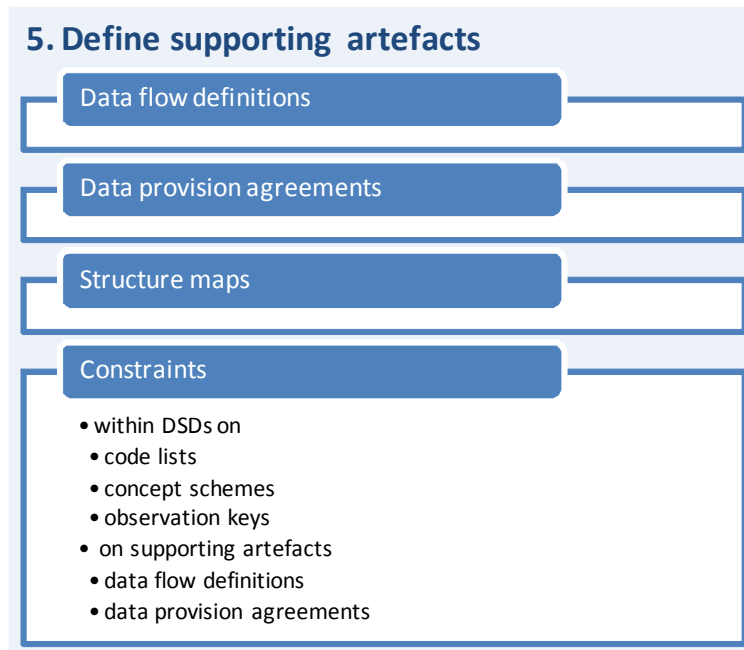
865 Figure 4 summarizes the aspects to be considered in the assessment of the suitability of  
 866 existing DSDs in step 3. For a detailed description of the cases of partial unsuitability see  
 867 section 2.1. above.



868  
 869 **Figure 4. Aspects of DSD suitability**

870

871 Figure 5 lists the most relevant artefacts required in addition to a DSD, its concept scheme,  
872 and code lists.



873

874

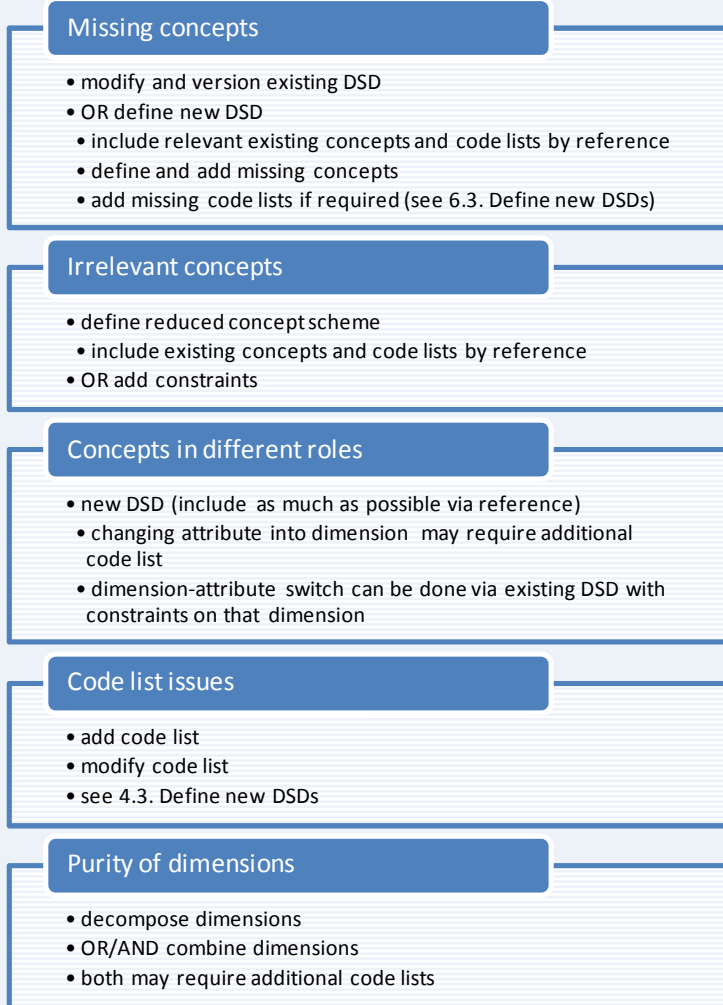
**Figure 5. Supporting artefacts**

## 875 **6.2 Defining modified DSDs**

876 Figure 6 briefly recapitulates the actions that can be taken to overcome partial unsuitability of  
877 DSDs. As far as possible, existing artefacts should be reused in this case. This means that  
878 even if a DSD cannot be reused as a whole, concepts and code lists from that DSD can be  
879 included in the new DSD by reference.



#### 4.1. Define modified DSDs

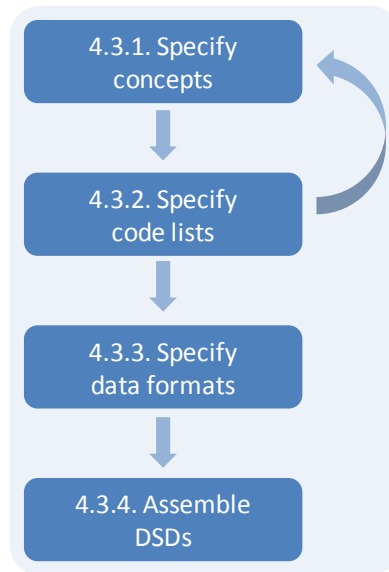


880  
881

**Figure 6. DSD modification scenarios**

#### 882 **6.3 Defining new DSDs**

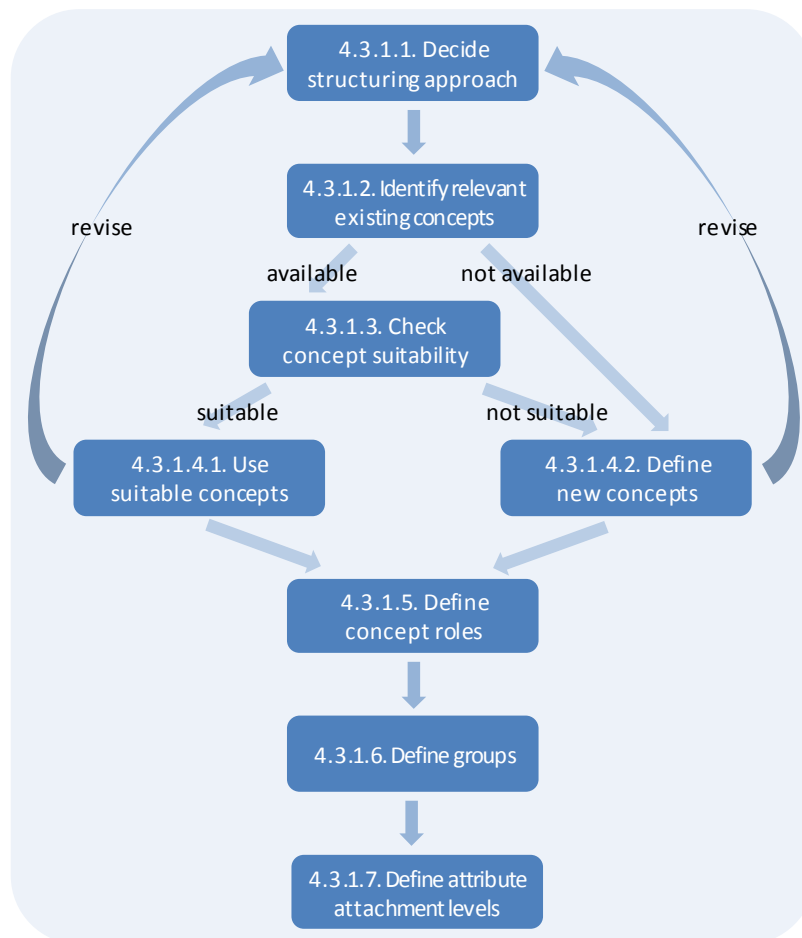
883 In case no (suitable) DSD is available, the actual process of specifying a new DSD is  
884 started. Figure 7 depicts this process (step 4.3. in Figure 1). It encompasses the  
885 specification of concepts, code lists, and data formats. All three specification steps include  
886 the identification of already existing artefacts that could be reused or modified to satisfy the  
887 requirements at hand and the definition of new artefacts in case no suitable artefacts are  
888 detected. Several iterations of steps 1 (specification of concepts; see Figure 8) and 2  
889 (specification of code lists; see Figure°13) may be necessary, including revisions of the  
890 decision concerning the data structuring approach. Finally all artefacts defined in the  
891 previous steps are put together into a DSD.



892  
893

**Figure 7. New DSD specification process**

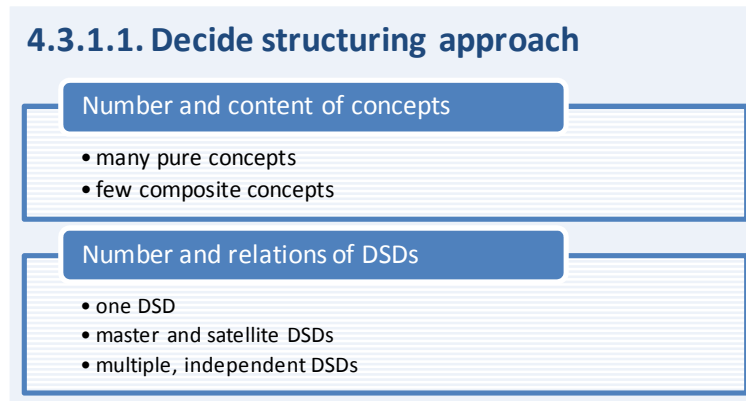
894 Figure 8 outlines step 4.3.1, the process of concept specification. It covers the decision on  
895 the structuring approach, the identification of relevant concepts and the assessment of their  
896 suitability, the definition of new concepts, concept roles, and attribute attachment levels.



897  
898

**Figure 8. Concept specification process**

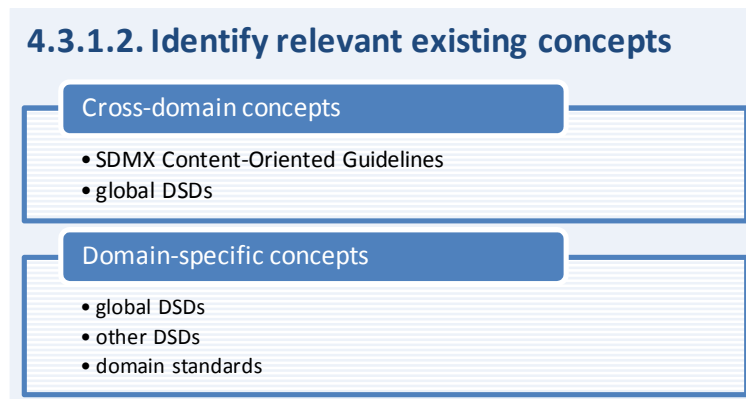
899 Both, the decision on reuse of existing concepts as well as the definition of new ones, may  
 900 lead back to a revision of the data structuring approach. For example, it could turn out that a  
 901 certain concept needs to be broken down further which may lead from a “few composite  
 902 dimensions” to a “many pure dimensions” approach. Figure 9 provides the design options  
 903 involved in the decision on a data structuring approach. The options are defined in terms of  
 904 the number of DSDs and the number of concepts (especially dimensions). The reasonability  
 905 and feasibility of these options depend on the context of the present data exchange(s) as  
 906 defined in the first step of the overall design process and on the content of the data  
 907 exchange with respect to concepts.



908  
909

**Figure 9. DSD design options**

910 In the second step of new DSD design, relevant existing concepts are identified. Figure 10  
 911 indicates potential sources of those concepts such as the SDMX COG for cross-domain  
 912 concepts, global or other DSDs as already identified earlier in the process, and domain  
 913 standards such as the UN’s System of National Accounts Manual 2008 for domain-specific  
 914 concepts.

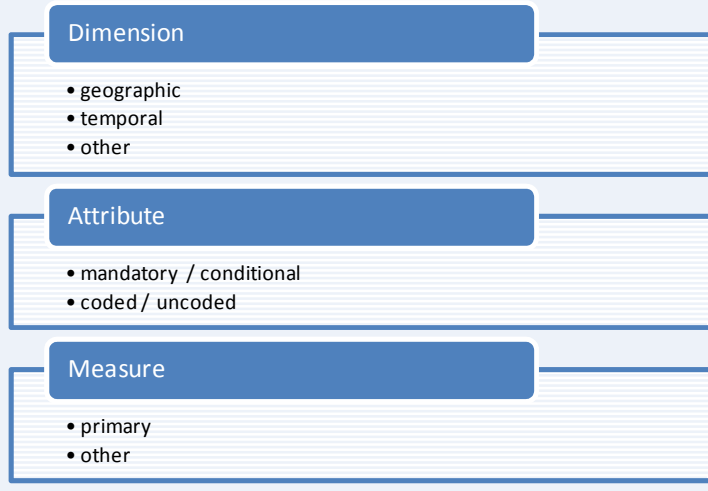


915  
916

**Figure 10. Potential sources of concepts and definitions**

917 The definition of new concepts (step 4.3.1.4.2.) is necessary if no (suitable) concept can be  
 918 reused. It entails giving each concept a name, a code, and a definition. Further details about  
 919 the usage of the concepts in the DSD are specified in steps 4.3.1.5. (concept roles), 4.3.1.6.  
 920 (dimension groups), and 4.3.1.7. (attribute attachment levels). Figure 11 and 12 summarize  
 921 the possible concept roles and attribute attachment levels.

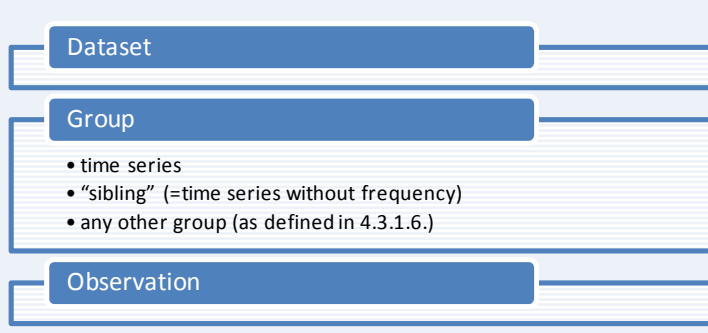
#### 4.3.1.5. Define role of concepts



922  
923

**Figure 11. Possible concept roles**

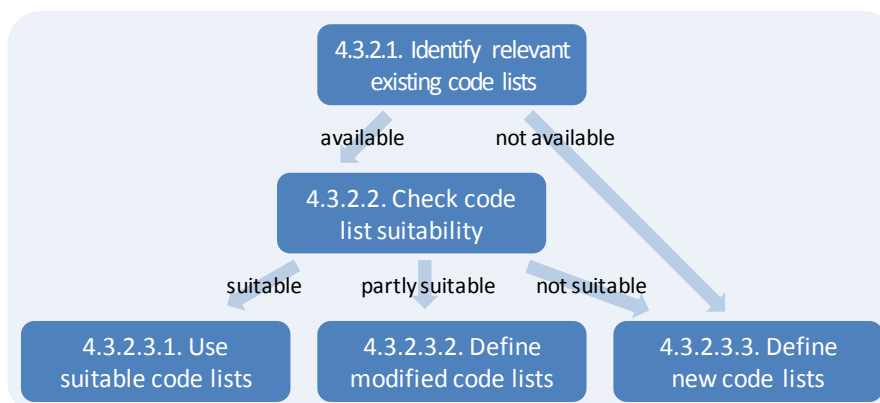
#### 4.3.1.7. Define attachment level of attributes



924  
925

**Figure 12. Possible attribute attachment levels**

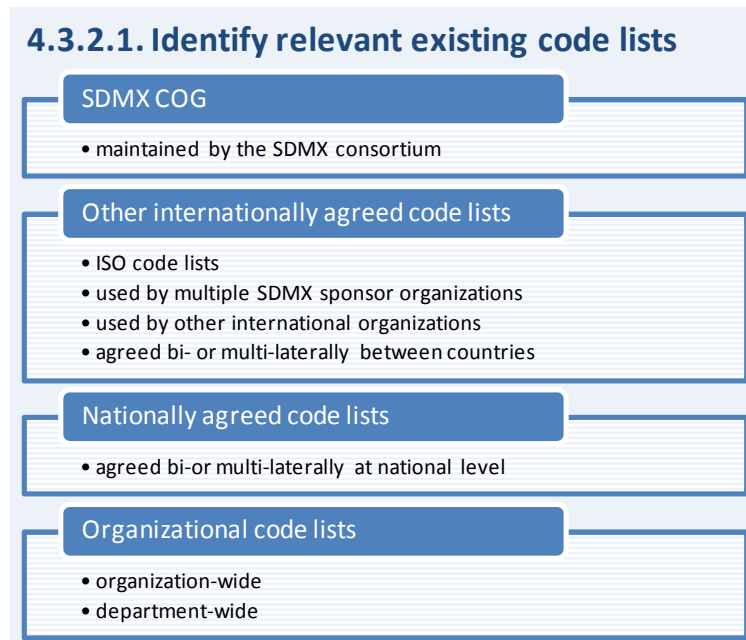
926 The second step in the process of defining a new DSD is the specification of code lists for all  
 927 coded concepts. All dimensions must be coded (with time being an exception to this rule);  
 928 attributes may be coded. For uncoded concepts, a data format has to be specified. Existing  
 929 formats may be reused or new ones defined. An example is the time format that is specified  
 930 in the SDMX COG. Figure 13 illustrates the code list specification process. If no relevant and  
 931 suitable code list exists, a new one will be defined or a partially suitable one will be adapted  
 932 (see Figure 16). Suitable code lists can simply be reused via reference.



933  
934

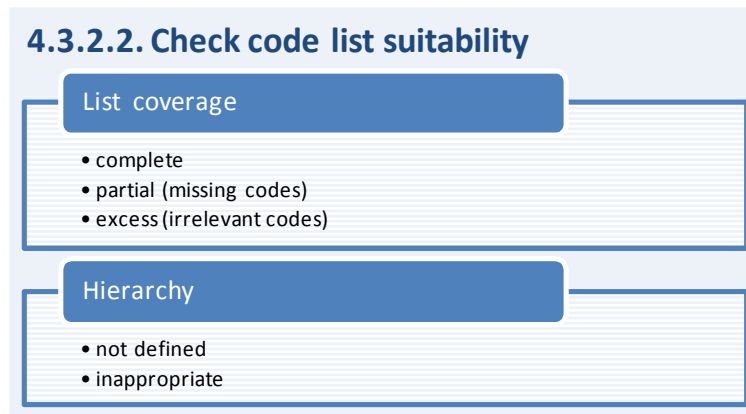
**Figure 13. Code list specification process**

935 Figure 14 recaps the priorities given to different types of existing code lists when searching  
 936 for candidates for reuse (step 4.3.2.1.). Code lists recommended by the SDMX COG (and  
 937 maintained by the SDMX consortium) are ranked the highest.



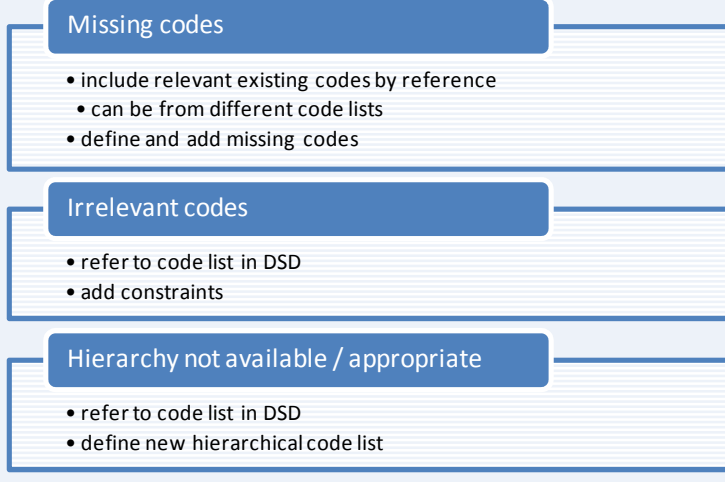
938  
 939 **Figure 14. Priority ranking of existing code lists for reuse**

940 Figure 15 summarizes the aspects to be considered in the evaluation of the suitability of  
 941 existing code lists (step 4.3.2.2.). Figure 16 summarizes the scenarios of adapting existing  
 942 code lists that do not fully meet the specified needs (step 4.3.2.3.2). For a detailed  
 943 description of the cases of partial unsuitability see section 2.1. above.



944  
 945 **Figure 15. Aspects of code list suitability**

#### 4.3.2.3.2. Define modified code lists



946

947

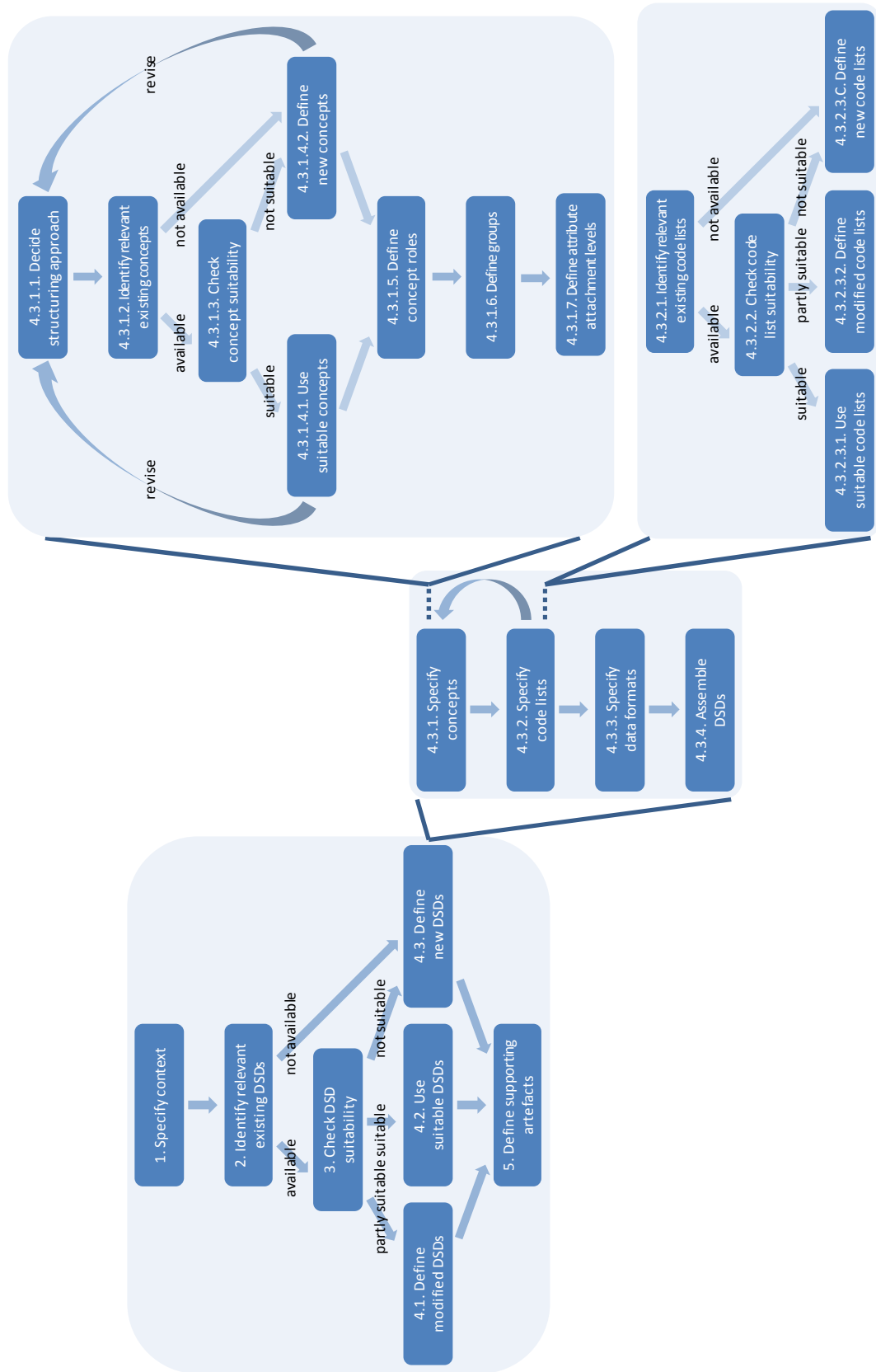
**Figure 16. Code list modification scenarios**

948 If new code lists need to be defined, please refer to the SDMX Guidelines for the creation of  
 949 code lists. Basically, code, name, and definition need to be provided for each item in the  
 950 code list. In addition, hierarchical code lists may be defined.

951 The final step in defining a new DSD covers the assembly of all components specified during  
 952 the process, i.e. concept scheme(s), code lists, data formats, concept roles, attribute  
 953 attachment levels, and the assignment of code lists and data formats to concepts.

#### 954 **6.4 Checklist for DSD Designers**

955 Figure 17 provides an overview of all steps in the DSD design process as described in the  
 956 previous subsections 1. to 3. Figure 18 compiles those steps into a checklist for DSD  
 957 designers to help them make sure all aspects are considered.



958  
959

Figure 17. DSD design process

1. Specify context
2. Identify relevant existing DSDs
3. Check DSD suitability
4. 1. If DSDs partly suitable: Define modified DSDs
4. 2. If DSDs suitable: Use them
4. 3. If DSDs not suitable or not available: Define new DSDs
4. 3. 1. Specify concepts
  4. 3. 1. 1. Decide DSD structuring approach
  4. 3. 1. 2. Identify relevant existing concepts
  4. 3. 1. 3. Check concept suitability
  4. 3. 1. 4. 1. If suitable: Use concepts
  4. 3. 1. 4. 2. If not suitable or not available: Define new concepts
  4. 3. 1. 5. Define concept roles
  4. 3. 1. 6. Define groups
  4. 3. 1. 7. Define attribute attachment levels
4. 3. 2. Specify code lists
  4. 3. 2. 1. Identify relevant existing code lists
  4. 3. 2. 2. Check code list suitability
  4. 3. 2. 3. 1. If suitable: Use code lists
  4. 3. 2. 3. 2. If partly suitable: Define modified code lists
  4. 3. 2. 3. 3. If not suitable or not available: Define new code lists
4. 3. 3. Specify data formats
4. 3. 4. Assemble DSDs
5. Define supporting artefacts

960

961

Figure 18. Checklist for DSD design process

## 962 7 ANNEX 1. GLOSSARY OF TERMS

963 **This glossary was removed in February 2014 because the system of SDMX glossaries**  
 964 **is being reviewed with the purpose of producing one centralised glossary**  
 965 **encompassing the whole SDMX terminology.**

## 966 8 ANNEX 2. WHAT IS A DSD?

967 *Data and metadata structure definitions* (DSDs & MSDs, respectively) associate *statistical*  
 968 *concepts* with their value domains and assign concept roles. Concepts are defined in a  
 969 *concept scheme*; value domains are specified as a *code list* or by a *data type/format*. For  
 970 example, the domain of a concept with a categorical representation such as “industrial  
 971 sector” is specified by a code list that provides the values for the “industrial sector”.  
 972 *Hierarchical code lists* can be used to specify parent-child relationships between (a subset  
 973 of) the codes of a (flat) code list. For concepts without categorical representation, for  
 974 instance “reference period”, the value domain can be defined by specifying a time format, a  
 975 numeric data type, or a text format. Such a format determines the structure of the admissible  
 976 values instead of explicitly enumerating them. Concept schemes and code lists do not have  
 977 to be defined together with a DSD; they are usually included by reference.

978 Concepts assume different roles in a data structure definition:



- 979 - *dimensions* are required to uniquely identify an observation (a data value); e.g., for  
980 time series, at least one geographic, one temporal, and one (“mixed”) subject-matter  
981 dimension are required to identify a data value (for instance: reference area =  
982 Mexico, time = 2002, indicator = GDP nominal, US\$)<sup>6</sup>;
- 983 - *measures* are the containers of the actual observation or data values;
- 984 - *attributes* provide additional meta-information required to interpret the data correctly  
985 but not to identify the observations; for instance, data for the same observation  
986 defined by a value combination of the dimensions (also termed “key”) will usually only  
987 be provided for one unit multiplier, e.g. in millions; hence unit multiplier is not  
988 necessary to identify an observation, but it is still required for a proper interpretation.  
989 Attributes can be defined as mandatory or not mandatory, and they can be attached  
990 at different levels, e.g. at observation level or at the level of groups defined by the  
991 value combinations of a predefined subset of dimensions (for example reporting  
992 currency may be attached at the country level).

993 Data are exchanged according to a *data flow definition*. A data flow definition identifies the  
994 DSD that defines the structure of the data exchanged, may be associated with a subject-  
995 matter domain, and may contain *constraints* that further restrict the admissible keys and thus  
996 the coverage of the data flow. A data provider may provide data for multiple data flows and  
997 multiple data providers may contribute to one and the same data flow. *Provision agreements*  
998 specify which data providers supply what data to which data flows. The agreements may  
999 contain a reporting or publishing calendar, *constraints* on the code lists and/or keys to define  
1000 what subset of the data flow is contributed, and the temporal coverage of the data provided.  
1001 For example, constraints may restrict the reference area dimension to a specific subset that  
1002 is provided by a certain data provider. The actual source of data is also stated in a provision  
1003 agreement in terms of a URL.

1004 To briefly summarize the above: A DSD requires a concept scheme and code lists that can  
1005 be assigned to concepts. It defines the roles of the concepts and the value domains of the  
1006 dimensions, attributes, and measures. It can use constraints to restrict the admissible set of  
1007 codes from the referenced code list(s) or the admissible observation keys. It is used in the  
1008 definition of a data flow and plays a major role in data exchange by providing a “shared  
1009 language”. A data flow definition may also use constraints to further restrict the data that  
1010 may be exchanged in that data flow. A data provision agreement defines who (= data  
1011 provider) provides what (= part of data flow defined by constraints on the DSD) to a data  
1012 flow. For more details on SDMX artefacts see Section 2 of the SDMX Standards.

## 1013 **9 ANNEX 3. REFERENCES**

### 1014 **9.1 SDMX Documents**

1015 The SDMX documents referred to in these guidelines as well as the complete technical  
1016 specification of the SDMX Technical Standard 2.1 (and earlier versions) are available online  
1017 at <http://sdmx.org/>. The SDMX documents currently under development by the Statistical  
1018 and Technical Working Groups will also be made available on the SDMX website.

#### 1019 **9.1.1 Existing documents**

1020 SDMX Content-Oriented Guidelines 2009 (5 Annexes).

---

<sup>6</sup> Please note that this is not a recommendation to always have three dimensions only. This is just a simplified example.

- 1021 SDMX Standards: Section 2 - Information Model: UML Conceptual Design.
- 1022 SDMX Standards: Part 5 - SDMX Registry Specification: Logical Functionality and Logical  
1023 Interfaces.
- 1024 SDMX Standards: Section 6 - Technical Notes Version 2.1.
- 1025 **9.1.2 Not yet available documents**
- 1026 Business Requirements and Use-Cases for the SDMX Global Registry.
- 1027 SDMX Guidelines for the Creation and Maintenance of Code Lists.
- 1028 SDMX Guidelines for the Governance and Maintenance of Artefacts.
- 1029 SDMX Guidelines for the Design of MSDs.
- 1030 **9.2 Non-SDMX Documents**
- 1031 6th Edition of the IMF's Balance of Payments Manual (BPM6). Available online at  
1032 <http://www.imf.org/external/pubs/ft/bop/2007/bopman6.htm>.
- 1033 METIS: Generic Statistical Business Process Model (GSBPM). Available online at  
1034 <http://www1.unece.org/stat/platform/display/metis/The+Generic+Statistical+Business+Process+Model>.
- 1035 UN's System of National Accounts Manual 2008 (SNA2008). Available online at  
1036 <http://unstats.un.org/unsd/nationalaccount/sna2008.asp>.