ICS 35.240.01
A11
备案号：

JR

中 华 人 民 共 和 国 金 融 行 业 标 准

JR/T 0107.7—2014/ISO 17369-7:2013

统计数据和元数据交换（SDMX）第7部分：Web服务用法指南

Statistical data and metadata exchange (SDMX)—
Part 7: Guidelines for the use of web services

（ISO 17369-7:2013, guidelines for the use of web services, IDT）

2014 - 08 - 28 发布　　2014 - 08 - 28 实施

中国人民银行　　发布

目　　次

前　言

JR/T 0107《统计数据和元数据交换（SDMX）》分为七个部分：

——第1部分：框架；

——第2部分：信息模型　UML概念设计；

——第3部分：SDMX-ML　模式和文档；

——第4部分：SDMX-EDI　语法和文档；

——第5部分：注册表规范　逻辑功能和逻辑接口；

——第6部分：SDMX技术说明事项；

——第7部分：Web服务用法指南。

本部分为JR/T 0107的第7部分。

本部分依据GB/T 1.1-2009规则起草。

本部分等同采用ISO 17369-7：2013《统计数据和元数据交换（SDMX） 第7部分：Web服务用法指南》。

本部分由中国人民银行提出。

本部分由全国金融标准化技术委员会（SAC/TC180）归口。

本部分主要起草单位：中国人民银行调查统计司、中国金融电子化公司。

本部分主要起草人：盛松成、徐诺金、姚力、巴运红、任全忠、潘润红、李曙光、韩建国、贾树辉李兴锋、王媛、司燕翔、刘蔚、张艳、吴永强、邓琳莹。

引　言

统计数据和元数据交换（SDMX）标准由SDMX国际组织发起并提出。SDMX国际组织是由国际清算银行（BIS）、经济合作与发展组织（OECD）、欧盟统计局（Eurostat）、欧洲中央银行（ECB）、国际货币基金组织（IMF）、联合国（UN）和世界银行（WB）七个国际组织联合建立，其制定发布的《统计数据和元数据交换》标准规定了统计人员在采集、处理和交换统计数据时所使用的统计概念和方法，规范了对外披露统计信息时统计数据的机构范围、地理区域、存流量性质、时间属性、频度以及文件格式等内容。

SDMX 标准提供了统计数据及元数据交换和共享的标准化格式，可以达到更好地扩展和高效率使用的目的。目前SDMX标准主要应用领域为部分国家中央银行和统计部门。本部分作用在于规范我国金融统计标准体系的内部处理和对外发布，促进金融统计的互联互通、信息共享和业务协同，提高信息共享的效率，满足金融综合统计的需要。

本部分中Web服务代表当代互联网技术。该服务允许计算机应用程序直接在互联网进行数据交换，特别是允许以比过去更加灵活的方式进行模块或分布式计算。然而，为使得Web服务发挥其功能，需要很多标准，这些标准用于申请和提供数据、用于表示用来打包交换数据的封装数据及用于Web服务的互相描述，可以使Web更容易地集成到使用其他的网络服务作为数据资源的应用程序中。

SDMX 关注于使用互联网技术的数据交换，将提供部分关于统计数据和元数据的标准。然而，很多Web服务标准已经存在，且无需为使其特别应用于统计领域而重新修订。特别是，SDMX能够使用简单对象访问协议（SOAP）和Web服务描述语言（WSDL）对需要标准化的数据和元数据交换格式进行补充。在Web 服务界，经常使用基于URL语义调用Web服务的REST（表述性状态转移）协议，这种基于REST的服务可以用Web应用描述语言（WADL）标准方式描述，与基于SOAP的XML调用Web服务使用WSDL描述方法一致。

尽管存在SOAP协议和WSDL，但在实施中已发现，实际上各供应商提供的执行程序并不能互用。为此才开展了Web服务-互用性（WS-I）倡议。该倡议由一些全部以同样方式、采用相同Web服务标准的供应商发起，且已通过互用性测试进行了验证。这些供应商出版了说明文件，来描述如何交互使用Web服务标准。SDMX酌情使用WS-I成果以满足统计领域的需求。

SDMX为这些使用标准提供了指南，这种方式提升了SDMX Web服务之间的互用性。允许通用客户应用程序的开发，该程序能够同实施这些指南的任一SDMX Web服务进行有意义的交流。

本部分为资料性的,其目的不在于给出将SDMX-ML文件和Web服务标准应用于统计数据和元数据交换的最佳实践惯例。本部分描述了XML格式的SDMX WSDL和WADL，这些规范性文件将APIs正规化。

统计数据和元数据交换（SDMX）
第7部分：Web服务用法指南

1 范围

本部分规定了统计数据和元数据交换的Web服务用法，包括Web服务和SDMX-ML、基于SOAP的SDMX Web服务：WSDL操作和行为、SDMX RESTful API、SDMX Web服务标准错误及示例附件等内容。

本部分适用于统计数据和元数据的交换和共享。

2 规范性引用文件

下列文件对于本文件的应用是必不可少的。凡是注日期的引用文件，仅注日期的版本适用于本文件。凡是不注日期的引用文件，其最新版本（包括所有的修改单）适用于本文件。

WS-I简介文件 Basic Profile Version 1.1

3 Web服务和SDMX-ML

传统应用程序和服务习惯上通过应用程序接口（APIs）实现其功能。Web服务也是如此，其提供了函数调用的公共版本，在网络上可通过使用Web服务协议（SOAP或REST）访问这些函数调用。为使得多个Web服务互用，必须基于这些公共函数建立标准抽象概念或模型。SDMX得益于具有公共信息模型，并且使用SDMX信息模型作为标准Web服务函数调用的基础，是一种很自然的扩展。

Web服务以XML格式交换数据，即不同Web服务中传递的数据是如何格式化的。SDMX-ML，作为统计领域内数据和结构元数据交换的标准XML，为Web服务数据的公共序列化提供了可用的XML格式。对于简单Web服务数据交换，存在一些经常使用的技术[远程过程调用（RPCs）]，一系列基于公共信息模型的XML交换，是实现互用性的更佳方法。

SDMX-ML存在几种不同的文件类型，所有文件类型对于SDMX Web服务的生成方和用户都非常重要。

a) “结构”报文：该类报文描述了概念、数据和元数据结构定义和代码表，这些内容定义了统计数据和参考元数据的结构。每个SDMX符合性数据集或元数据集必须具有一个对其进行描述的相应的数据或元数据结构定义对其进行描述。当请求XML说明时，必须能够从SDMX Web服务中获得。

b) “通用”数据报文：该报文是标记SDMX数据集的“通用”方式。该方案描述了SDMX数据交换的非特定数据结构定义格式，且该方案要求每个SDMX Web服务至少使其数据在该格式中可用可以预见，并行服务也将支持其他特定数据结构定义的XML格式来表示数据。

c) “特定结构”数据报文：该报文是一种标准模式格式，来源于标准化映射和诸多标准标记的结构描述。该报文仅针对特定数据结构定义的结构，因此每个数据结构定义将具有自己的“特定结构”方案。该报文用于大数据集的交换，Web服务是否提供的数据格式，取决于其数据交换的要求。

d) “查询”报文：该报文用于产生基于SOAP的SDMX Web服务。所有报文与主模板保持一致，但被分解为特定查询，以便每个服务仅支持对模板报文有意义的领域。该查询报文在所有数据

和元数据结构定义中通用，需要按照具体结构概念（见结构说明）规定的值进行查询。该报文允许用户查询数据、概念、代码表、数据和元数据结构定义。

e) “注册接口”报文：所有注册接口（Registry Interfaces）都是该 SDMX-ML 报文类型的子元素。这些接口将在 SDMX 注册表规范中详细描述。

f) “通用”元数据报文：该报文是用于报告参考元数据概念，该报文通常与所有类型的参考元数据结构说明通用。

g) “特定结构”元数据报文：该报文用于向特定的元数据结构定义报告参考元数据概念。

4 基于 SOAP 的 SDMX Web 服务：WSDL 操作与行为

4.1 介绍

本部分针对基于SOAP Web 服务的操作和行为。最重要的是标准WSDL操作列表，它是标准WSDL XML 实例的基础，并与该实例一起应用于发展软件包。本部分还为支持互用性实施的Web 服务提供一些指南。

所有SDMX SOAP Web 服务都可以使用WSDL实例进行描述。每次交换回复内容里，都应包括SDMX中每个XML数据和元数据格式的全局元素。以下给出了每个已识别模式的函数名称以及SDMX-ML有效载荷数据的类型。

由于不支持SOAP RPC，每个函数的“参数”仅为适当SDMX-ML报文类型的实例。注意，<wsdl:import>应用于规定多重报文交换的方案。分布式WSDL文件说明了SOAP报文的使用方法。

4.2 SDMX Web 服务命名空间

SDMX Web 服务命名空间[1)]包含一系列针对基于SOAP服务使用的报文。每个操作都会产生一个Web 服务报文和响应报文。在每个案例中，为了适应操作实施，对其他SDMX报文进行改良。此外，错误代码表在SOAP封套中使用（见SDMX报文中的错误元素范例部分）。

4.3 WSDL 操作支持

即使该支持很小并仅包含解释未实现申请操作错误的产生，SDMX Web服务也必须支持所有已列明的操作，这对互用性很有必要。

4.4 WSDL 操作表

SOAP和WSDL使用的操作表遵循1.1版Web服务互用性规范。

4.4.1 数据

相关内容如下：

——GetStructureSpecificData：通过 GetStructureSpecificDataRequest 报文调用该操作，接收 GetStructureSpecificDataResponse 作为反馈。

——GetGenericData：通过 GetGenericDataRequest 报文调用该操作，接收 GetGenericDataResponse 作为反馈。

——GetStructureSpecificTimeSeriesData：通过 GetStructureSpecificTimeSeriesDataRequest 报文调用该操作，接收 GetStructureSpecificTimeSeriesDataResponse 作为反馈。

——GetGenericTimeSeriesData：通过 GetGenericTimeSeriesDataRequest 报文调用该操作，

1) 例如，SDMX WSDL 定义的公开命名空间。

接收 GetGenericTimeSeriesDataResponse 作为反馈。

4.4.2 MetaData

相关内容如下：

——GetGenericMetadata：通过 GetGenericMetadataRequest 报文调用该操作，接收 GetGenericMetdataResponse 作为反馈。

——GetStructureSpecificMetadata：通过 GetStructureSpecificRequest 报文调用该操作，接收 GetStructureSpecificResponse 作为反馈。

4.4.3 结构用法

相关内容如下：

——GetDataflow：通过 GetDataflowRequest 报文调用该操作，接收 GetDataflowResponse 作为反馈。

——GetMetadataflow：通过 GetMetadataflowRequest 报文调用该操作，接收 GetMetadataflowResponse 作为反馈。

4.4.4 结构

相关内容如下：

——GetDataStructure：通过 GetDataStructureRequest 报文调用该操作，接收 GetDataStructureResponse 作为反馈。

——GetMetadataStructure：通过 GetMetadataStructureRequest 报文调用该操作，接收 GetMetadataStructureResponse 作为反馈。

4.4.5 项目方案

相关内容如下：

——GetCategoryScheme：通过 GetCategorySchemeRequest 报文调用该操作，接收 GetCategorySchemeResponse 作为反馈。

——GetConceptScheme：通过 GetConceptSchemeRequest 报文调用该操作，接收 GetConceptSchemeResponse 作为反馈。

——GetCodelist：通过 GetCodelistRequest 报文调用该操作，接收 GetCodelistResponse 作为反馈。

——GetHierarchicalCodelist：通过 GetHierarchicalCodelistRequest 报文调用该操作，接收 GetHierarchicalCodelistResponse 作为反馈。

——GetOrganisationScheme：通过 GetOrganisationsSchemeRequest 报文调用该操作，接收 GetOrganisationSchemeResponse 作为反馈。

——GetReportingTaxonomy：通过 GetReportingTaxonomyRequest 报文调用该操作，接收 GetReportingTaxonomyResponse 作为反馈。

4.4.6 其他可维护工具

相关内容如下：

——GetStructureSet：通过 GetStructureSetRequest 报文调用该操作，接收 GetStructureSetResponse 作为反馈。

——GetProcess：通过 GetProcessRequestt 报文调用该操作，接收 GetProcessResponse 作为

反馈。

——GetCategorisation：通过 GetCategorisationRequest 报文调用该操作，接收 GetCategorisationResponse 作为反馈。

——GetProvisionAgreement：通过 GetProvisionAgreementRequest 报文调用该操作，接收 GetProvisionAgreementResponse 作为反馈。

——GetConstraint：通过 GetConstraintRequest 报文调用该操作，接收 GetConstraintResponse 作为反馈。

4.4.7 XML 模式（XSD）

相关内容如下：

——GetDataSchema: 通过 GetDataSchemaRequest 报文调用该操作，接收 GetDataSchemaResponse 作为反馈。

——GetMetadataSchema: 通过 GetMetadataSchemaRequest 报文调用该操作，接收 GetMetadataSchemaResponse 作为反馈。

4.4.8 结构元数据通用查询

相关内容如下：

——GetStructures：通过 GetStructuresRequest 报文调用该操作，接收 GetStructuresResponse 作为反馈。

4.5 其他行为

4.5.1 默认版本

当调用服务的报文没有指定版本时，默认返回申请源最新产生的版本。

4.5.2 解决参考和特定返回的对象问题

2.1版的SDMX-ML查询报文提供了解决参考类型和特定类型对象返回问题的新功能。SOAP API依靠该机制解决参考和特定返回的对象问题（见参考属性的应用和意义部分）。

4.5.3 压缩

能够使用适当的HTTP头字段（浏览器支持的编码类型）进行压缩。

4.5.4 基于 SOAP 的 SDMX Web 服务的实施

自WSDL有了标准规范以来，在SDMX Web服务中的一个新进展是契约优先。此外它是一个已备XML申请/反馈报文的Web服务（例如，应用程序逻辑接口是XML报文），因此，没有必要为SOAP有效载荷同本地语言类型之间的序列化和非序列化生成桩。参考方法是完全控制XML报文的请求/反馈。根据RPC范例，当使用自动生成代码时，将为SOAP申请操作的参数中增加一个额外元素，但是根据标准的SDMX WSDL，在SOAP规范中这个额外元素并不是必要的。

当在java中使用Apache Axis时，工具包会提供一个服务接口，该接口用DOM元素（DOM元素在Axis2中）读取/返回XML有效载荷。此外，当使用具有XML网络服务（JAX-WS）的java API时，开发商可以使用Provider（SOAPMessage）接口，他负责创建SOAP请求和反馈报文并指定标准化的WSDL服务。

然而在.Net环境下，没有类似的解决办法。服务开发商将在.NET网络方式中使用XmlAnyElement参数。这说明服务方式的参数可以是任何XML元素，从而允许开发商控制XML有效载荷。这种方法的细节见SDMX标准06部分中的“附件1：如何消除.NET SDMX网络服务中的额外元素”。

4.5.5 WS-I 符合性

为确保SDMX Web服务间的互用性，推荐所有SDMX Web服务均应符合WS-I简介文件1.1的要求。该文档可在http://www.ws-i.org/Profiles/BasicProfile-1.1-2004-08-24.html获得。推荐内容是关于SOAP和WSDL的用法。服务通用描述发现和集成（UDDI）对于SDMX Web服务存在性的推广非常有用，但对于SDMX互用性并非必须。

5 SDMX RESTful API

5.1 REST 简介

如下说明，SDMX API基于REST准则：

——REST 中，特定信息被称作“资源”。在 SDMX 中，特定资源可以是代码表、概念方案、数据结构定义、数据流等。每一项资源都能够通过全球标示符访问（例如：一个 URI）。

——通过 HTTP 协议中定义的方法（例如：查询、修改、上传、删除），可以完成资源操作。该 API 主要用于数据检索，因此，本标准仅包含 HTTP 查询的使用方法。

——资源的格式有很多种表示方式（例如不同类型和版本的 SDMX-ML 标准）。通过使用 HTTP 协商协议可以选择恰当的表示方式。

5.2 API 范围

RESTful API注重简洁，其目的在于简化有限标准查询的执行，而非丰富SDMX-ML查询报文的语义。不同于SDMX规范中的其他部分，RESTful API主要进行数据检索（通过HTTP查询）。

更确切地说，API允许：

——通过 ID、机构 ID 和版本号的组合，检索结构元数据。

——通过关键字（包含通配符选择和 OR 操作符支持）、数据或元数据流及数据或元数据供应方，检索统计数据或参考元数据。

——通过时间信息（起止期间）进一步细化统计数据或参考元数据的查询。

——仅检索更新和修订。

——以各种格式返回查询结果。所需格式和返回报文版本在 HTTP 内容协商（和 HTTP 协议机制）中指定。

——结构元数据指导 Web 服务解析引用信息（例如，查询数据结构定义时，可在返回的结果中检索概念和代码表）和使用相匹配模块的模块组（例如，借助匹配的数据结构定义检索数据流）。

——为保证效率，结构元数据可对结果进行最少信息版本的检索（例如，检索全部的代码表，若没有代码，可检索名称、id 等）。

——有些元素对被检索资源进行识别，有些元素能够就所需结果给出更多信息或进一步过滤，二者之间应当进行区分。URL 路径部分和查询字符串分别指定了属于第一类别和第二类别的元素。

5.3 结构元数据查询

5.3.1 资源

包含了下列资源的定义：

——datastructure[2)]
——metadatastructure[3)]
——categoryscheme
——conceptscheme
——codelist
——hierarchicalcodelist
——organisationscheme[4)]
——agencyscheme[5)]
——dataproviderscheme
——dataconsumerscheme
——organisationunitscheme
——dataflow
——metadataflow
——reportingtaxonomy
——provisionagreement
——structureset
——process
——categorization
——contentconstraint
——attachmentconstraint
——structure[6)]

5.3.2 参数

5.3.2.1 资源识别参数

下列参数用于识别资源，见表1

表1

参数	类型	描述
agencyID	符合 SDMX 的字符串 常见：NCNameIDType	维护待返回模块的机构
resourceID	符合 SDMX 的字符串 常见：IDType	待返回模块的 id
version	符合 SDMX 的字符串 常见:：VersionType	待返回模块的版本

2) 为简化 URLS,DataStructureDefinition 的缩写。
3) 为简化 URLS,MetadataStructureDefinition 的缩写。
4) 当组织方案（例如，维护机构）的作用未知或无关时，使用该资源。
5) OrganisationScheme 的三个子类（AgencyScheme，DataProviderScheme 和 DataConsumerScheme）、id 和版本参数都有固定值。更多内容请参考 SDMX 信息模型中第三部分。
6) 可用于检索同供应参数匹配的任一结构元数据。

以下语法详细说明了以上参数：protocol:// ws-entry-point/resource/agencyID/resourceID/version

进一步说明可能用到的关键字：见表2

表2

关键字	范围	描述
All[7)]	agencyID	返回由任一机构维护的模块[8)]
All	resourceID	返回由资源参数定义的全部类型的资源
All	version	返回资源的全部版本
latest	version	返回资源生成中的最新版本

遵循下列规则：

a) 没有指定版本时，应该返回产品的当前版本，即相当于使用关键字“latest”

b) 没有指定 agencyID 时，应返回由任何机构维护的匹配工具，即相当于使用关键字“all[9)]”。

c) 没有指定 resourceID 时，应返回匹配的全部模块（根据其它的使用标准），即相当于使用关键字“all”。

d) 没有指定参数时,应该返回由任一机构维护的,并由资源参数识别的“all”类型资源的“latest”版本。

5.3.2.2 进一步描述所需结果的参数

一旦资源被识别出，下列参数用于进一步描述所需的结果。正如3.2所提到的，这些参数在URL查询字符串中出现过。见表3。

表3

参数	类型	描述	默认值
Detail	字符串	该属性指定了待返回信息所需的数量。例如，可能指导 web 服务仅返回可维护模块的基本信息（如 id、agencyid、版本和名称）。值得注意的是，不会返回项目表中的项（如不会返回代码表查询中的代码）。可能的值包括：“allstubs”（全部模块应返回生成桩[10)]），“referencestubs”（参考模块应返回生成桩[11)]），full（应返回的全部模块的一切信息[12)]）。	Full
References	字符串	该属性指导Web服务是否返回被待返回模块引用(例如，与查询匹配的数据结构定义所使用的代码表和概念)和使用匹配模块（例如，与查询匹配的数据结构定义所使用的数据流）的模块组。值可能是“none”（不返回引用），“parents”（使用匹	None

7) 在 SDMX RESTful API 中，“all”是一个保留的关键字，最好不用其作代理、资源或特定版本的标识符。

8) 如没有指定，采取默认值。

9) 如果不同的代理向资源提供了同一个标识符，可能导致返回多个模块（例如，http://ws-entry-point/codelist/all/CL-FREQ 当不止一个代理维护着 id 为” CL-FREQ”的代码表，会返回多个代码表）。

10) 即 SDMX-ML 中的“stub 查询级和引用级”查询。

11) 即 SDMX-ML 中的“full 查询级和 stub 引用级”查询。

12) 即 SDMX-ML 中的“full 查询级和引用级”查询。

		配查询模块的模块组），“parentsandsiblings”（使用匹配模块和被查询模块引用的模块组），“children”（被待返回模块引用的模块），“descendants”（返回一切层级的参考引用），“all”（parentsandsiblings和descendants的结合）。此外，也可能要使用3.3.1中定义的具体的资源类型（例如，引用=代码表）	

5.3.2.3 引用属性的适用性和意义

下表列出了引用参数为“all”时所返回的内容。见表4。

表4

可维护模块	返回模块
Categorisation	All
CategoryScheme	Categorisations
Codelist	HierarchicalCodelist
ConceptScheme	Codelists
Constraint	OrganisationSchemes DataProviderSchemes DataStructureDefinitions Dataflows MetadataStructureDefinitions Metadataflows ProvisionAgreements
Dataflow	Constraints DataStructureDefinitions ProvisionAgreements ReportingTaxonomies StructureSets
DataProviderScheme	Constraint ProvisionAgreement
HierarchicalCodelist	Codelists
DataStructureDefinition	Codelists ConceptSchemes Constraints Dataflows StructureSets
Metadataflow	Constraints MetadataStructureDefinitions ProvisionAgreements ReportingTaxonomies StructureSets
MetadataStructureDefinition	ConceptSchemes

	Codelists DataProviderSchemes DataConsumerSchemes AgencySchemes OrganisationSchemes Constraints Metadataflows StructureSets
OrganisationScheme	None
Process	All
ProvisionAgreement	DataProviderSchemes Dataflows Metadataflows
ReportingTaxonomy	Dataflows Metadataflows
StructureSet	DataStructureDefinitions MetadataStructureDefinitions CategorySchemes DataProviderSchemes DataConsumerSchemes AgencySchemes OrganisationSchemes ConceptSchemes Codelists HierarchicalCodelists

5.3.3 实例

检索由ECB维护的id为ECB_EXR1的1.0版本DSD，以及检索DSD中使用的代码表和概念：
http://ws-entry-point/datastructure/ECB/ECB_EXR1/1.0?references=children
检索由ECB维护的id为ECB_EXR1的最新版本DSD，不检索DSD中的代码表和概念：
http://ws-entry-point/datastructure/ECB/ECB_EXR1
检索由ECB所维护的所有DSD，以及使用这些DSD的数据流：
http://ws-entry-point/datastructure/ECB?references=dataflow
检索由所有维护代理维护的所有代码表的最新版本，但不检索代码：
http://ws-entry-point/codelist?detail=allstubs
检索由ECB维护的所有可维护artefact的最新版本：
http://ws-entry-point/structure/ECB?detail=allstubs

5.4 数据和元数据查询

5.4.1 来源

需要支持以下资源：

a) 数据

b） 元数据

5.4.2 参数

5.4.2.1 用于识别资源的参数

以下参数用于在数据查询中识别资源：
见表5

表5

参数	类型	描述
flowRef[13)]	数据流字符串识别。语法是代理 id，模块 id，版本，之间用‘，’分隔，例如：AGENCY_ID,FLOW_ID,VERSION 如果字符串中仅包含 3 个元素中的一个，则是流 id，例 如:ALL,FLOW_ID,LATEST 如果字符串中仅包含 3 个元素中的两个，则这两个元素为代理 id 和流 id，例如：AGENCY_ID,FLOW_ID,LATEST	数据（或元数据）返回的数据（或元数据）流。
key	字符串符合 SDMX WADL 中定义的 KeyType。	所返回内容的关键字。通过略去通配维度的维度码来支持通配。例如，如果下列关键字能够识别美元对欧元的日汇率，D.USD.EUR.SP00.A，那么下列关键字就可用来检索所有货币对欧元的汇率：D..EUR.SP00.A。支持或操作符使用“+”，如下列关键字可同时检索美元、日元对欧元的汇率：D.USD+JPY.EUR.SP00.A。
providerRef[14)]	用于识别供应方的字符串。语法为：用“，”分隔代理 id 和供应方 id（例如，AGENCY_ID,PROVIDER_ID 如果字符串仅包含 2 个元素中的一个，那么这个元素就是供应方 id，即 ALL，PROVIDER_ID）。	检索数据（或元数据）供应方。如果没有提供，则返馈的报文将包含任意供应方提供的数据（或元数据）。

以上参数用以下语法来说明：
protocol://ws-entry-point/resource/flowRef/key/providerRef
而且，还要用到一些关键字：见表6

13) 在基于 SDMX 的 Web 服务中，这是一个常见用例，即流 id 已可以充分识别唯一的数据流。如果不能识别，可以将代理 id、数据流版本与流 id 联合使用，以识别特定数据流。
14) 在基于 SDMX 的 Web 服务中，这是一个常见用例，即供应方 id 已可以充分识别唯一的数据供应方。如果不能识别，可以将代理与供应方 id 联合使用，以识别特定数据供应方。

表6

关键字	范围	描述
All	Key	返回所有数据，该数据属于由特定供应方提供的特定数据流。
All[15)]	providerRef	返回所有数据，该数据与给定关键字匹配并属于由任意数据供应方提供的特定数据流。

使用参数要遵循下列规则：

a) 如果没有指定关键字，那么所有属于数据流（或元数据流）并通过 flowRef 进行识别的数据（或元数据）都应提供。这种情况相当于使用了关键字“all”。

b) 如果没有指定 providerRef 参数，应该返回任意数据供应方提供的匹配数据（或元数据）。相当于使用关键字“all”。

5.4.2.2 用于进一步过滤所需结果的参数

识别资源后，用以下参数对所需结果进行描述（或过滤）。在 3.2（SDMX Web 服务命名空间）中提到，在 URL 的查询字符串部分会用到这些参数。见表 7。

表7

参数	类型	描述
startPeriod	在 SDMXCommon.xsd schema 中的定义常见:StandardTimePeriodType，其描述如下[16)]： dateTime：所有介于日历日期之间的数据都将匹配 Gregorian Period：所有介于日历日期之间的数据都将匹配 Reporting Period：将返回所有介于指定时期之间的报告期数据。当将报告期频度为星期和日期的数据与更长的频度（如季度）数据进行比较时，必须根据频度覆盖的实际时间段来决定此数据是否应包括在内。如果这个数据介于指定报告频度期间，且基于一个 1 月 1 日开始的报告年度，则返回不同范围的数据或 Gregorian periods 数据。 如果涉及：或+符号，参数必须使用客户端编码率[17)]。 注意，我们是假定这个值包含在所查	提供查询结果的开始时期（包含开始时期）。

15) 在 SDMX RESTful API 中，“all”是一个保留关键字，因此不建议使用它作为供应方标识。

16) 详见第 6 部分 SDMX 技术说明事项中的 4.2.14 部分

17) 详见 http://en.wikipedia.org/wiki/URL_encoding#Percent-encoding_reserved_characters.

	找的数据范围之内。	
endPeriod	同上	提供查询结果的结束时期(包含结束时期)。
updatedAfter	xs:日期时间	数据库客户端执行查询的最后时间。如果用到这个属性,返回的信息应该仅包含自那个时点以来数据库的最新变更情况（更新和修订)。 应该包括： 自从最后一次执行查询语句后,增加的观测值[18)]（INSERT)。 自从最后一次执行查询语句后,更新的观测值（UPDATE)。 自从最后一次执行查询语句后,删除的观测值（DELETE)。 如果没有指定偏移量，默认为 web 服务的本地时间。
firstNObservations	正整数	该整数规定从第一个观测开始,每个匹配系列所返回观测的最大数量。
lastNObservations	正整数	该整数规定从最近的观测倒数,每个匹配系列所返回观测的最大数量。
dimensionAtObservation[19)]	与 SDMX 一致的字符串层级:NCNameIDType	依附于观测层级的维度 ID。
detail	字符串	这个属性指定了返回信息的所需数量。例如,可以让 web 服务仅返回数据（即无属性)。可能的选项为：“full”（默认为所有数据、文档和注释),“dataonly”（返回报文中不包含属性和组),“serieskeysonly”(仅返回序列关键字中的序列元素及维度。仅返回与特定查询匹配的序列,而不返回实际数据对于提升系统性能有用),“nodata”（在没有观测值的情况下，返回组和序列，包括属性和注释,)。

下表定义了参数组合的含义：见表8。

表8

startPeriod with no endPeriod	最近的

18) 如果在观测层级没有关于数据何时更新的信息，Web 服务将返回已更新的序列(如果信息附属于序列级)或数据流（如果信息附属于数据流级)。

19) 该参数用于横截面数据查询，指出哪个维度应该附属于观测层级。

endPeriod and no startPeriod	从开始
startPeriod and endPeriod	在给定时间范围之内
lastNObservations + startPeriod/endPeriod	在给定时间范围之内，从最后一个开始的特定观测量
firstNObservations + startPeriod/endPeriod +updatedAfterDate	在给定时间范围之内，给定时间戳变更后，从头开始的观测值
updatedAfterDate + startPeriod/endPeriod	在给定时间范围之内，给定时间戳变更后的观测值

5.4.3 实例

a) 检索由 ECB 为 ECB_EXR1_WEB 数据流提供的 M.USD.EUR.SP00.A 系列数据：
http://ws-entry-point/data/ECB_EXR1_WEB/M.USD.EUR.SP00.A/ECB
在这个例子中，假设数据流id（ECB_EXR1_WEB）能够识别唯一数据流，并且数据供应方id（ECB）能识别唯一数据供应方。

b) 检索由 ECB 为 ECB_EXR1_WEB 数据流提供的数据，对于给定的序列关键字使用二维通配字符。
http://ws-entry-point/data/ECB,ECB_EXR1_WEB,LATEST/M..EUR.SP00.A/ECB
在这个例子中，提供数据流的全引用（ECB是维护代理，ECB_EXR1_WEB是数据流id，LATEST是版本）

c) 检索匹配给定序列关键字数据的升级和版本情况，对第二维度使用或 r 操作符，对于 updatedAfterDate 要用编码率：
http://ws-entry-point/Data/ECB_EXR1_WEB/M.USD+GBP+JPY.EUR.SP00.A?updatedAfter=2009-05-15T14 %3A 15 %3A 00%2B01%3A00

d) 检索与给定序列关键字匹配的数据，并限定开始和结束日期：
http://ws-entry-point/data/ECB_EXR1_WEB/D.USD.EUR.SP00.A?startPeriod=2009-05-01&endPeriod=2009-05-31

5.5 模式查询

5.5.1 资源

定义下列资源：

模式：允许客户端请求返回一个XML 模式的服务，该模式定义了一定语境下的数据（或参考元数据）合法性。此服务必须考虑适用于语境的约束（DSD或MSD，数据流或元数据流，或提供协议）。

5.5.2 参数

5.5.2.1 识别资源的参数

以下参数用于识别资源：见表 9。

表9

参数	类型	描述
context	下列之一： datastructure,metadatastructure, dataflow, metadataflow or provisionagreement.	这个参数值定义了生成模式时需要考虑的约束条件。如果用到 datastructure 或 matadatastructure,生成模式时,

		必须使用与DSD或MSD相关的约束条件。如果用到 dataflow 或 metadataflow，生成模式时，必须使用与 dataflow 或 medadataflow、以及与 dataflow 或 metadataflow 中的 DSD 或 MSD 相关的约束条件。如果用到 provisionagreement，则生成模式时，必须使用与提供协议、协议中 dataflow 或 metatataflow 以及 dataflow 或 metadataflow 中的 DSD 或 MSD 相关的约束条件。
agencyID	与 SDMX 一致的字符串 常见:NCNameIDType	维护能够生成待返回方案模块的机构。
resourceID	与 SDMX 一致的字符串 common: IDType	生成待返回方案模块的 id。
version	与 SDMX 一致的字符串 common:VersionType	生成待返回方案模块的版本。

以上参数的指定使用以下语法：

protocol:// ws-entry-point/schema/context/agencyID/resourceID/version

而且，必须要用到一个关键字[20)]：见表 10。

表10

关键字	范围	描述
latest	版本	返回生产资源的最新版本

应遵循以下规则：

如果没有指定版本属性，应该返回产品当前版本，即相当于使用关键字“latest”。

5.5.2.2 进一步描述所需结果的参数

以下参数用于资源识别后，对结果进一步描述：见表 11。

表11

参数	类型	描述
dimensionAtObservation	与 SDMX 一致的字符串 common:NCNameIDType	与观测层级相关的维度 id
explicitMeasure	Boolean	为了对横截面数据进行有效性验证，表明是否需要强制输入观测（默认为 false）

5.5.3 实例

20) 由于方案查询必须至少符合一个模块，关键字“all”不支持 agencyID 和 resourceID

检索由 ECB 维护的 1.0 版本提供协议 EXR_WEB 的模式数据：

http://ws-entry-point/schema/provisionagreement/ECB/ EXR_WEB/1.0/

这个例子中，服务返回的模式必须考虑与提供协议相关的约束、提供协议中用到的数据流，以及数据流中用到的数据结构定义。

5.6 适合表述选择

可通过定义 http 内容协商机制[21)]为反馈报文选择合适的格式。在 http 内容协商机制下，客户端可以借助 Accept HTTP header[22)]来指定所需资源的格式和版本。

随着官方的 mime 类型(如:文本/ html,xml /应用等)，该标准还定义了一个允许服务定义自身类型的语法。SDMX Restful API 采用了此项功能，该语法如下：

application/vnd.sdmx.[format]+xml;version=[version[23)]], [format] 可以替换为所需格式（如: genericdata, structurespecificdata, structure 等），[version] 可以替换为一种 SDMX 标准版，最低为 SDMX 2.1（如： 2.1, 未来的 SDMX 版本 等)。

以下列出了几个实例：

——SDMX-ML 通用数据格式，2.1 版本:

application/vnd.sdmx.genericdata+xml;version=2.1

——SDMX-ML 特定结构数据模式，2.1 版：

application/vnd.sdmx.structurespecificdata+xml;version=2.1

——SDMX-ML 结构格式 2.1 版本:

application/vnd.sdmx.structure+xml;version=2.1

如果客户端没有指定反馈报文所需的格式和版本,或仅指定了通用 application/xml 格式,则 SDMX RESTful web 服务将返回：

——服务支持的 SDMX-ML 元数据结构查询的结构格式的最近版本；

——服务支持的 SDMX-ML 数据查询的通用数据格式的最近版本；

——服务支持的 SDMX-ML 元数据查询的通用元数据格式的最近版本。

下列内容显示了 SDMX RESTful 网络服务的有效格式，符合 2.1 版 SDMX 标准：

application/vnd.sdmx.genericdata+xml;version=2.1

application/vnd.sdmx.structurespecificdata+xml;version=2.1

application/vnd.sdmx.generictimeseriesdata+xml;version=2.1

application/vnd.sdmx.structurespecifictimeseriesdata+xml;version=2.1

application/vnd.sdmx.genericmetadata+xml;version=2.1

application/vnd.sdmx.structurespecificmetadata+xml;version=2.1

application/vnd.sdmx.structure+xml;version=2.1

application/vnd.sdmx.schema+xml;version=2.1

5.7 数据压缩

应该使用合适的 HTTP 协议头字段(浏览器接受的编码类型) 进行压缩。

6 SDMX Web 服务标准错误

21) 更多信息，请参考 http://www.w3.org/Protocols/rfc2616/rfc2616-sec12.html

22) 更多信息，请参考 http://www.w3.org/Protocols/rfc2616/rfc2616-sec14.html

23) 暂时只有 V2.1 被支持成为版本号。

6.1 介绍

在SDMX-ML 2.1版中，错误元素被应用于因查询而反馈的所有报文之中。如：Structure、MetadataStructure、GenericData、DSDData和Metadata。如果发生错误，错误元素会被添加至反馈报文中的structure:Structures | generic:GenericDataSet | message:DataSet | genericmetadata:MetadataSet | metadatareport:MetadataSet元素中。

这些元素来源于Message方案，并使用Common方案文件中的StatusTextType来表示。本标准末尾有一份方案文件的摘要，用于列示错误元素。

XML报文的错误部分适用以下2个用例：

a) 任何在SDMX数据传输到客户端之前检测到的错误都要在SDMX报文命名空间里定义的错误元素中反馈。

b) 错误发生在一些SDMX数据已经传输至客户端之后，那么就通过SDMX载荷中的“页脚”元素来提供错误信息。

6.2 REST Web服务错误处理

RESTful Web服务应该使用适当的HTTP状态代码指出错误。另外，无论何时，要首先使用SDMX-ML 2.1版中提供的错误报文来反馈错误。

6.3 SOAP Web服务

SOAP Web服务应该能够用标准的SOAP错误机制以及特定的命名空间来指示错误。另外，无论何时[24)]，要首先使用SDMX-ML 2.1版中提供的错误报文来反馈错误。

发生错误时，应该在SOAP 封包中使用以下元素：

a) <faultcode> 元素为错误号；

b) <faultstring> 元素为描述性信息；

c) <faultactor> 元素为Web服务的方法，这些方法以url为编写语言；

d) <detail> 元素是可选项，服务供应方可以使用它提供任何额外的有用信息。

6.4 错误分类

通过为错误信息进行编号可以将信息分为三类，并能为Web服务实现定制的报文提供支持。相关内容如下：

a) 000 – 499： 客户端导致的错误；

b) 500 – 999： 服务器导致的错误；

c) 1000 及以上： 自定义报文。

6.5 客户端导致的错误

6.5.1 No results found - 100

这一报文在SOAP和REST Web服务中的含义完全相同。如果查询结果为空，Web服务会反馈此报文。用这种方式提醒客户端查询结果为空。

6.5.2 Unauthorized - 110

当需要验证但验证失败或者尚未在系统中注册时使用此报文。

24) 根据SOAP版本框架1.2中规定，<faultcode>元素和其他反馈信息不可同时出现。

6.5.3 Response Too Large Due to Client Request 130

申请查询结果的信息量超过了客户端能够处理信息量的上限时反馈此报文。使用SDMX-ML查询时，客户端可能会对服务器反馈查询结果的信息量进行限制设定，如果查询结果的信息量超过客户端设定限值，服务器就会反馈此错误代码。

6.5.4 Syntax error - 140

以下情况会出现此错误代码：

a) SOAP 中：提供的 SDMX-ML 查询信息无效（XML 验证失败）；

b) REST 中：查询字符串不符合 SDMX RESTful 接口。

6.5.5 Semantic error - 150

当查询申请在语法上正确，但在语义不正确或违背了约定的业务规则时，Web服务将反馈此错误。

6.6 服务器导致的错误

6.6.1 Internal Server Error - 500

当没有其它更好的错误代码来描述服务失败原因时，Web服务将返回此错误代码作为一种有意义的反馈。

6.6.2 Not implemented - 501

如果Web服务还不能实现API中定义的方法时，Web服务将返回此错误代码。

注：即使SDMX Web服务唯一的功能是返回错误报文，所有SDMX Web服务应使用所有标准接口。这将使SDMX-compliant Web服务之间的互用性变得简单，并且将也降低通用SDMX Web服务客户端的开发难度。

6.6.3 Service unavailable - 503

如果 Web 服务因为服务器正在维护或其它类似原因而暂时不可用，则 Web 服务将返回此错误代码。

6.6.4 Response size exceeds service limit - 510

查询反馈结果的信息量大小超出了服务器的处理能力。

当存在Web服务能够向使用者提供在后期下载大量信息查询结果（如使用异步Web服务）的可能性时，Web服务可以将文件地址作为错误报文的一部分显示。在SOAP中，可以使用错误元素<faultstring>实现这一功能。

6.7 Custom Errors - 1000+

Web服务可以使用1000及以上的代码传送特定服务的错误报文。然而，不同服务可能针对不同错误使用相同的错误代码，因此，当遇到这类错误代码时，应该查询一下特定服务自身标准。

6.8 SDMX 与 HTTP 错误映射

下表给出了RESTful Web服务中的SDMX错误代码与http状态代码的映射关系，并指出了相应错误在SOAP中是如何返回的。见表12。

表12

SDMX 错误	REST 中的 http 错误代码	SOAP 中的错误代码
客户端错误		
100 No results found	404 Not found	SOAP Fault
110 Unauthorized	401 Unauthorized	SOAP Fault
130 Response too large due to client request	413 Request entity too large	SOAP Fault
140 Syntax error	400 Bad syntax	SOAP Fault
150 Semantic error	400 Bad syntax	SOAP Fault
服务器错误		
500 Internal Server error	500 Internal server error	SOAP Fault
501 Not implemented	501 Not implemented	SOAP Fault
503 Service unavailable	503 Service unavailable	SOAP Fault
510 Response size exceeds service limit	413 Request entity too large	Payload
1000+	500 Internal server error	SOAP Fault

有关Web服务客户端的查询示例见附录A。

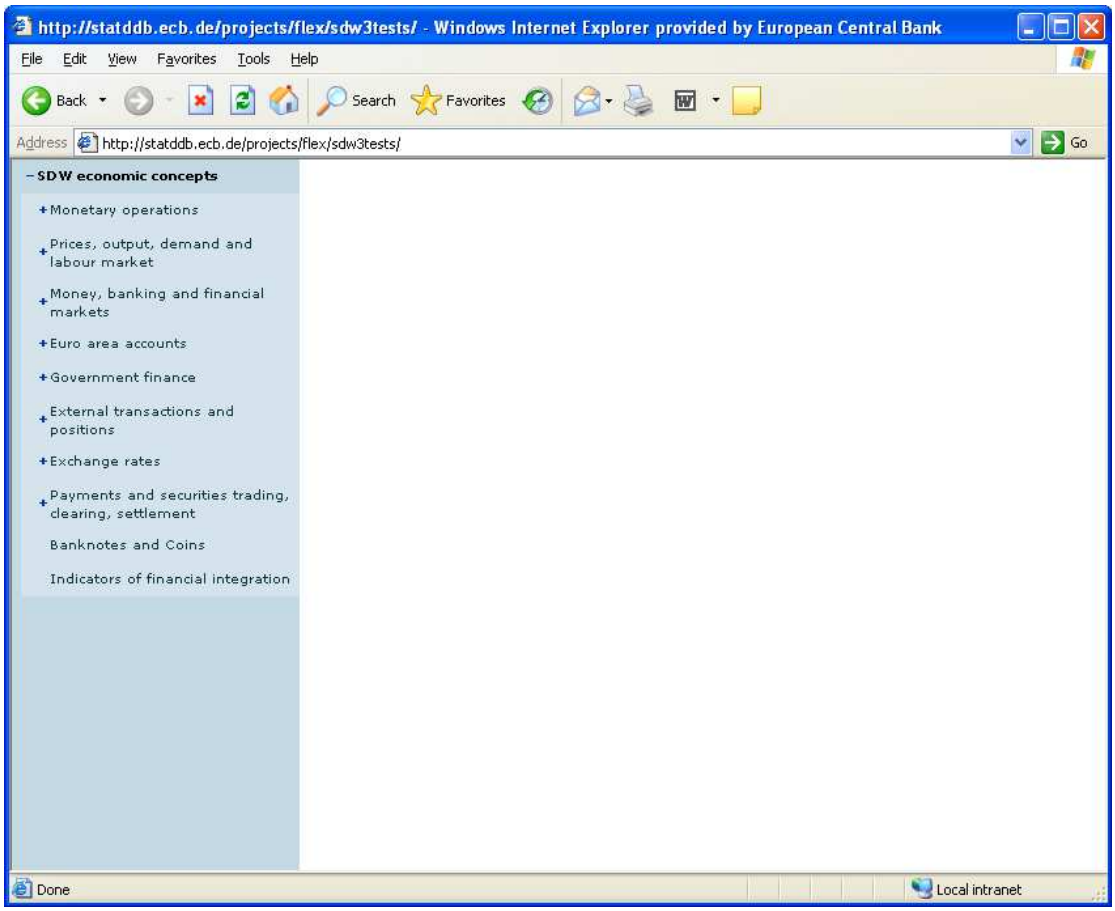
附 录 A
（资料性附录）
示例

A.1 Web服务客户端的查询示例

A.1.1 步骤1：使用主题领域列表浏览SDMX数据资源

A.1.1.1 用例

Web客户端使得通过浏览一系列主题领域来检索数据成为可能。客户端要求使用目前由ECB进行维护的SDW_ECON类别方案提供的版本。见图A.1。



图A.1

A.1.1.2 使用RESTful API申请查询

http://ws-entry-point/categoryscheme/ECB/SDW_ECON?references=categorisation

注意：使用值为“categorisation”的参考属性，将返回分类模式中用到的分类，并且这些分类中将包含类属于该分类的数据流参数。

A.1.1.3 使用SOAP API申请查询

```
<query:CategorySchemeQuery referenceResolution=”Shallow”>
  <query:References>
<query:Default/>
</query:References>
<query:CategorySchemeWhere>
<query:ID>SDW_ECON</query:ID>
<query:AgencyID>ECB</query:AgencyID>
</query:CategorySchemeWhere>
</query:CategorySchemeQuery>
```

注：为了描述得更清楚，这里忽略了SOAP封套。

A.1.1.4 反馈

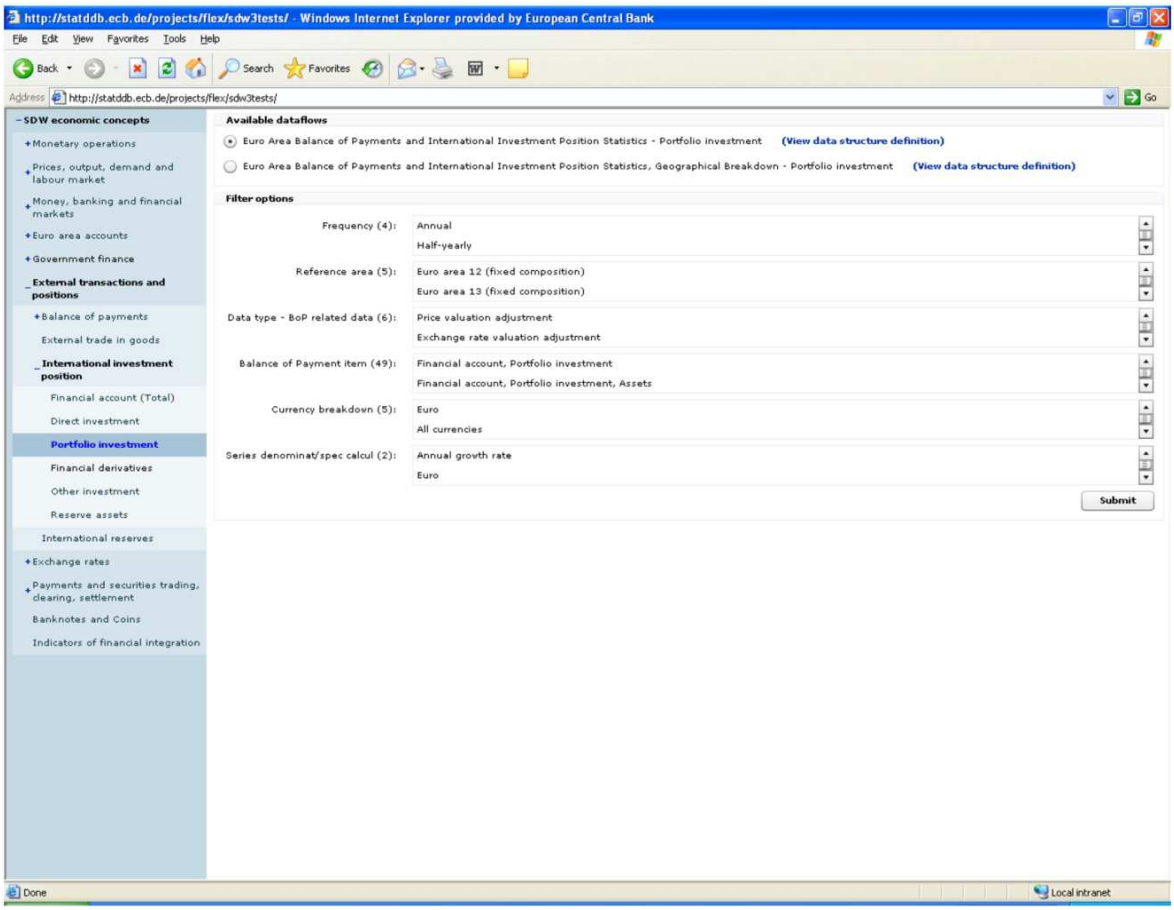
将返回包含着类别方案的SDMX-ML结构报文，同时返回带有数据流参数的分类。SDMX-ML结构报文的结构如下所示（为表述清楚，忽略了根元素、首元素和重复元素）：

```
<structure:Structures>
 <structure:CategorySchemes>
<structure:CategoryScheme>
</structure:CategoryScheme>
</structure:CategorySchemes>
<structure:Categorisations>
<structure:DataflowCategorisation>
</structure:DataflowCategorisation>
</structure:Categorisations>
</structure:Structures>
```

A.1.2 步骤2：数据选择

A.1.2.1 用例

一旦选定了主题领域和数据流，用户需要填充一个复选框，以便用户选择所需数据。为了保证只对数据库中存在的数据进行查询，需要对数据流进行约束。见图A.2。



图A.2

A.1.2.2 使用RESTful API申请查询

在这个查询示例中，数据流id为123456，代理id为ECB，版本为1.2。使用参数属性，将返回数据结构定义和约束条件。

http://ws-entry-point/dataflow/ECB/123456/1.2?references=all

A.1.2.3 使用SOAP API申请查询

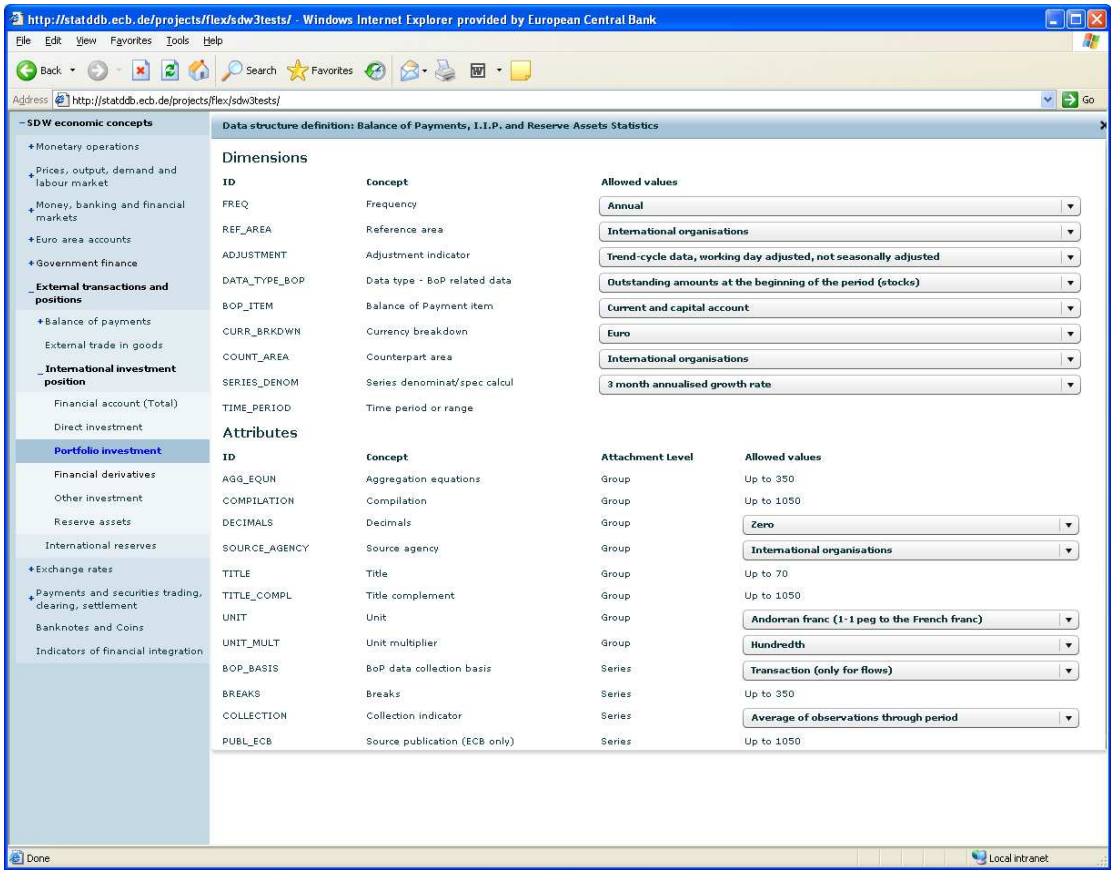
```
<query:DataflowQuery>
        <query:References>
<query:Default/>
</query:References>
<query:DataflowWhere>
<query:ID>123456</query:ID>
<query:Version>1.2</query:Version>
<query:AgencyID>ECB</query:AgencyID>
</query:DataflowWhere>
</ query:DataflowQuery>
```

A.1.2.4 反馈

SDMX-ML结构报文中包含所需数据流，数据结构定义以及相关的数据流约束条件。SDMX-ML结构报文的结构如下（忽略根元素和首元素）：

```
<structure:Structures>
<structure:Dataflows>
        <structure:Dataflow>
</structure:Dataflow>
</structure:Dataflows>
<structure:Codelists>
</structure:Codelists>
<structure:Concepts>
</structure:Concepts>
<structure:DataStructures>
</structure:DataStructures>
<structure:Constraints>
<structure:ContentConstraint>
</structure:ContentConstraint>
</structure:Constraints>
</structure:Structures>
```

选择数据前，如果用户想要查看数据流使用的数据结构定义，不用使用额外的查询命令，因为反馈的报文中已经包含了这些信息。见图A.3。

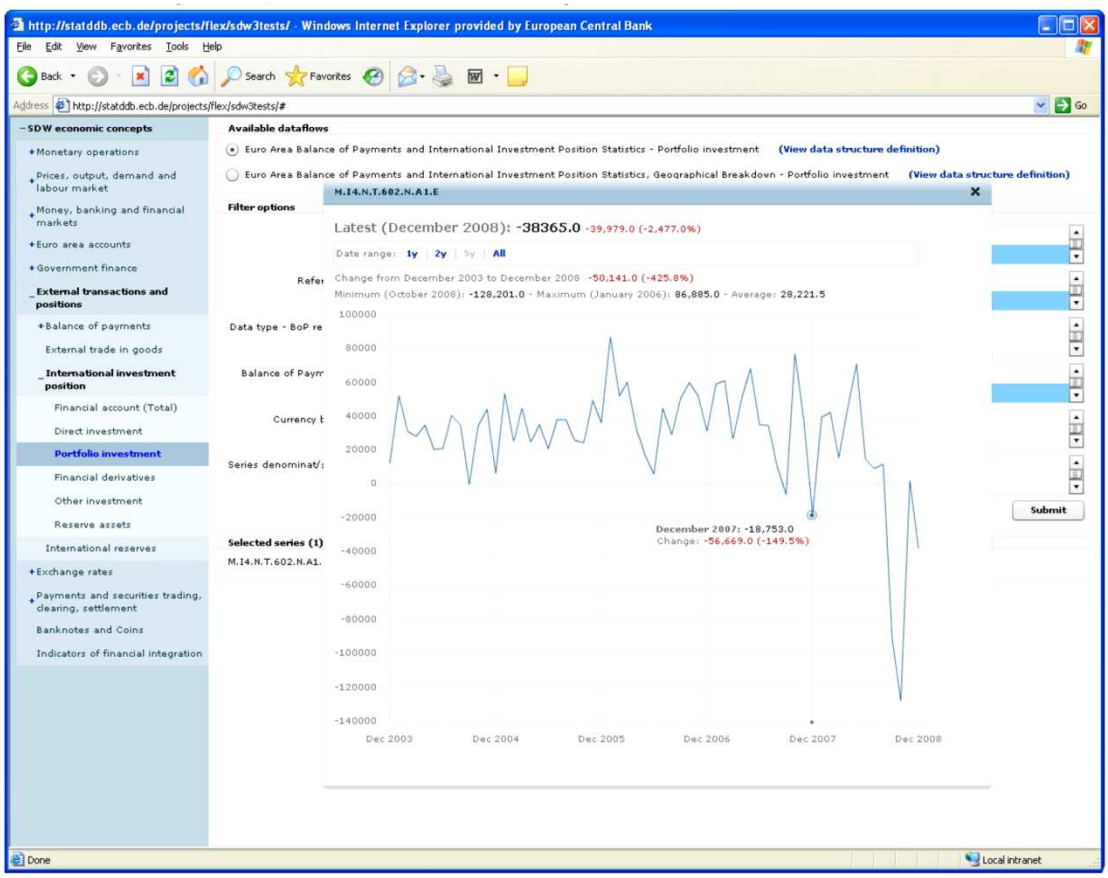


图A.3

A.1.3 步骤3：数据选择

A.1.3.1 用例

用户使用维度筛选框来检索感兴趣的数据。见图A.4。



图A.4

A.1.3.2 使用RESTful API申请查询

http://ws-entry-point/data/123456/M.I4.N.9.339+340+341.N.A1.A/ECB?startPeriod=2009-01&endPeriod=2009-12&detail=dataonly

注：除了数据流id（123456）外，数据的提供者被设定为ECB，并且对于第五个维度，序列关键字使用OR运算符。而且，仅返回2009年的数据。当返回数据的目的显示在图表中时，最详细的信息仅到数据这一层。因此，反馈报文中不包含属性和组信息。对于这种特定的Web服务，数据流参数则用简短形式来描述，数据流id和数据供应方id足够识别唯一数据流和数据供应方。如若不然，就必须提供全部参数（例如，使用ECB+123456+1.2，而不是123456）。

A.1.3.3 使用SOAP API申请查询

```
<query:Query>
        <query:DataWhere>
            <query:DataProvider>
                <common:OrganisationSchemeRef>
                <common:AgencyID>ECB</common:AgencyID>
```

```
        <common:ID>DataProviderScheme</common:ID>
        </common:OrganisationSchemeRef>
        <common:DataProviderRef>
                <common:ID>ECB</common:ID>
        </common:DataProviderRef>
    </query:DataProvider>
    <query:StructureUsage>
        <common:DataflowReference>
              <common:Ref>
        <common:AgencyID>ECB</common:AgencyID>
        <common:ID>123456</common:ID>
        <common:Version>1.2</common:Version>
              </common:Ref>
        </common:DataflowReference>
    </query:StructureUsage>
    <query:DimensionValue>
              <query:ID>FREQ</query:ID>
              <query:Value>M</query:Value>
    </query:DimensionValue>
    <query:DimensionValue>
              <query:ID>REF_AREA</query:ID>
              <query:Value>I4</query:Value>
    </query:DimensionValue>
    <query:DimensionValue>
              <query:ID>ADJUSTMENT</query:ID>
              <query:Value>N</query:Value>
    </query:DimensionValue>
    <query:DimensionValue>
              <query:ID>DATA_TYPE_BOP</query:ID>
              <query:Value>9</query:Value>
    </query:DimensionValue>
    <query:DimensionValue>
              <query:ID>CURR_BRKDWN</query:ID>
              <query:Value>N</query:Value>
    </query:DimensionValue>
    <query:DimensionValue>
              <query:ID>COUNT_AREA</query:ID>
              <query:Value>A1</query:Value>
    </query:DimensionValue>
    <query:DimensionValue>
              <query:ID>SERIES_DENOM</query:ID>
              <query:Value>A</query:Value>
    </query:DimensionValue>
```

```
                    <query:TimeDimensionValue>
                              <query:ID>TIME_PERIOD</query:ID>
                              <query:TimeValue
operator="GreaterThanOrEqualTo">2009-01</query:TimeValue>
                              <query:TimeValue
operator="LessThanOrEqualTo">2010-12</query:TimeValue>
                              </query:TimeDimensionValue>
                              <query:Or>
                                   <query:DimensionValue>
                                        <query:ID>BOP_ITEM</query:ID>
                                        <query:Value>339</query:Value>
                                   </query:DimensionValue>
                                   <query:DimensionValue>
                                        <query:ID>BOP_ITEM</query:ID>
                                        <query:Value>340</query:Value>
                                   </query:DimensionValue>
                                   <query:DimensionValue>
                                        <query:ID>BOP_ITEM</query:ID>
                                        <query:Value>341</query:Value>
                                   </query:DimensionValue>
                    </query:Or>
               </query:DataWhere>
</query:Query>
```

A.1.3.4 反馈

SDMX-ML 通用数据报文中包含申请的时间序列。

SDMX-ML 数据报文结构如下（忽略根元素和首元素）：

```
  <message:DataSet>
        <generic:Series>
        </generic:Series>
  </message:DataSet>
```

A.2 SDMX报文中的错误元素范例

```
<xs:element name="Error" type="ErrorType">
               <xs:annotation>
                    <xs:documentation>Error is used to communicate
                    that an error has occurred when responding to a
                    request in an non-registry environment. The
                    content will be a collection of error messages.
                    </xs:documentation>
               </xs:annotation>
          </xs:element>
```

```
<xs:complexType name="ErrorType">
    <xs:annotation>
        <xs:documentation>ErrorType describes the
        structure of an error response.
        </xs:documentation>
    </xs:annotation>
    <xs:sequence>
        <xs:element name="ErrorMessage"
    type="common:StatusTextType" maxOccurs="unbounded">
            <xs:annotation>
                <xs:documentation>ErrorMessage
                contains the error message. It can
                occur multiple times to communicate
                message for multiple errors, or to
                communicate the error message in
                parallel languages. If both messages
                for multiple errors and parallel
                language messages are used, then each
                error message should be given a code
                in order to distinguish message for
                unique errors.
                </xs:documentation>
            </xs:annotation>
        </xs:element>
    </xs:sequence>
</xs:complexType>
```

A.3 SOAP错误示例

```
<?xml version = "1.0" encoding = "UTF-8" ?>
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:sdmxerror="http://www.SDMX.org/resources/SDMXML/webservice/iso/v_
2_0_draft/error"
xmlns:sdmxws="http://www.SDMX.org/resources/SDMXML/webservice/iso/v_2_
0_draft">
<soapenv:Body>
<soapenv:Fault>
<faultcode>sdmxerror:500</faultcode>
<faultstring>Internal server error</faultstring>
<faultactor>sdmxws:GetCodelist</faultactor>
```

```
<detail>
<sdmxws:composite>
<sdmxws:code>1028</sdmxws:code>
<sdmxws:titles>
<sdmxws:title lang="de">Could not get connection from pool</sdmxws:title>
<sdmxws:title lang="en">Could not get connection from pool</sdmxws:title>
<sdmxws:title lang="fr">Could not get connection from pool</sdmxws:title>
</sdmxws:titles>
<sdmxws:source>SdmxRegistryService error: could not get connection from
pool</sdmxws:source>
</sdmxws:composite>
</detail>
</soapenv:Fault>
</soapenv:Body>
</soapenv:Envelope>
```