內容包裝 — 中文草案

中華民國國家標準 CNS	內容包裝	總號.	xxxxx-x
		類號.	xxxx-x

IMS Content Packaging

目錄

公布日期 年. . 月. . 日	經濟部標準檢驗局印行	修訂公布日期 年. . 月. . 日

印行年月年月

1.適用範圍

1.1 概述

IMS內容包裝標準(IMS Content Packaging Specification)描述資料結構、XML繫結(XML binding)，以及用於提供網際網路互通所伴隨的最佳實務(accompanying best practices)；所謂的網際網路互通係基於與內容產生工具、學習管理系統(LMS)及執行環境一起出現之內容。IMS內容包裝標準的範圍集中在定義系統間的互運性，以期能輸入、輸出、聚合、解聚內容包裝。

The IMS Content Packaging specification describes data structures, XML binding and accompanying best practices that are used to provide interoperability for Internet based content with content creation tools, learning management systems (LMS), and run time environments. The scope of the IMS Content Packaging specification is focused on defining interoperability between systems that wish to import, export, aggregate, and disaggregate Packages of content.

1.2 與其他標準之間的關係

透過IMS內容框架的整體目標圖，可完整了解IMS內容包裝標準的完整及相關範圍。此目標透過本標準提供足夠的指示讓人們建立、管理以及與線上學習素材互動。

The entire, extended scope of the IMS Content Packaging specification is complemented by the overall goals of the IMS Content framework. Those goals are to provide enough guidance, through this specification, that people may build, manage, and interact with interoperable, online learning materials.

下列著作爲原始的1.1版框架發展之產出物：

The following historical work was considered in the development of the original version 1.1 framework:

(1) IMS API draft specification version 0.6 (6/98)。

(2) IMS Packaging draft specification version 0.6 (2/99)。

(3) The Aviation Industry CBT Committee’s (AICC) API for Web Implementation of AICC/IEEE CMI specification (9/99)。

(4) The Advanced Distributed Learning Initiative’s (ADL) Sharable Content Object Reference Model (11/99)。

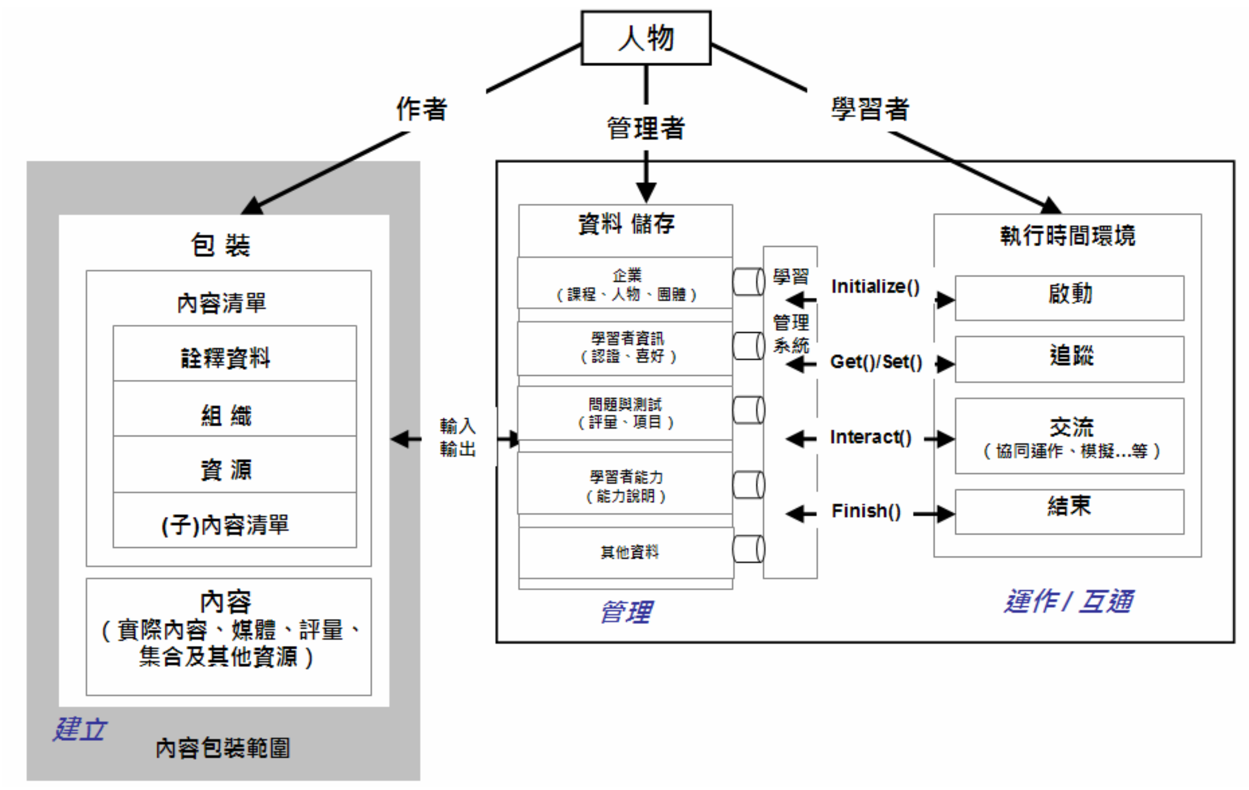
(5) The Microsoft Learning Resource Interchange (LRN) specification (01/00)。

IMS內容標準的範圍經過一系列會議與團體討論後製成圖表。內容範圍的概括圖如圖1.1所示。

The scope of the IMS Content specification was captured in a diagram through a series of meetings and group discussions. This expanded view of the Content scope is depicted in Figure 1.1.

圖1.1 IMS內容框架

Figure 1.1 IMS Content framework



IMS內容框架整個範圍大而複雜。爲了降低複雜度與減少完成初步標準的時間，而將範圍分爲三部分，主要分爲：內容包裝、資料模型、執行時間環境。下面幾節就對每個部分補充說明與更詳細的描述。

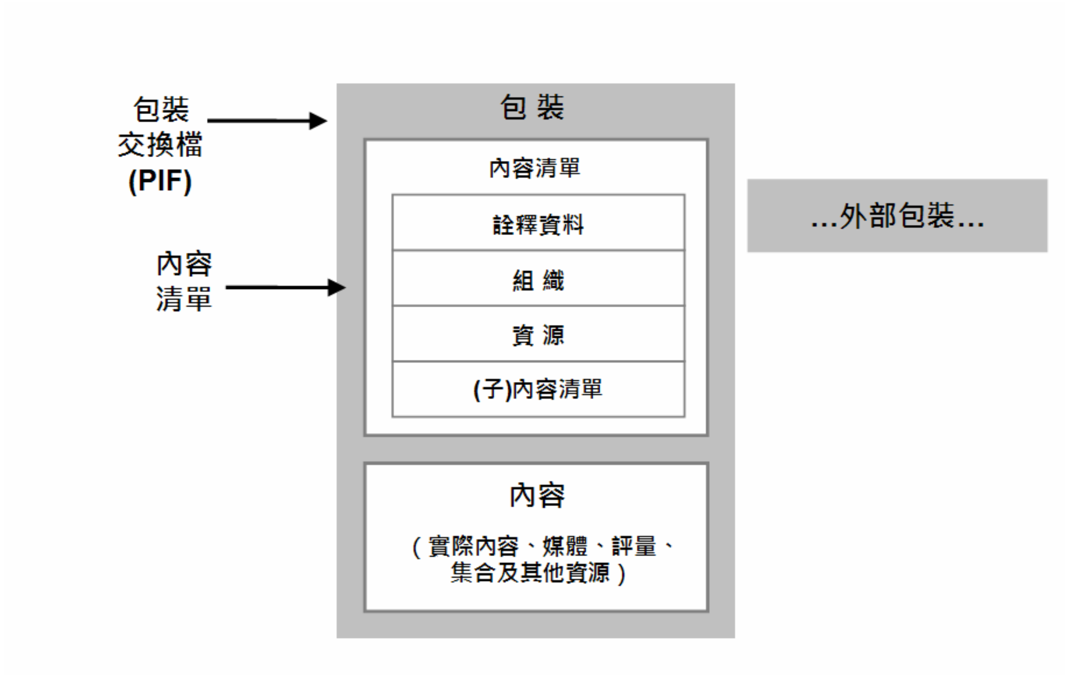
The complete, identified scope of the IMS Content framework is large and complex. To reduce the complexity and decrease the amount of time needed to complete a first specification, the scope was broken down into three, main parts: Content Packaging, Data Model, and Run Time Environment. Each of these topics requires additional explanation and each is described in more detail in the following sections.

(1)內容包裝(Content Packaging)

IMS內容包裝是IMS內容框架的一部分，說明內容資源集合的政策、課程組識與詮釋資料。構成IMS內容包裝標準集合範圍的文件全部描述於圖1.2。

The IMS Content Packaging portion of the IMS Content framework represents the section that deals with the issues of content resource aggregation, course organization, and meta-data. All of the documents that comprise the IMS Content Packaging specification are focused on the scope represented in Figure 1.2.

圖 1.2 IMS 內容包裝範圍



(2)資料模型(Data Model)

IMS內容格式的未來版本將滿足普遍而可擴充的內容資料模型的重要核心問題。此資料模型說明IMS內容框架的部分中哪些內容被匯入、被儲存、被管理與進行教學用途之處理。學習管理系統業者和電腦平台業者在定義標準的部分中，扮演著關鍵的角色。

A future version of an IMS Content specification will address important, core issues of a general and extendable content data model. The data model represents that portion of the IMS Content framework where content is imported, stored, managed, and manipulated for instructional purposes. LMS vendors and computer platform vendors will play a key role in defining this portion of the specification.

未來IMS內容標準也將考慮到IMS企業、問題與測試的溝通、與學習者資訊包裝標準在資料模型中扮演怎麼樣的角色。考量其他的作用，例如：ADL與AICC範圍內所做的工作，我們可以分辨哪些是泛領域皆共同的、而哪些部分需要特別區分爲一特定團體。內容團隊也將謹慎地規定如何將資料模型說明予以延伸的機制，讓不同團體皆能運用IMS內容框架。

A future IMS Content specification will also take into account how the IMS Enterprise, Question and Test Interoperability, and Learner Information Package specifications play a role in the data model. Other efforts such as the work that has been done within the ADL and AICC are being considered to determine which parts we can agree on that are common across all domains and which parts are specific to a particular community. The content team will also carefully determine a mechanism for how extensions to the data model may be represented so that different communities can use the IMS Content framework.

(3)執行時間環境(Run Time Environment)

一種未來的IMS內容標準說明將負責處理執行時間環境周遭的問題。IMS內容框架的執行時間環境部分是說明學習者與呈現給他們的內容互動此點。格式中此部分的關鍵需求之一將是確定執行時間環境與學習管理系統(LMS)溝通的標準機制。

A future IMS Content specification will deal also with the issues surrounding run time environments. The run time environment portion of the IMS Content framework represents the point where learners will interact with the content presented to them. One of the key requirements for this portion of the specification will be the identification of standard mechanisms to enable communication between a run time environment and an LMS.

2.用語釋義

2.1 一般術語

2.1.1 ADL(Advanced Distributed Learning)

此係在 1997 年由美國白宮成立先進分散式學習先導計畫，其目的是爲了提升線上訓練的使用。

2.1.2 AICC(Aviation Industry CBT Committee)

航空工業 CBT 委員會爲會員制的國際論壇，發展航空業界的交流學習技術。

2.1.3 字元集

電腦用以顯示資訊所使用的字元。

2.1.4 選擇

測試者可以選擇的可能回應之一。選擇包括正確的與錯誤的回答。

2.1.5 符合性聲明

符合性聲明爲顧客提供一種可以公正比較評價工具與內容的業者之機制。

2.1.6 資料庫

資訊/資料的集合，通常構築在表格、電腦大量儲存系統之下。資料庫以能提供電腦軟體快速搜索與取出的方法來結構。以下的資料庫是用於測試系統：項目、測試定義、列表與結果。

2.1.7 DTD(Document Type Definition)

文件型式定義。

2.1.8 動態排序

以測試者前一次回應爲基礎的項目或章節之排序。

2.1.9 元件

一種 XML 術語。按照電腦可理解的方式來定義 XML 文件中之元件。

2.1.10 元件內容

一種 XML 術語，用以描述元件之內容。

2.1.11 元件屬性（element attributes）

提供關於元件的額外資訊。

2.1.12 IEEE（Institute of Electrical and Electronics Engineers）

國際電機電子工程師協會，提供發展標準和標準的論壇。

2.1.13 IMS（IMS Global Learning Consortium）

致力於發展分散式學習標準的組織。

2.1.14 LTSC(Learning Technology Standards Committee)

學習技術標準委員會

2.1.15 LMS(Learning Management System)

學習管理系統，負責管理學習經驗的系統。

2.1.16 Meta-data

詮釋資料：資料的描述資訊。可想像爲資料的資料。IMS 標準通常使用 Metadata 來描述學習資源。

2.1.17 W3C(World Wide Web Consortium)

全球資訊網聯盟

2.1.18 XML

可延伸標誌語言是一種標準，由 W3C 創作。

2.1.19 XSD（XML Schema Definition）

XML 架構定義。

2.Terms and definitions

2.1General Terms

2.1.1 ADL

Advanced Distributed Learning Initiative was started by the United States White House in 1997, and aims to advance the use of online training.

2.1.2 AICC

Aviation Industry CBT Committee is a membership-based international forum that develops recommendations on interoperable learning technologies for the aviation industry.

2.1.3 character set

The characters used by a computer to display information.

2.1.4 choice

One of the possible responses that a test taker might select. Choices contain the correct answers and distracters.

2.1.5 conformance statement

A conformance statement provides a mechanism for customers to fairly compare vendors of assessment tools and content.

2.1.6 database

A collection of information/data, often organized within tables, within a computer’s mass storage system.Databases are structured in a way to provide for rapid search and retrieval by computer software. The following databases are used by testing systems: item, test definition, scheduling, and results.

2.1.7 DTD

Document Type Definition.

2.1.8 dynamic sequencing

The sequencing of items or sections is based upon previous responses from a test taker.

2.1.9 element

An XML term that defines a component within an XML document that has been identified in a way a computer can understand.

2.1.10 element contents

An XML term used to describe the content of the element.

2.1.11 element attributes

Provides additional information about an element.

2.1.12 IEEE

Institute of Electrical and Electronics Engineers that provides a forum for developing specifications and standards.

2.1.13 IMS

An organization dedicated to developing specification for distributed learning.

2.1.14 LTSC

Learning Technology Standards Committee.

2.1.15 LMS

Learning Management System which is the system responsible for the management of the learning experience.

2.1.16 Meta-data

Meta-data: Descriptive information about data. Can be thought of as data about data. IMS specifications typically use meta-data to describe learning resources.

2.1.17 W3C

World Wide Web Consortium.

2.1.18 XML

Extensible Mark-up Language is a specification, produced by the W3C.

2.1.19 XSD

XML Schema Definition.

2.2 內容包裝元件與屬性

2.2.1 default

指定以哪個組織架構爲預設。

2.2.2 dependency

定義包含其從屬檔案的資源位址。

2.2.3 file

資源所屬之檔案。

2.2.4 CPI(Content & Packaging Interchange)

內容與套裝交換

2.2.5 href

URL 之參照。

2.2.6identifier

一個識別符在內容清單中係唯一。

2.2.7 identifierref

在內容清單或資源中之一個識別符的參考。

2.2.8 isvisible

指示一個項目在包裝顯示或呈現時是否要顯示。

2.2.9 item

一個組織裡之節點。

2.2.10 manifest

一個可重覆使用的教學單元。包含了 metadata、organizations 及 resource 之參考。

2.2.11 metadata

描述內容清單之詮釋資料。

2.2.12 organization

定義一個特定階層式組織。

2.2.13 organizations

描述一或多個結構，或是此包裝之組織。

2.2.14 parameters

在創立時要被傳進資源裡之靜態參數。

2.2.15 resource

一個資源之參考。

2.2.16 resources

一個資源參考之集合。並沒有設想順序或層級。

2.2.17 schema

描述定義和控制此內容清單所使用之綱要。

2.2.18 schemaversion

描述上述架構之版本（例：1、0、1.1）。

2.2.19 title

此組織之標題。

2.2.20 type

指示資源之型式。

2.2.21 version

定義此 manifest 之版本(例：1.0)。

2.2.22 xml base

為包裝裡的相關 URI 提供相關路徑分支。

2.2Content Packaging Elements and Attributes

2.2.1 default

Indicates which organization scheme is the default one.

2.2.2 dependency

Identifies the location of a resource that contains dependent files.

2.2.3 file

A reference to a file that a resource is dependent on.

2.2.4 CPI

Content & Packaging Interchange

2.2.5 href

A reference to a URL.

2.2.6 identifier

An identifier that is unique within the manifest.

2.2.7 identifierref

A reference to an identifier in the manifest or in a resource.

2.2.8 isvisible

Indicates whether or not an item is displayed when the package is displayed or rendered.

2.2.9 item

A node within the organization.

2.2.10 manifest

A reusable unit of instruction. Encapsulates meta-data, organizations, and resource references.

2.2.11 metadata

Meta-data describing the manifest.

2.2.12 organization

Defines a particular hierarchical organization.

2.2.13 organizations

Describes one or more structures, or organizations for this package.

2.2.14 parameters

Static parameters to be passed to the resource at launch time.

2.2.15 resource

A reference to a resource.

2.2.16 resources

A collection of references to resources. There is no assumption of order or hierarchy.

2.2.17 schema

Describes the schema that defines and controls the manifest.

2.2.18 schemaversion

Describes version of the above schema (e.g., 1,0, 1.1).

2.2.19 title

Title of the organization.

2.2.20 type

Indicates the type of resource.

2.2.21 version

Identifies the version of this manifest (e.g., 1.0).

2.2.22 xml base

Provides a relative path offset for relative URIs in the package.

3.引用標準

[CP, 04a]	IMS Content Packaging Information Model v1.1.4, C.Smythe, A.Jackl, IMS Global Learning Consortium, Inc., October 2004.
[CP, 04b]	IMS Content Packaging XML Binding v1.1.4, C.Smythe, A.Jackl, IMS Global Learning Consortium, Inc., October 2004.
[CP, 04c]	IMS Content Packaging Best Practice and Implementation Guide v1.1.4, C.Smythe, A.Jackl, IMS Global Learning Consortium, Inc., October 2004.
[CP, 04d]	IMS Content Packaging Summary of Changes v1.1.4, C.Smythe, A.Jackl, W.Kraan, IMS Global Learning Consortium, Inc., October 2004.
[IMSBUND, 01]	Using IMS Content Packaging to Package Instances of LIP and Other IMS Specifications v1.0, B.Olivier, M.McKell, IMS Global Learning Consortium, Inc., August 2001.
[IMSPLID, 01]	IMS Persistent, Location-Independent, Resource Identifier Implementation Handbook v1.0, M.McKell, IMS Global Learning Consortium, Inc., April 2001.
[ISO/IEC10646]	ISO (International Organization for Standardization). ISO/IEC 10646-1993 (E). Information technology - Universal Multiple-Octet Coded Character Set (UCS) - Part 1: Architecture and Basic Multilingual Plane. [Geneva]: International Organization for

Standardization, 1993 (plus amendments AM 1 through AM 7).

[MD, 04] IMS Meta-data Best Practice Guide for IEEE 1484.12.1-2002 Standard for Learning Object Metadata v1.3, P.Barker, L.Campbell, A.Roberts, IMS Global Learning Consortium, Inc., May 2004.

[MD, 901] IMS Learning Resource Meta-data v1.2.1, S.Thropp, M.McKell, IMS Global Learning Consortium, Inc., September 2001.

[MD, 501] IMS Learning Resource Meta-data v1.2, T.Anderson, M.McKell, IMS Global Learning Consortium, Inc., May 2001.

[Unicode, 96] The Unicode Consortium. The Unicode Standard, Version 2.0. Reading, Mass.: Addison-Wesley Developers Press, 1996.

[XML, 98] XML 1.0 Specification of the W3C: http://www.w3.org/TR/1998/REC-xml-19980210.

[XML, 99] XML Namespace Recommendation of W3C: http://www.w3.org/TR/1999/REC-xml-names-19990114.
XML Schema Recommendation of W3C: http://www.w3.org/TR/2001/REC-xmlschema-0-20010502/.

4.內容包裝資訊模型

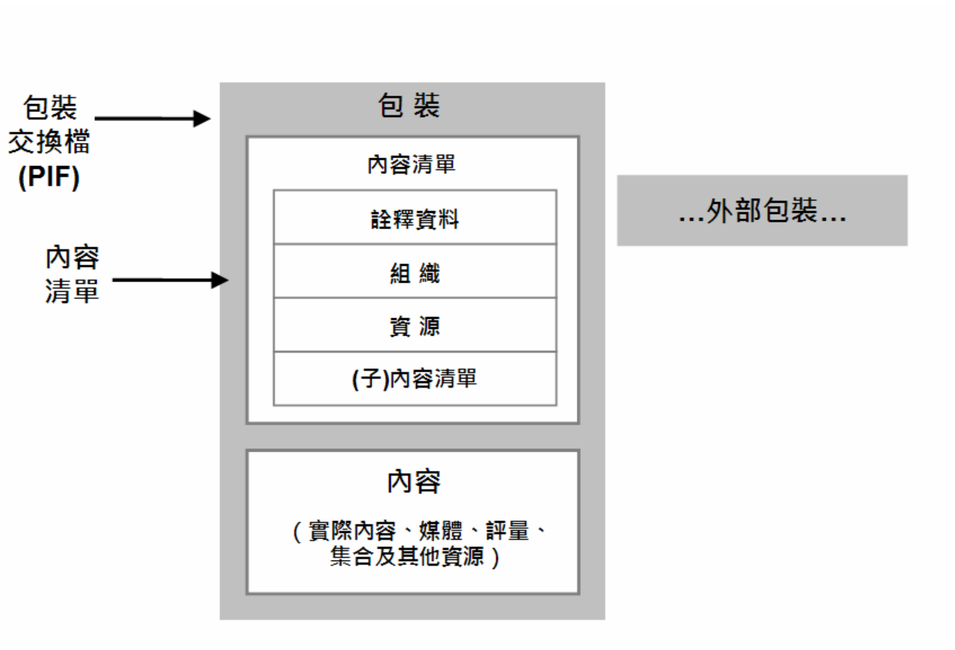
圖 4.1 爲描述 IMS 內容套裝資訊模型組件之概念圖。此爲 IMS 內容套裝最佳實務指南中，形成其他標準的主要基礎框架。

4. IMS Content Packaging Conceptual Model

Figure 4.1 is a conceptual diagram that illustrates the components of the IMS Content Packaging Information Model. As indicated in the IMS Content Packaging Best Practice Guide, this is part of the larger IMS Content Framework, which forms the basis for this and future specifications.

圖 4.1 IMS內容包裝範圍

Figure 4.1 IMS Content Package Scope



4.1 概念模型詳述(Conceptual Model Discussion)

圖 4.1 描繪的 IMS 套裝由兩種主要元件組成：一個特別的 XML 檔案用以描述套裝中之內容組織和資源，以及由此檔案所描述的檔案資源。此特別的 XML 檔案稱爲 IMS 內容清單檔案，因爲課程內容和組織都是基於內容清單的上下文來描述的。當一個套裝被合併成用以傳輸的單一檔案，此檔案稱爲套裝交換檔案。這些部份與內容容器之間的關係描述如下：

The IMS Package depicted in Figure 4.1 consists of two major elements: a special XML file describing the content organization and resources in a Package, and the file resources being described by the XML. The special XML file is called the IMS Manifest file, because course content and organization is described in the context of ‘manifests’. Once a Package has been incorporated into a single file for transportation, it is called a Package Interchange File. The relationship of these parts to the content container is described below:

(1) 套裝交換檔案（Package Interchange File）：單一的檔案。（例：.zip、.jar、.cab 檔），其中包含一個命名爲 imsmanifest.xml 的最上層內容清單，和其他藉由內容清單識別的檔案。套裝交換檔案係一個簡要的 Web 傳送格式，用以運送相關連且結構化的資訊。建議採用 PKZip v2.04g(.zip)做爲預設的套裝交換檔案格式。任何 ZIP 檔都必須遵守 RFC1951 規範。

Package Interchange File – a single file, (e.g., ‘.zip’, ‘.jar’, ‘.cab’) which includes a top-level manifest file named “imsmanifest.xml” and all other files as identified by the Manifest. A Package Interchange File is a concise Web delivery format, a means of transporting related, structured information. PKZip v2.04g (.zip) is recommended as the default Package Interchange File format. Any ZIP file format MUST conform to RFC1951.

(2) 套裝（Package）：一個邏輯目錄。其包含了特別命名的 XML 檔案、任何直接參考的 XML 控制文件（例如：DTD 或者 XSD 檔案）、以及實際檔案資源。檔案資源則可能組織在子目錄。

Package – a logical directory, which includes a specially named XML file, any XML control documents it directly references (such as a DTD or XSD file), and contains the actual file resources. The file resources may be organized in sub-directories.

(2.1) 最上層內容清單：必備的 XML 元件，用以描述包裝本身之資訊。其可能也包含選項的子內容清單。每個內容清單實例包含下列段落：
Top-level Manifest – a mandatory XML element describing the Package itself. It may also contain optional sub-Manifests. Each instance of a manifest contains the following sections:

(a) 詮釋資料段：用以描述全體內容清單之 XML 元件；
(b) 組織段：用以描述內容清單中 0 個、1 個或多個之內容組織之 XML 元件。
(c) 資源段：參考在內容清單中所有需要的實體資源與媒體元件之 XML 欄位。包括了描述資源的詮釋資料與所有參考任何外部檔案。
(d) 子內容清單：1 或多個選項的邏輯巢狀內容清單。

(a) Meta-data section – an XML element describing a manifest as a whole;
(b) Organizations section – an XML element describing zero, one, or multiple organizations of the content within a manifest;
(c) Resources section – an XML element containing references to all of the actual resources and media elements needed for a manifest, including meta-data describing the resources, and references to any external files;

(d) sub-Manifest – one or more optional, logically nested manifests;

(2.2) 檔案資源：及內容清單中所敘及之實體媒體元件、文字檔案、圖像，以及其他資源。這些檔案資源可能組織於子目錄中。

File Resources – these are the actual media elements, text files, graphics, and other resources as described by the manifest(s). The file resources may be organized in sub-directories.

套裝表示一個可使用（與可重複使用）的內容單位。其可能是某一教育課程中課程組織之外的一部分，可被獨立遞送，作爲一整個課程或者是一個課程集合。一旦包裝到達其目的地如學習管理系統業者並執行服務，套裝必須可以被集合或解開到其他套裝。套裝必須能保持獨立，亦即當它被解開時，須包含所有使用其內容來學習時所需要的全部資訊。

Package – A Package represents a unit of usable (and reusable) content. This may be part of a course that has instructional relevance outside of a course organization and can be delivered independently, as an entire course or as a collection of courses. Once a Package arrives at its destination to a run time service, such as an LMS vendor, the Package must allow itself to be aggregated or disaggregated into other Packages. A Package must be able to stand alone; that is, it must contain all the information needed to use the contents for learning when it has been unpacked.

套裝不需合併爲套裝交換檔案。一套裝也可能被分散在 CD-ROM 或其他可移動的媒介而不壓縮爲單一檔案。直接被它參考（如 DTD、XSD）的 IMS 內容清單檔案和任何其他的支援 XML 檔案必須放在存放媒體的根目錄下。

Packages are not required to be incorporated into a Package Interchange File. A Package may also be distributed on a CD-ROM or other removable media without being compressed into a single file. An IMS Manifest file and any other supporting XML files directly referenced by it (DTD, XSD) must be at the root of the distribution medium.

(3) 內容清單（Manifest）：內容清單是用 XML 對教學資源所做的描述。內容清單可能也包含將此教學資源組織以呈現的 0 或多個固定的方法。

Manifest – A manifest is a description in XML of the resources comprising meaningful instruction. A manifest may also contain zero or more static ways of organizing the instructional resources for presentation.

內容清單的範圍是彈性的，可以描述課程中除了課程背景（教學物件）以外的個別部分、一門課程，或是整個課程集合。此裁量是由內容開發者來判斷，照他們想要它聚合或解聚

來描述其內容。原則上一個套裝必會包含一最上層內容清單，它可能包含一個或多個子內容清單。最上層內容清單通常負責描述此套裝。而巢狀子內容清單則描述其子內容清單該層級範圍中的內容，例如：課程、教學物件等。

The scope of manifest is elastic. A manifest can describe part of a course that can stand by itself outside of the context of a course (an instructional object), an entire course, or a collection of courses. The decision is given to content developers to describe their content in the way they want it to be considered for aggregation or disaggregation. The general rule is that a Package always contains a single top-level manifest that may contain one or more sub-Manifests. The top-level manifest always describes the Package. Any nested sub-Manifests describe the content at the level to which the sub-Manifest is scoped, such as a course, instructional object, or other.

舉例來說，如果組成一課程的全部內容皆相當緊密地的結合，除了課程文章之外就沒有任何部分能被提出來，內容開發者需產生單獨的內容清單來描述此課程的資源和組織。總之，產出此類能跟其他教學物件配合產生不同課程呈現的教學物件的內容開發者，他們要描述每個教學物件自己的內容清單，然後集合此類內容清單到包含課程組織的較上層內容清單裡。最後，內容開發者要搬移多個課程至單一包裝（一門課程），並使用最上層內容清單來容納所有課程層級內容清單，以及各課程所包含的所有的教學物件內容清單。

For example, if all content comprising a course is so tightly coupled that no part of it may be presented out of the course context, a content developer would want to create a single manifest to describe that course’s resources and organization. However, content developers who create “instructional objects” that could be recombined with other instructional objects to create different course presentations would want to describe each instructional object in its own manifest, then aggregate those manifests into a higher-level manifest containing a course organization. Finally, a content developer who wants to move multiple courses in a single Package (a curriculum), would use a top-level manifest to contain each course-level manifest and any instructional object manifests that each course might contain.

(4) 資源（Resource）：內容清單中描述之素材，例如：網頁、媒體檔案、文字檔案、評量物件、或是其他檔案格式的資料段落等。資源可能也包含包裝外部的可得資源，透過子內容清單的 URL 或資源收集來描述。此類資源的整合通常被歸類爲內容。每個資源可在內容清單的 XML 中透過各別的<resource>欄位描述。此欄位包括了使用此資源需要的全部條件列表。包裝中包含的檔案則被列表在<resource>欄位中的<file>欄位裡。

Resource – The resources described in the manifest are assets such as Web pages, media files, text files, assessment objects or other pieces of data in file form. Resources may also include assets that are outside the Package but available through a URL, or collections of resources described by sub-Manifests. The combination of resources is generally categorized as content. Each resource may be described in a <resource> element within a manifest’s XML. This element includes a list of all the assets required to use the resource. The files included in the Package are listed as <file> elements within such <resource> elements.

當全部的內容清單裡的資源內容皆被引用時，就不需要將內容清單內所有的公佈資源皆引用到<organization>欄位裡的<item>欄位。此特徵適用於以下兩種情況：

While all content should be referenced in the resource section of the manifest, it is not necessary for all declared resources in a manifest to be referenced by <item> elements in the <organization> section of a manifest. This feature can be useful in two cases:

(a) 當檔案包含不需要呈現給學習者的資料時。
(b) 當包裝被用於內容檔案而不打算呈現給學習者時。

(a) When a file needs to be included that does not need to be presented to the learner;
(b) When the package is used as a content archive that is not designed to be presented to a learner.

4.2 內容清單檔案之標準名稱(Standard Name for the Manifest File)

根據 IMS 內容包裝標準，發佈的內容必須包含一 IMS 內容清單檔案。為了確保在包裝中找得到 IMS 內容清單檔案，它必須有預先定義好的命名以及存放位置：

imsmanifest.xml

Content distributed according to the IMS Content Packaging specification must contain an IMS Manifest file. To ensure that the IMS Manifest file can always be found within a Package, it has a pre-defined name and location:

若缺乏此檔案，該包裝就不算是 IMS 包裝且無法被處理。此名稱必須保持以小寫字母組成。

In the absence of this file, the package is not an IMS Package and cannot be processed. It is required that the name be kept, as above, in all lowercase letters.

IMS 內容清單檔案和它所直接引用的 XML 控制檔案(DTD、XSD)皆必須被安置在套裝交換檔案或是任何其他包裝影像檔（像 CD-ROM）的根節點。間接引用到的 XML 控制檔案則必須照命名空間與路徑名稱來放。遠端或（local 本地端）生效之檔案使用方式需依靠安裝的方式。

The IMS Manifest file and any of its directly referenced XML control files (DTD, XSD) must be placed at the root of the Package Interchange File or any other packaging image (like a CD-ROM). XML control files that are indirectly referenced can be located as required by the namespace and path names. The usage of remote or local validation files is implementation dependent.

不過，被使用的本地端檔案必須與線上的相同。若要使用 W3C xml.xsd 的本地副本來執行本地確認，而且此確認程序是在不連續的環境下執行，那麼也將需要本地版（即副本）的搭配檔案：datatypes.dtd 與 XMLSchema.dtd。
被使用的此類 DTD 由 W3C 所提供，可從 W3C 獲得。

However, if local files are used then these must be identical to those online. If “local” validation is going to be performed using a local copy of the W3C xml.xsd and the validation process is going to be done in a “disconnected” environment, then “local” versions (i.e., copy of) of the following files will also be needed: datatypes.dtd and XMLSchema.dtd. These DTDs are used by the W3C provided xml.xsd and can be obtained from the W3C.

4.3 延伸性(Extensibility)

有效支援的延伸性爲IMS Content Packaging標準書之重要基礎。
An important underpinning of the IMS Content Packaging specification is rich support for extensibility.

Content Packaging資訊模型運用IMS Meta-data標準1.2.1版之大量詮釋資料元件，以及IEEE 1484.12.3爲LOM Schema XML Binding（繫結）標準，（關於IEEE LOM標準的實作，請見IMS Meta-Data v1.3 [MD, 04] 最佳實作與指導綱要），本標準僅定義（網站內容）organization及resources之基本結構。

While the base Content Packaging Information Model leverages the rich set of meta-data elements defined in the IMS Meta-Data Specification v1.2.1 or the IEEE 1484.12.3 Standard for eXtensible Markup Language (XML) Schema Binding for Learning Object Metadata (see IMS Meta-Data v1.3 [MD, 04] for best practices and guidelines in implementing the IEEE LOM specification), it defines only the basic structures for organization and resources (Web Content).

本標準在實作時，應定義resources與organizations之新型式，以描述並傳送豐富的學習資源。未來，還可在後續版本中加入常被使用的相關延伸性標準。

It is expected that implementers of this specification will define new types of resources and organizations to describe and transport rich learning resources, and over time, it may be possible to incorporate widely used extensions into future versions of this specification.

4.4 內容清單元件(Manifest Elements)

本節針對內容清單之元件做簡要說明。圖 4.2 表示內容清單之主要元件：

This section provides a conceptual, informative description of the elements contained in a Manifest. Figure 4.2 illustrates the primary elements of a Manifest.

圖 4.2、內容清單元件

Figure 4.2 – Manifest elements.

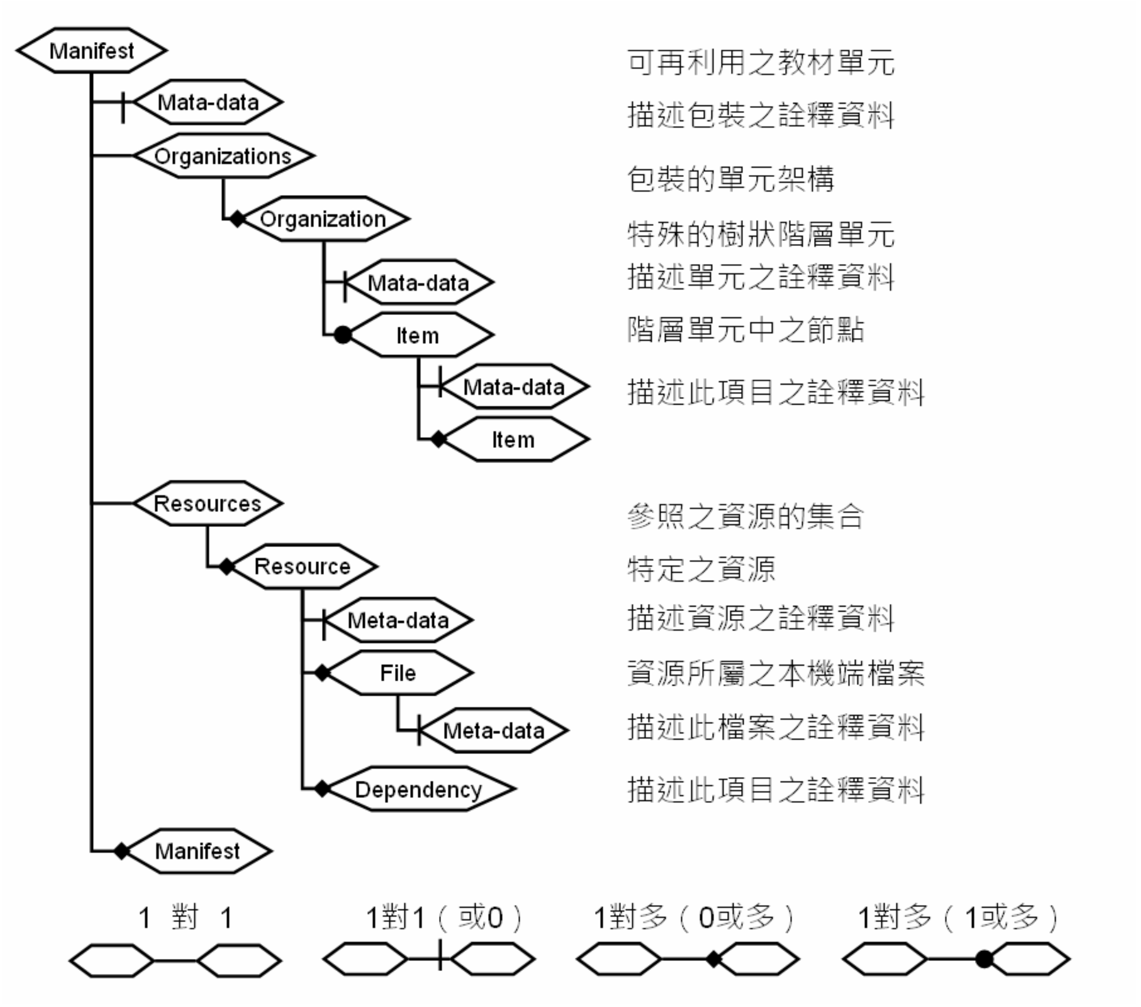


表4.1說明內容物件，表格各欄位說明如下：

Table 4.1 provides a conceptual, informative description of the data objects. The columns used in the table refer to:

編號(No)：	資料元件的編號。元件可能由子元件組成，從結構之編號可顯示其關係。 The number of the data element. An element may be composed of sub-elements. The numbering scheme reflects these relationships.
名稱(Name)：	此元件的描述性名稱。The descriptive name of the element.
說明(Explanation)：	關於此元件的簡要功能敘述。A brief functional description of the element.
必備/選項 (Reqd)：	說明元件爲必備或選項。 Indicates if the element is required. M = 若該元件包含於較高層級之中，則其必須包含於資料物件當中。 mandatory element that must be included in the data object, if the element at the higher level is included. C = 條件元件，其存在與否是根據其他元件的資料值而定。 conditional element, existence is dependent on values of other elements. O = 選項元件(optional element)。
重複(Mult)：	元件的重複次數。元件的重複亦意味著其下所有子元件也會重複。 Multiplicity of the element. Repeatability of an Element implies that all sub-elements repeat with the Element. 空白(-) = 出現一次。 Blank (-) = single instance. 數量 = 表示元件最多可重複之次數。 Number = maximum number of times the element is repeatable. n = 可重複出現，沒有限制。 n = multiple occurrences allowed, no limit.
型式(Type)：	關於資料元件格式規則之敘述：“型式”包含該元件之最大資料長度。欄位內容皆需採用ISO 10646規定之國際字元集。 A description of formatting rules for the data element: Type includes the maximum length of the element. The international character set specified by ISO 10646 will be used for all fields. 容器 = “tag”元件，爲固定資料長度。 Container = ‘tag’ element, of fixed length. ID = 用以表示該物件之唯一識別元件。 ID = element used to uniquely identify an object. IDRef = ID之參考。 IDRef = a reference to an ID. 字串（n）= 描述性元件（最大容許值的下限） String (n) = descriptive element (smallest permitted maximum).

	布林字元 ＝ 是｜否。 Boolean = True | False.

備考：關於元件的其他描述性資訊。

1. 下表中，CP資訊模型中的Manifest元件以大小寫混合模式描述，以便閱讀。此類標準之應用應偏重於特殊的繫結標準，例如：XML繫結方式與W3C一致，所有元件皆使用小寫。
2. 大括號中的元件({})，表示該資訊模型中應包含其他資訊模型或標準元件之處。
3. 表4.1中元件的次序對資訊模型來說並不重要，但相應的XML繫結則會採用此順序為IMS內容清單之需求標準。

Notes: Additional descriptive information about the element.

1) In the table below, the Manifest elements contained in the Content Packaging Information Model are described using mixed case to enhance readability. Implementers of this specification should refer to particular binding specifications. For example, some XML bindings follow the W3C convention of using lowercase for all elements;
2) Elements surrounded by braces ({}) indicate areas in the Information Model where elements from other information models or specifications are expected to be included;
3) The order of elements described in Table 4.1 is not significant from the perspective of the information model. However, the corresponding XML binding imposes this implied order as a requirement for IMS manifests.

表4.1 – 內容包裝資料物件

Table 4.1 - Content packaging data objects.

編號 (No)	名稱(Name)	說明(Explanation)	必備/選項 (Reqd)	重複 (Mult)	型式 (Type)	備註(Note)
1	Manifest	課程可再利用之單位，含詮釋資料(meatadata)、組織(organizations)、與參照之資源(resource references)。 A reusable unit of instruction. Encapsulates meta-data, organizations, and resource references.	M	-	Containe r	
1.1	Identifier	內容清單中包含之唯一識別符。 An identifier that is unique within the Manifest.	M	-	ID	關於識別符的使用，參照最佳實踐指導綱要。 See the Best Practice Guide for guidelines

						on the use of identifiers.
1.2	Version	此內容清單之版本（例如：1.0） Identifies the version of this Manifest (e.g., 1.0).	O	-	String (20)	用於包裝(Package)有修改時。兩份內容清單檔之識別符皆相同。 Used to identify if there have been any changes to the Package. Identifier is the same in two Manifest files.
1.3	Xml:base	提供內容檔案之相關路徑。 Provides the relative path offset for the content file(s).	O	-	String (2000)	‘href’ 與‘xml:base’最大字元長度皆爲2000個位元組。若‘xml:base’之值超過2000，則系統行爲並未定義。 The maximum length of both ‘href’ and ‘xml:base’ is defined as 2000 octets. In cases where multiple ‘xml:base’ values need to be concatenated to create the full path then care must be taken to ensure that the total length does not exceed that of the ‘href’. If the path length is greater than 2000 octets then the system behavior is undefined
1.4	Meta-data	描述內容清單之詮釋資料。 Meta-data describing the Manifest.	O	-	Container	
1.4.1	Schema	描述定義與控制內容清單之方案。 Describes the schema that defines and controls the Manifest.	O	-	String (100)	若未顯示方案元件，則預設爲“IMS Content”。 If no schema element is present, it is assumed to be “IMS Content”.
1.4.2	SchemaVersion	描述方案之版本（例如：1,0,1.1） Describes the version of the above schema (e.g., 1,0, 1.1).	O	-	String (20)	若無版本說明，則預設爲“1.1”版。 If no version is present, it is assumed to be “1.1”
1.4.3	{Meta-Data}	採用合適的資訊模型置入詮釋資料 This is where Meta-data is inserted using an appropriate information model.	O	n	-	本節預設值爲IMS詮釋資料標準中之定義 By default, the information contained in this section is

						defined by the IMS Meta-Data specification.
1.5	Organizations	描述此包裝之一或多個結構或單元 Describes one or more structures or organizations for this package.	M	-	Container	可由詮釋資料來選定單元 An organization can be selected by its meta-data.
1.5.1	Default	預設之單元架構 Indicates which organization scheme is the default one.	O	-	IDRef	使用時需直接指向內容清單之子<organization>。例如：不可指向子內容清單之<oraganization>。若未提供，第一個出現的organization元件則視為預設值。 When used, this must point to a direct child <organization> in the manifest i.e., it cannot point to an <organization> in a sub-manifest. If not supplied, the first organization element encountered is assumed to be the default.
1.5.2	Organization	特定階層之單元 Describes a particular hierarchical organization.	O	n	Container	以不同角度或單元路徑內容呈現內容，描述單元之多重情況 Different views or organizational paths through the content can be described using multiple instances of organization.
1.5.2.1	Identifier	單元之識別符，於內容清單檔案中為唯一。 An identifier, for the organization, that is unique within the Manifest file.	M	-	ID	關於識別符之使用，參照最佳實作指導綱要。 See the Best Practice Guide for guidelines on the use of identifiers.
1.5.2.2	Structure	有階層式之預設值，用以說明單元之型態。 Has a default value of “hierarchical” for describing the shape of an organization.	O	-	String (200)	其他值將列入未來標準之中 Other values for structure will likely become part of a future specification.
1.5.2.3	Title	單元之標題 Describes the title of the organization.	O	-	String (200)	協助使用者決定單元之選擇Used to

						help user decide which organization to choose.
1.5.2.4	Item	單元型態之節點 A node that describes the shape of the organization.	M	n	Container	用於排序或巢狀之階層式架構 Can be used in a hierarchical organizational scheme by ordering and nesting.
1.5.2.4.1	Identifier	項目之識別符，於內容清單檔案中爲唯一。 An identifier, for the Item, that is unique within the Manifest file.	M	-	ID	關於識別符之使用，參照最佳實作指導綱要。 See the Best Practice Guide for guidelines on the use of identifiers.
1.5.2.4.2	IdentifierRef	資源段落或子內容清單中參照之識別符 A reference to an identifier in the resources section or a sub-Manifest.	O	-	String (2000)	
1.5.2.4.3	Title	項目之標題 Title of the item.	O	-	String (200)	
1.5.2.4.4	IsVisible	當包裝之結構顯示或不顯示時，說明該項目是否出現。 Indicates whether or not this item is displayed when the structure of the Package is displayed or rendered.	O	-	Boolean	若不出現，預設值爲“true”，此值只會影響該項目，不影響與項目之子項、該項目相關之資源。 If not present, value is assumed to be “true”. This value only affects the item for which it is defined and not the children of the Item or a resource associated with an Item.

編號 (No)	名稱 (Name)	說明 (Explanation)	必備/選項 (Reqd)	重複 (Mult)	型式 (Type)	備註(Note)
1.5.2.4.5	Parameters	啓動時所傳遞之靜態參數。 Static parameters to be passed to the resource at launch time.	O	-	String (1000)	#<parameter> <name>=<value>(&<name>=<value>)*(#<parameter>) ?<name>=<value>(&<name>=<value>)*(#<parameter>) <parameter>爲實作時定義字串值 Where <parameter> is some implementation defined characterstring value <name>爲實作定義名稱 Where <name> is some implementation defined name

						<value>為實作時定義值 Where <value> is some implementation defined value “=”用以區隔<name>和<value>對。 The "=" is required to separate the <name> and <value> pair “&”用以區分多組<name>和<value>對。 The "&" is required to separate multiple sets of <name> and <value> pairs (&<name>=<value>)*意即0或多個<name>和<value>對可連結在一起。 The (&<name>=<value>)* indicates that the 0 or more <name> and <value> pairs can be concatenated together 備考：參數欄位所使用之字串需為URL碼。RFC2396定義URLs編碼之規則及規範。 NOTE: The characters used in the parameters field may need to be URL encoded. RFC 2396 defines the rules and requirements for encoding URLs.
1.5.2.4.6	Item	該單元之子節點 A sub-node within this organization.	O	n	Container	此為子項目，且需重複<item>之內容。 This is a sub-item and repeats all the parts of <item>.
1.5.2.4.7	Meta-data	項目之詮釋資料 Meta-data describing this item.	O	-	Container	參照第1.4.3節item之說明。 See item 1.4.3 above
1.5.2.4.7.1	{Meta-Data}	採用合適的資訊模型置入詮釋資料 This is where Meta-data is inserted using an appropriate information model.	O	n	-	本節預設值為IMS詮釋資料標準中之定義 By default, the information contained in this section is defined by the IMS Meta-Data specification.
1.5.2.4.8	Meta-data	該組織之詮釋資料Meta-data describing this organization.	O	-	Container	參照第1.4.3節item之說明。 See item 1.4.3 above
1.5.2.4.8.1	{Meta-Data}	採用合適的資訊模型置入詮釋資料 This is where Meta-data is inserted using an appropriate	O	n	-	本節預設值為IMS詮釋資料標準中之定義 By default, the information contained in this section is defined by the IMS Meta-Data specification.

		information model.				
1.6	Resources	參照資源之集合。無預設順序或階層。 A collection of references to resources. There is no assumption of order or hierarchy.	M	-	Container	
1.6.1	Xml:base	提供內容檔案之相關路徑。 Provides the relative path offset for the content file(s).	O	-	String (2000)	
1.6.2	Resource	參照之資源 A reference to a resource.	O	n	Container	
1.6.2.1	Identifier	關於資源之識別符，於內容清單檔案範圍中為唯一。 An identifier, of the resource, that is unique within the scope of its containing manifest file.	M	-	ID	關於識別符之使用，參照最佳實作指導綱要。See the Best Practice Guide for guidelines on the use of identifiers.

編號(No)	名稱(Name)	說明(Explanation)	必備/選項(Reqd)	重複(Mult)	型式(Type)	備註(Note)
1.6.2.2	Type	資源之型式 Indicates the type of resource.	M	-	String (1000)	唯一現有型式為“網路內容”，其定義為能由網路瀏覽器（HTML內容、需由外掛程式，如：Flash、Real Media）開啓者；此標籤定義於“Using IMS Content Packaging to Package Instances of LIP and Other IMS Specifications, Version 1.0 Implementation Handbook”第七節（2001八月出版）。 The only current types are: - “webcontent”, defined as content that can be hosted in or launched by an Internet browser (this includes

						HTML-based content, content that requires plug-ins e.g., Flash, Real Media and executables that are launched by a browser); - The labels defined in Section 7 of the document 'Using IMS Content Packaging to Package Instances of LIP and Other IMS Specifications, Version 1.0 Implementation Handbook, August 2001'; Note that IMS specifications and versions of IMS specifications that had not yet been released at the time of writing of this document, should follow the syntax as specified in section 7.’
1.6.2.3	Href	URL之參照 A reference to a URL.	O	-	String (2000)	若URL亦依據<item>之“parameter”屬性參照<resource>，則應以第4.2節所定義之演算法建立完整URL。“Href”值的語法需符合RFC2396。 If the URL is also based upon the contents of the ‘parameter’ attribute of the <item> referencing the <resource> then the algorithm defined in Section 4.2 should be used to construct the full URL. The syntax for the ‘Href’ value must conform to RFC2396.
1.6.2.4	Xml:base	提供內容檔案之相關路徑。 Provides the relative path offset for the content file(s).	O	-	String (2000)	參照第1.4.3節item之說明。 See item 1.3 above.
1.6.2.5	Meta-data	該資源之詮釋資料 Meta-data describing this resource.	O	-	Container	參照第1.4.3節item之說明。 See item 1.4.3 above
1.6.2.5.1	{Meta-Data}	採用合適的資訊模型置入詮釋資料 This is where Meta-data is inserted using an appropriate	O	n	-	本節預設值爲IMS詮釋資料標準中之定義 By default, the information contained in this section is defined by the IMS Meta-Data specification.

		information model.				
1.6.2.6	File	資源所屬之檔案列表 A listing of files that this resource is dependent on.	O	-	Container	指向該資源所屬之單一檔案，特定資源之各檔案可重複。 An element identifying a single file this resource is dependent on. Repeat as needed for each file for a given resource.
1.6.2.6.1	Href	檔案位置 Identifies the location of the file.	M	n	String (2000)	“Href”值的語法需符合RFC2396 The syntax for the ‘Href’ value must conform to RFC2396.
1.6.2.6.2	Meta-data	該檔案之詮釋資料 Meta-data describing this file.	O	-	Container	參照第1.4.3節item之說明。 See item 1.4.3 above.
1.6.2.6.2.1	{Meta-Data}	採用合適的資訊模型置入詮釋資料 This is where Meta-data is inserted using an appropriate information model.	O	n	-	本節預設值爲IMS詮釋資料標準中之定義 By default, the information contained in this section is defined by the IMS Meta-Data specification.
1.6.2.7	Dependency	資源所屬之檔案 Identifies a resource whose files this resource depends upon.	O	n	IDref	該元件指稱單一資源作爲該資源所屬之多個檔案之容器。 This element identifies a single resource which can act as a container for multiple files that this resource depends upon.
1.6.2.7.1	IdentifierRef	資源段落參照之識別符 A reference to an identifier in the resources section.	M	-	String (2000)	此識別符無法參照子內容清單之識別符 This indentifierref cannot reference an identifier in a sub-manifest.
1.7	Manifest	課程之可再利用單元，封入詮釋資料、單元與參照之資源。 A reusable unit of instruction. Encapsulates meta-data, organizations, and resource references.	O	n	Container	參照第4.1節子內容清單。 See section 4.1 sub-Manifests below for more information.

4.4.1 子內容清單（sub-Manifests）

當<organization>之<item>的identifierref 參照爲一子內容清單，而非參照其他型式之資源

時，應解譯如下：

若子內容清單沒有包含任何<organization>，參照就無法解析，此應視同爲一空值參考識別符（null identifierref）。

When the identifierref of an <item> in an <organization> references a sub-Manifest rather than another type of resource, this shall be interpreted as follows:

If the sub-Manifest does not include any <organization>, the reference cannot be resolved. This shall be treated as a null identifierref;

若子內容清單包含<organization>，則該organizaton之根節點（例如：<organization>元件本身）應合併內容清單參照之<item>。若<item>與<organization>參照同樣的屬性，但有不同值時，<organization>應優先於<item>之值。亦即，子屬性優先於父屬性。此動作會顯現於瀏覽樹狀清單中，但不會影響內容清單之XML呈現。

If the sub-Manifest includes an <organization>, the root node of that organization (i.e., the <organization> element itself) shall merge with the <item> that references the sub-Manifest. If the same attribute is specified for both the <item> and the <organization> it references, but with different values, the value defined for the referred <organization> manifest shall override the value defined for the referring <item>. That is, child attributes take precedence over parent attributes. This behavior is expected of the rendering of the navigation tree, but does not need to affect the XML of the manifest(s).

圖4.3以瀏覽樹狀清單顯示子內容清單之內容如何合併參照到的內容清單。圓圈代表organization結構中的項目（items）。在此範例中，子內容清單之organization沒有標題屬性。

The diagram in Figure 4.3 explains how the content of a sub-Manifest is merged with the content of a referencing manifest in the rendering of a navigation tree. The circles represent items in an organization structure. In this example, the organization of the sub-manifest does not have a title attribute.

圖 4.3 –　由 子 內 容 清 單 合 併 <organization>
Figure 4.3 - Merging <organization> from a sub-Manifest.

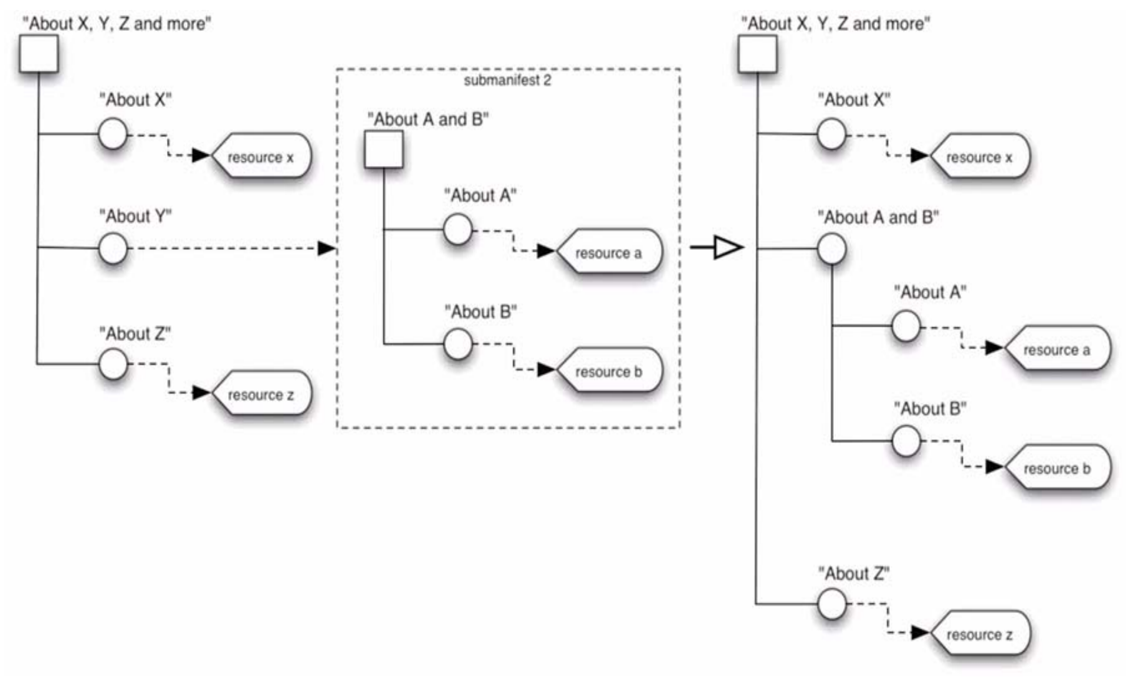


圖4.4顯示子內容清單中，<organization>有title屬性之例子。

The diagram in Figure 4.4 shows an example where the <organization> of the sub-Manifest does have a title attribute.

圖 4.4 – 由 子 內 容 清 單 合 併 <organization>屬 性
Figure 4.4 Merging an attributeless <organization> from a sub-Manifest.

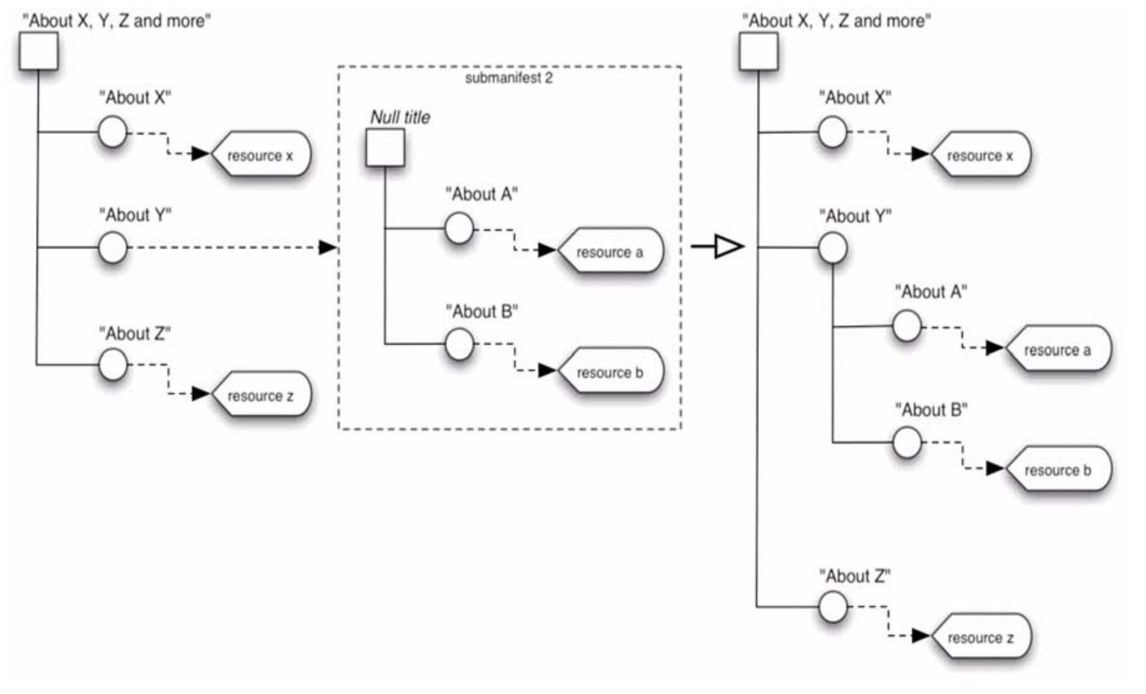
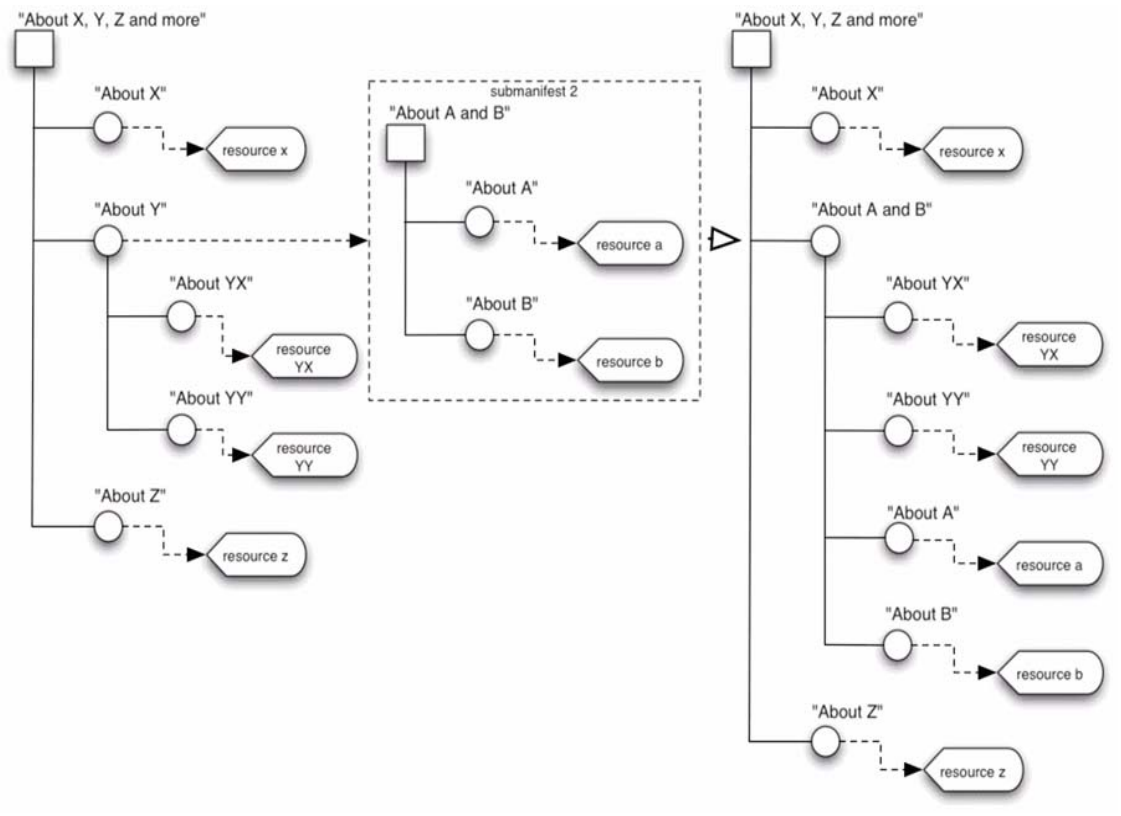


圖4.5中說明了子內容清單之內容如何合併一有子節點的項目（item），同時也說明參照之子節點優先於其他合併項目。在此範例中，子內容清單之組織沒有標題屬性。

In Figure 4.5, the diagram explains how the content of a sub-Manifest is merged with an item that has children of its own, and how the referencing item's children take precedence over those they merge with. In this example, the organization of the sub-Manifest does not have a title attribute.

圖 4.5由 子 內 容 清 單 合 併 <organization>，參照之項目所有子項

Figure 4.5 Merging <organization> from a sub-Manifest when the referencing item has children.



4.4.2 Href URL建立演算法(Href URL Construction Algorithm)

本例中，“Href”中所填的完整URL參照是依據<item>欄位之“parameter” 屬性的參數，用以建立完整的URL：

In the cases where full URL referenced in the ‘Href’ value is also based upon the parameters passed in the ‘parameter’ attribute of the <item> referencing the <resource> then the following algorithm is to be used to construct the full URL:

```
While  first char of parameters is in”?&”
    Clear first char of parameters
If  first char of parameters is “#”
  If the URI contains “#”
      Discard parameters
  Else
      Append parameters to the URI
   Done processing URI
If the URI contains “?”
   Append “&” to the URI
```

```
Else
    Append “?” to the URI
Append parameters to the URIDone processing the URI
```

此演算法的定義爲標準規範。

The definition of this algorithm is normative.

5.內容包裝 XML 繫結

5.1 XML 基礎 (XML Basics)

內容包裝資料模型以階層方式來定義。階層性的模型對於表達具有許多元件及子元件之資料來說較爲簡便。XML能適合呈現階層式之模型。其係爲元件組合而成的階層式文件，元件內含內容及屬性。

5.IMS Content Packaging XML Binding

5.1 XML Basics

The Content Packaging data model can be defined as a hierarchy. Hierarchical models are convenient for representing data consisting of many elements and sub-elements. XML is perfectly suited for representing hierarchical models. An XML document is a hierarchy comprised of elements that have contents and attributes.

5.1.1 元件(Elements)

元件係爲文件之組件，且應以電腦能瞭解的方式來識別。各元件皆有一標籤名稱。標籤名稱以 “<TAGNAME>”方式出現（標籤名稱之前及之後出現大於/小於符號）時，即代表此爲該元件之起始標籤。當標籤名稱爲“</TAGNAME>”方式出現，則表示此爲結束標籤。元件的起始與結束標籤之間可能有內容，有時伴隨一或多個屬性，當XML元件有成對的開始與結束標籤（亦稱開放與封閉標籤），即視爲“well-formed” XML。開始與結束標籤中包含元件內容的顯示方式如下：

<TAGNAME>contents</TAGNAME>

An element is a component of a document that has been identified in a way a computer can understand. Each element has a tag name. When a tag name is shown as “<TAGNAME>”, with less-than and greater-than symbols before and after the tag name, it serves as the start-tag to mark the beginning of an element. When that same tag name has a forward slash “/” added, it serves as an end-tag such as “</TAGNAME>”. An element may have contents between its start and end-tags, and may have one or more attributes. When an XML element has a start and end-tag (also called an opening and closing tag) with a common name, it is considered to be “well-formed” XML. The contents of an

element are placed between the start and end-tags as shown below:
<TAGNAME>contents</TAGNAME>

(1)元件內容(Element Contents)

元件中可能會包含其他元件、Parsed Character Data (PCDATA)、Character Data (CDATA)，或是同時混合元件及PCDATA。元件之可允許內容爲該元件的內容模型。PCDATA實際上表示不包含元件的任何字串字元，CDATA爲大多數元件使用的形式，因其能輸入任何字元資料而不會被剖析處理，例如：在CDATA段落中加上JavaScript程式碼指令。處理器會略過CDATA中的任何標記，直到該段落結束爲止。

An element may contain other elements, Parsed Character Data (PCDATA), Character Data (CDATA), or a mixture of PCDATA and elements. The allowable contents of an element are its content model. PCDATA really means any character string that does not contain elements. PCDATA is what the bulk of elements will use between their start and end-tags. CDATA is different in that it is a method for adding any character data that should not be processed. For example, you could add some JavaScript code instructions using a CDATA section. A CDATA section tells the parser not to look for any mark-up until after it locates the end of the CDATA section.

(2)元件屬性(Element Attributes)

屬性提供關於該元件的額外資訊。屬性係爲附加字元，或爲文件中元件的性質。屬性可能不只一個，元件可能包含一個以上的屬性。屬性標明於元件之起始標籤之中，其由屬性名稱“=”符號及屬性值的形式表示：

An attribute provides additional information about an element. Attributes are a way of attaching characteristics or properties to the elements of a document. An element may have more than one attribute. Attributes are contained within the start tag of an element. Attributes are represented by an attribute name followed by an equal sign and the attribute value in quotation marks:

```
<timeframe>
    <begin restrict="1"> 1999-07-23 </begin>
</timeframe>
```

上例中，<timeframe>元件包含<begin>元件，<begin>元件有一個“restrict”之屬性，“restrict”屬性的值爲“1”，而<begin>元件的屬性值則爲“1999-07-23”。 此兩元件組成開始時間。

In this example, the <timeframe> element contains another element: the <begin> element. The <begin> element has one attribute ‘restrict’, with the value “1”. The value for the element <begin> is “1999-07-23”. These two elements then make up a timeframe begin date.

(3)元件名稱(Element Names)

元件的標籤係爲唯一的命名，且大小寫有別，以XML剖析該標籤名稱。IMS Content Packaging XML Binding需遵守下列標籤名稱規則：

Each element has a unique name, referred to as the tag name. XML is case-sensitive in its processing of tag names.
The IMS Content Packaging XML Binding adheres to the following tag name rules:

(a) 所有標籤名稱需遵守XML 1.0版標準所定義之元件名稱規則。
(b) 不允許以XML作爲名稱開頭。
(c) IMS binding僅能使用小寫的標籤及元件名稱。
(d) 元件名稱不得包含XML標準中之保留字元，包括：
    DOCTYPE
    ELEMENT
    ATTLIST
    ENTITY
(e) 已由IMS binding定義之標籤名稱不得重複定義。

(a) All tag names will conform to the rules for element naming as given within the XML 1.0 Specification.
(b) Names beginning in XML in any case or mix of cases are not permitted.
(c) The IMS binding will use only lowercase tag and element names.
(d) Element names may not include words reserved by the XML specification.
    These include:
    DOCTYPE
    ELEMENT
    ATTLIST
    ENTITY
(e) Tag names defined by the IMS binding may not be redefined.

5.1.2 文件型式定義(Document Type Definitions)

標籤名稱、內容模型、元件屬性皆定義於文件型式定義（Document Type Definition ,DTD）宣告，此係爲外部檔案、或內含於XML文件的文字。外部DTD定義之元件通常優先於內部DTD，所以需要格外注意。DTD定義了可以

使用的元件，同時也可以用以定義元件的內容。

The tag name, content model, and attributes of elements are defined in a Document Type Definition (DTD) statement. This may exist as an external file or a block of text internal to an XML document. Internal DTDs are used to override elements defined in external DTD files, so an internal DTD should be used with care. The DTD defines the elements that may be used and may define the contents of the elements.

本標準定義DTD (imscp_rootv1p1.dtd) 僅作爲參考之用，在本標準中，不列爲正式規範。有些XML編輯者會利用DTD以協助引導發展者在XML檔案中適當位置產生合適的元件，有些發展者會利用DTD驗證XML文件，以確保定義於DTD之元件名稱及位置是一致的。建立DTD的細節並非本標準的範圍，但第1.5節會提及如何將DTD與XML 1.0版標準做連結。

This specification defines a DTD (imscp_rootv1p1.dtd) as a non-normative reference. Some XML editors may make use of a DTD to help guide the developer in creating the proper elements at the proper locations in an XML file. Other developers will make use of DTDs to validate their XML documents to ensure their document is consistent with all of the element names and locations defined in the DTD. Details of the construction of DTDs are outside the scope of this document, but links to the XML 1.0 Specification are included in section 1.5 of this document.

5.1.3 XML 架構(XML Schemas)

架構(schema)係爲元件名稱之正式標準，說明XML文件中所允許之元件與元件的組合。新的架構語言(schema language)如XML架構工作小組定義的內容，提供了如同DTD般的基本功能。但由於架構語言具有延伸性，可讓開發者能增加額外的資訊，例如：資料型式、繼承及呈現規則等，使得新的架構語言相較於DTD來說，具有更強的能力。關於XML架構的相關資訊，請參照W3C第1.5節XML Schema Recommendation的連結。

A schema is a formal specification of element names that indicates which elements are allowed in an XML document, and in which combinations. New schema languages, such as those defined in the XML-Schemas Working Group, provide the same baseline functionality as a DTD. However, because these schema languages are extensible, developers can augment them with additional information, such as data types, inheritance, and presentation rules. This makes these new schema languages far more powerful than DTDs. For more information about XML schemas, there is a link to the W3C XML Schema

Recommendation in section 1.5.

本標準定義之W3C XML 架構僅作爲參考之用，在本標準中，不列爲正式規範。有些XML編輯者會利用schema以協助引導發展者在XML檔案中適當位置產生合適的元件，有些發展者會利用schema驗證XML文件，且能定義IMS Content Packaging Binding之延伸性。建立schema的細節並非本標準的範圍。

This specification defines a W3C XML Schema as a non-normative reference. Some XML editors may make use of schemas to help guide the developer in creating the proper elements at the proper locations in an XML file. Other developers will make use of schemas to validate their XML documents and/or to define extensions to the IMS Content Packaging Binding. Details of the construction of schemas are outside the scope of this document.

5.1.4 合法字元集(Valid Character Sets)

內容包裝紀錄需採用ISO 10646定義之UTF-8或UTF-16編碼作爲字元集，參照XML 1.0版以了解何謂well-formed XML。

A Content Packaging record must use UTF-8 or UTF-16 encoding of the character sets as defined in ISO 10646. See the XML Version 1.0 for more details on the specification of well-formed XML.

5.1.5 特殊操作需求（Special Handling Requirements）

(1) XML保留字元（XML Reserved Characters）

XML 1.0標準中第2.4節規定，在XML定義用途之外，若使用到XML字元，皆必須避開。此類字元爲(&)、(<)、(>)、(‘)與(“)。

Some characters used in XML must be escaped when used outside of their XML-defined usage as found in section 2.4 of the XML 1.0 Specification. These characters are ampersand (&), less than (<), greater than (>), apostrophe (‘), and the double-quote character (“).

若需使用此類字元，可用數字字元表示，或是以字串“&”、“<”、“>”、“&apos;”、“””方式表示。

These characters may be represented using either numeric character references or the strings “&”, “<”, “>”, “&apos;”, and “””.

下面爲W3C XML標準之完整引用：

Below is a more complete quote from the W3C XML standards:

Quote from Extensible Markup Language (XML) 1.0
W3C Recommendation 10-February-1998
2.4 Character Data and Markup
“正文由字元資料及標記混和而成。標記以開始標籤、結束標籤、空元件標籤、個體參照、字元參照、註解、CDATA段落區隔字元及文件型式宣告，以及處理指令等型式組成。

“Text consists of intermingled character data and markup. Markup takes the form of start-tags, end-tags,empty-element tags, entity references, character references, comments, CDATA section delimiters, document type declarations, and processing instructions.

“所有非標記的正文組成文件的字元資料。
“ (&)符號與小於符號(<)只有在作爲區隔字元、註解、處理指令、或者CDATA段落時才具有字面上的意義。作爲外部個體宣告之個體值時，上述符號亦可視同字面上之意義；請參照原標準“4.3.2 Well-Formed Parsed Entities”。若用於他處，必須以數字字元或個別的“&”與“<”字串以避免被省略。當字串不是用以標示CDATA段落的結束時，大於符號(>)須用字串“>”表示，且爲求一致，一定要用“>”，或是在內文出現字串“]]>”時使用字元參考。

“All text that is not markup constitutes the character data of the document.
“The ampersand character (&) and the left angle bracket (<) may appear in their literal form only when used as markup delimiters, or within a comment, a processing instruction, or a CDATA section. They are also legal within the literal entity value of an internal entity declaration; see "4.3.2 Well-Formed Parsed Entities". If they are needed elsewhere, they must be escaped using either numeric character references or the strings "&" and "<" respectively. The right angle bracket (>) may be represented using the string ">", and must, for compatibility, be escaped using ">" or a character reference when it appears in the string "]]>" in content, when that string is not marking the end of a CDATA section.

“在元件內容中，字元資料可以是任意字串，其中並不包含任何標記之啓始字元。在CDATA 段，字元資料則是不包含CDATA段落結束字元—"]]>"之任意字串。
“爲允許屬性値內含單引號與雙引號，單引號(')需以“&apos;”方式出現，而雙引號(")則表示爲""".”。

“In the content of elements, character data is any string of characters, which does not contain the start-delimiter of any markup. In a CDATA section, character data is any string of characters not including the CDATA-section-close delimiter, "]]>".
“To allow attribute values to contain both single and double quotes, the apostrophe or single-quote character (') may be represented as "&apos;", and the double-quote character (") as """.”

(2) 空白字元操作(White Space Handling)
XML檔案在系統傳輸時，網路資料傳輸工具常省去或改變空白字元，此種情況容易造成問題。為避免此類情形，參考以下引用之W3C XML標準，說明資料中出現之空白字元應保留的處理方式。

Questions often arise as to whether Web-based data transmission tools might inadvertently strip off or transform some of the white space characters embedded in data transmitted between systems using XML. To eliminate concern about this issue, refer to the following quote from the W3C XML standards, which indicate that all white space must be preserved where it is part of the data.

Quote from Extensible Markup Language (XML) 1.0
W3C Recommendation 10-February-1998
2.10 White Space Handling

“編輯XML文件時，常會使用“空白字元(white space)”（空白鍵、tab鍵、空白列、在標準中以非結束字元的S標示）以示區別。此種空白字元通常不用在文件傳送的版本中。另一方面，“明顯的(significant)”空白字元一般則會保留，例如：詩歌與程式碼中。

“In editing XML documents, it is often convenient to use "white space" (spaces, tabs, and blank lines, denoted by the non-terminal S in this specification) to set apart the mark-up for greater readability. Such white space is typically not intended for inclusion in the delivered version of the document. On the other hand, "significant" white space that should be preserved in the delivered version is common, for example in poetry and source code.

“XML處理器需傳遞文件中所有的未標記字元給應用程式。驗證XML的處理

器亦會通知應用程式出現於元件內容裡頭之空白字元。

“An XML processor must always pass all characters in a document that are not mark-up through to the application. A validating XML processor must also inform the application which of these characters constitute white space appearing in element content.

“特別的xml:space屬性會附加於元件中，以指明空白字元在該元件中之意含。應用程式應保留其空白字元。合法驗證的文件中，此屬性（如同其他屬性）,若使用到空白字元需先宣告,以列舉方式的宣告,其值只有“default”與“preserve”舉例來說：

“<!ATTLIST poem xml:space (default|preserve) 'preserve'>

““default”一值意指此元件可接受應用程式預設空白字元處理模式；

““preserve”一值意指應用程式應保留所有空白字元。此項宣告表示，除非置換其他份xml:space屬性，否則所有元件應以內文規定的方式呈現”。

“A special attribute named xml:space may be attached to an element to signal an intention that in that element, white space should be preserved by applications. In valid documents, this attribute, like any other, must be declared if it is used. When declared, it must be given as an enumerated type whose only possible values are

“<!ATTLIST poem xml:space (default|preserve) 'preserve'>

“The value "default" signals that applications' default white-space processing modes are acceptable for this element; the value "preserve" indicates the intent that applications preserve all the white space. This declared intent is considered to apply to all elements within the content of the element where it is specified, unless overridden with another instance of the xml:space attribute."

5.2.XML 繫結敘述（Narrative Description of XML Binding）

本標準使用敘述方式定義XML格式，用以實作此抽象格式之XML DTD與XML架構僅作爲本標準參考之用，在本標準中，不列爲正式規範。

This specification defines the XML format using narrative. XML DTDs and XML Schemas that implement this abstract format are referenced as non-normative parts of this specification.

5.2.1 <manifest>元件(<manifest> Elements)

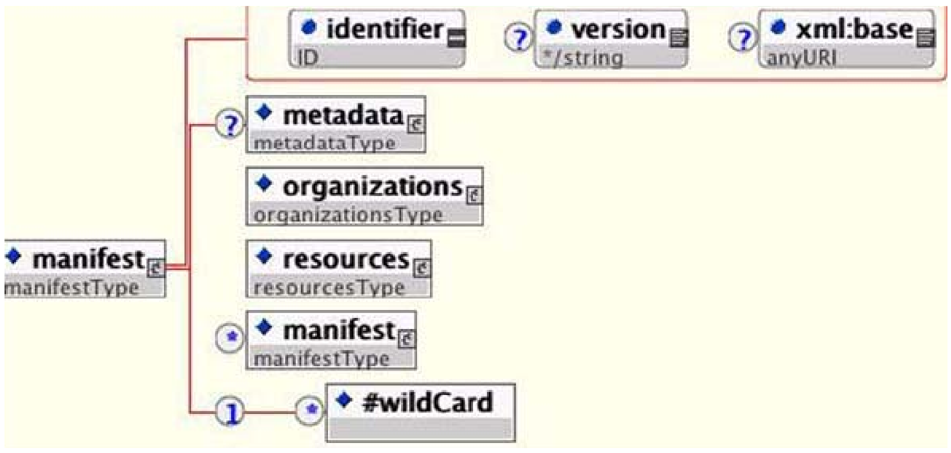
(1)說明

內容清單中的最上層（第一層）<manifest>元件將所有參考資料包入。在此<manifest>元件之後出現的<manifest>元件用以區分檔案、詮釋資料、聚合結構(organization structure for aggregation)、解構及再利用。IMS Content Packaging標準之實作會將每個“學習物件(learning object)”或“最小學習單位(atomic unit of learning)”放在其<manifest>元件之內。

Description. The first, top-level <manifest> element in the Manifest encloses all the reference data. Subsequent occurrences of the <manifest> elements inside the top-level <manifest> are used to compartmentalize files, meta-data, and organization structure for aggregation, disaggregation, and reuse. The best-practice use of the IMS Content
Packaging specification will result in each “learning object” or “atomic unit of learning” being placed within its own <manifest> element.

圖5.1 <manifest> 元件

Figure 5.1 <manifest> elements.



(2)重複性(Multiplicity)

在IMS Manifest檔中，最上層<manifest>出現一次且係唯一。

The top-level <manifest> occurs once and only once within the IMS Manifest file.

(3)屬性(Attributes)

(a) identifier (必備)。由編輯者或編輯工具提供的識別符，在內容清單中係唯一。資料型式 = 字串；

identifier (required). An identifier, provided by an author or authoring tool, that is unique within the Manifest.

Data type = string;

(b) version (選項)。標示內容清單的版本，與identifier共同用以區別不同的內容清單。資料型式 = 字串；

version (optional). Identifies the version of the Manifest. Is used to distinguish between manifests with the same identifier.
Data type = string;

(c) xml:base (選項)。提供內容檔案之相對路徑，關於此元件之使用定義於W3C編制的XML Base工作草案。資料型式 = 字串；

xml:base (optional). This provides a relative path offset for the content file(s). The usage of this element is defined in the XML Base Working Draft from the W3C. Data type = string.

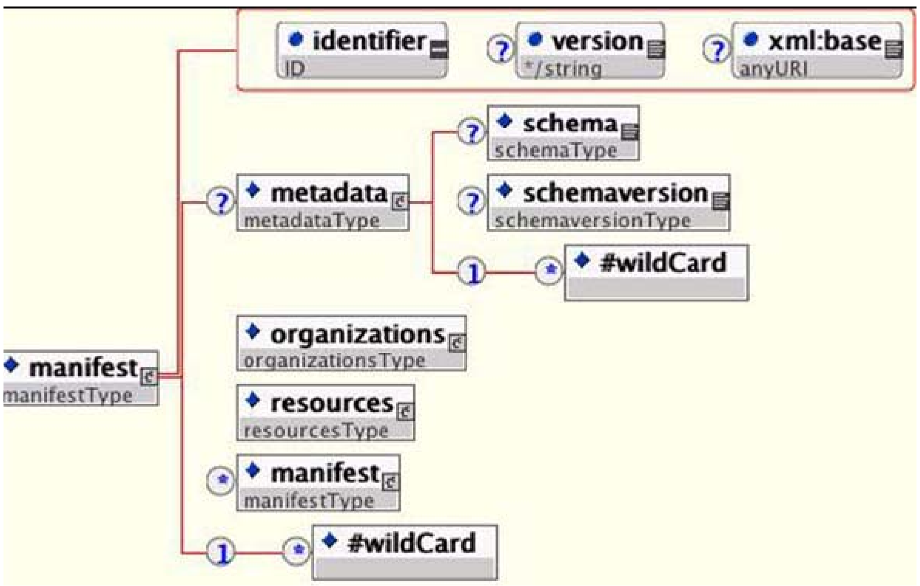
(4)元件(Elements)

(a) <metadata>
(b) <organizations>
(c) <resources>
(d) <manifest>

5.2.1.1 <metadata>

圖5.2 <metadata>元件

Figure 5.2 <metadata> elements.



(1)重複性(Multiplicity)

在<manifest>元件中出現0或1次。

Occurs zero or once within a <manifest> element.

(2)元件(Elements)

(a) <schema>

(b) <schemaversion>

(c) 詮釋資料(Meta-Data)：開發者可自由選擇IMS Meta-data標準或其他詮釋資料標準所定義之詮釋資料元件。

Meta-Data: Implementers are free to choose from any of the meta-data elements defined in the IMS Meta-Data specification or other meta-data standards.

(3)範例(Example)

```
<metadata>
  <schema>IMS Content</schema>
  <schemaversion>1.1</schemaversion>
  <imsmd:lom>
    <imsmd:general>
      <imsmd:title>
        <imsmd:langstring xml:lang="en-US">Simple
      Manifest</imsmd:langstring>
      </imsmd:title>
    </imsmd:general>
  </imsmd:lom>
</metadata>
```

5.2.1.2 <organizations>

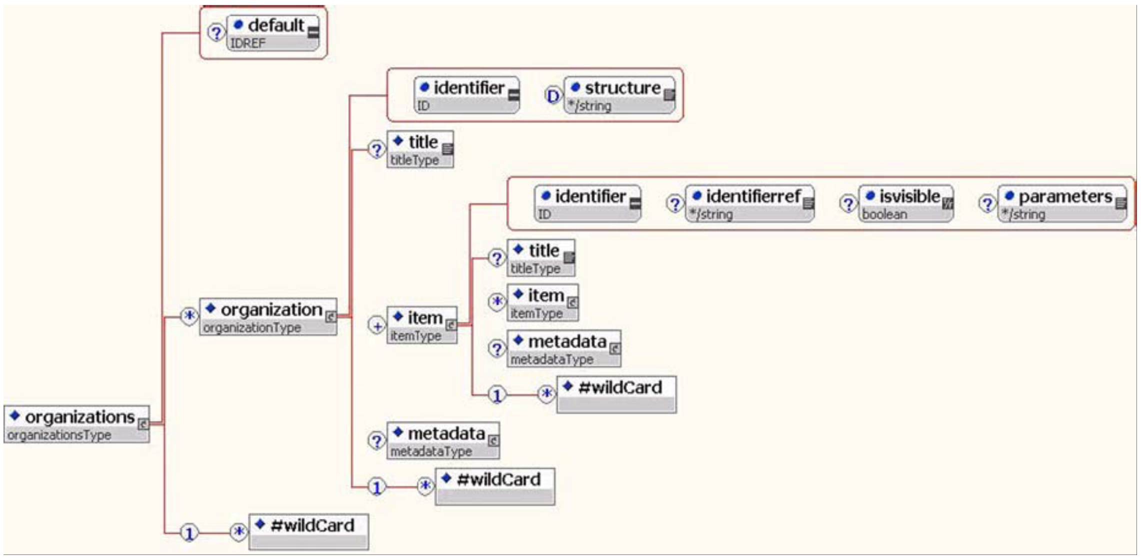
(1)說明(Description)

描述該套裝中的0個、1個或多個結構/組織（例如：<organization>元件）。

Describes zero, one, or more structures or organizations (i.e., <organization> elements) for this package.

圖5.3 <organizations>元件

Figure 5.3 <organizations> elements.



(2)重複性(Multiplicity)

在<manifest>元件中僅能出現1次。

Occurs once within a <manifest> element.

(3)屬性(Attributes)

(a) default (選項)。預設使用的organization。資料型式 = 字串；

default (optional). Identifies the default organization to use. Data type = idref.

(4)元件(Elements)

<organization>

(5)範例(Example)

```
<organizations default="TOC1">
  <organization identifier="TOC1" structure="hierarchical">
    <title>default</title>
    <item identifier="ITEM1" identifierref="RESOURCE1"
    isvisible="true">
      <title>Lesson 1</title>
      <item identifier="ITEM2" identifierref="RESOURCE2"
      isvisible="true">
        <title>Introduction 1</title>
      </item>
      <item identifier="ITEM3" identifierref="RESOURCE3"
      isvisible="true">
```

```
              <title>Content 1</title>
            </item>
            <item identifier="ITEM4" identifierref="RESOURCE4"
            isvisible="true">
              <title>Summary 1</title>
            </item>
          </item>
          <item identifier="ITEM5" identifierref="RESOURCE5"
          isvisible="false">
            <title>Lesson 2</title>
            <item identifier="ITEM6" identifierref="RESOURCE6"
            isvisible="false">
              <title>Introduction 2</title>
            </item>
            <item identifier="ITEM7" identifierref="RESOURCE7"
            isvisible="false">
              <title>Content 2</title>
            </item>
            <item identifier="ITEM8" identifierref="RESOURCE8"
            isvisible="false">
              <title>Summary 2</title>
            </item>
          </item>
          <item identifier="ITEM9" identifierref="RESOURCE9"
          isvisible="true">
              <title>Lesson 3</title>
            <item identifier="ITEM10" identifierref="RESOURCE10"
            isvisible="true" parameters="foo">
              <title>Introduction 3</title>
            </item>
            <item identifier="ITEM11" identifierref="RESOURCE11"
            isvisible="true">
              <title>Content 3</title>
            </item>
            <item identifier="ITEM12" identifierref="RESOURCE12"
            isvisible="true">
              <title>Summary 3</title>
            </item>
          </item>
        </organization>
      </organizations>
```

5.2.1.3 <resources>

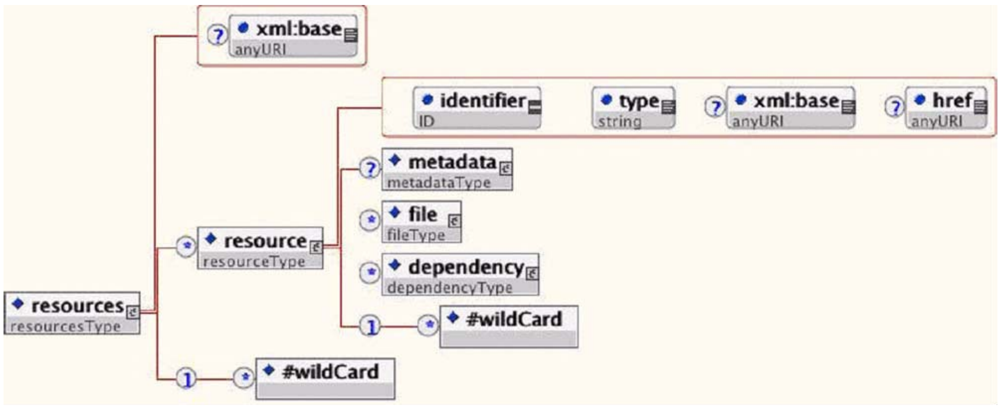
(1)說明(Description)

此元件表示內容檔案之集合。

This element identifies a collection of content files.

圖5.4 <resources>元件

Figure 5.4 - <resources> elements.



(2)重複性(Multiplicity)

在<manifest>元件中僅能出現1次。

Occurs once and only once within a <manifest> element.

(3)屬性(Attributes)

xml:base (選項)。提供內容檔案之相對路徑。此元件之使用定義於W3C的

XML Base 工作草案。資料型式 = 字串；

xml:base (optional). This provides a relative path offset for the content file(s). The usage of this element is defined in the XML Base Working Draft from the W3C. Data type = string.

(4)元件(Elements)

<resource>

(5)範例(Example)

```
<resources>
  <resource identifier="RESOURCE1" type="webcontent"
  href="lesson1.htm">
        <file href="lesson1.htm"/>
  </resource>
  <resource identifier="RESOURCE2" type="webcontent"
  href="intro1.htm">
        <file href="intro1.htm"/>
  </resource>
  <resource identifier="RESOURCE3" type="webcontent"
  href="content1.htm">
        <file href="content1.htm"/>
        </resource>
        <resource identifier="RESOURCE4" type="webcontent"
        href="summary1.htm">
              <file href="summary1.htm"/>
        </resource>
      </resources>
```

5.2.2 <metadata>元件(<metadata>Elements)

5.2.2.1 <schema>

(1)說明(Description)

描述使用的Schema(例如：IMS Content)，若無Schema元件，則預設爲“IMS Content”。資料型式 = 字串；

Describes the schema used (e.g., IMS Content). If no schema element is present, it is assumed to be “IMS Content”. Data type = string.

(2)重複性(Multiplicity)

在<metadata>中出現0或1次。

Occurs zero or once within <metadata>.

(3)範例(Example)

```
<schema>IMS Content</schema>
```

5.2.2.2 <schemaversion>

(1)說明(Description)

描述上述Schema之版本（例如：1、0、1.1版），若無版本說明，則預設爲“1.1”。資料型式 = 字串；

Describes version of the above schema (e.g., 1,0, 1.1). If no version is present, it is assumed to be “1.1”. Data type = string.

(2)重複性(Multiplicity)

在<metadata>中出現0或1次。

Occurs zero or once within <metadata>.

(3)範例(Example)

```
<schemaversion>1.1</schemaversion>
```

5.2.2.3 Meta-data

(1)說明(Description)

IMS Meta-Data標準對於詮釋資料用於描述及登錄內容包裝有更詳盡的說

明。IMS Meta-Data 1.2.1版爲預設之標準，但是亦可採用其他標準或標準。

See the IMS Meta-Data specification for more detail on the meta-data that are available for describing and cataloguing content packages. The IMS Meta-Data v1.2.1 is the default specification but other specifications/standards are permitted.

(2)重複性(Multiplicity)

定義相關詮釋資料標準。

Defined in the relevant meta-data specification.

(3)範例(Example): In-line meta-data

```
<metadata>
  <imsmd:lom>
    <imsmd:general>
      <imsmd:title>
        <imsmd:langstring xml:lang="en-US">Simple
      Manifest</imsmd:langstring>
      </imsmd:title>
    </imsmd:general>
  </imsmd:lom>
</metadata>
```

5.2.3 <organizations>元件 (<organizations>Elements)

5.2.3.1 <organization>

(1)說明(Description)

此元件描述素材之特定且中性的組織。

This element describes a particular, passive organization of the material.

(2)重複性(Multiplicity)

在<organizations>中出現0或多次。

Occurs zero or more times within <organizations>.

(3)屬性(Attributes)

(a) identifier (必備)。由編輯者或編輯工具提供的識別符，在內容清單中係唯一。資料型式 = id；

identifier (required). An identifier, provided by an author or authoring tool, that is unique within the Manifest.

Data type = id.

(b) structure (選項)。預設值爲“hierarchical”，樹狀或結構化之資料呈現。資料型式 = 字串；

structure (optional). Assumes a default value of “hierarchical”, such as is common with a tree view or structural representation of data.

Data type = string.

(4)元件(Elements)

(a) <title>

(b) <item>

(c) <metadata>

(5)範例(Example)

```
<organization identifier="TOC1">
  <title>default</title>
  <item identifier="ITEM1" identifierref="RESOURCE1" isvisible="true">
        <title>Lesson 1</title>
  </item>
  <item identifier="ITEM2" identifierref="RESOURCE2" isvisible="true">
        <title>Introduction 1</title>
  </item>
</organization>
```

5.2.3.2 <title>

(1)說明(Description)

此元件描述<item>之標題。

This element describes the title of an <item>.

(2)重複性(Multiplicity)

在<item>中出現0或多次。

Occurs zero or more times within <item>.

5.2.3.3 <item>

(1)說明(Description)

此元件描述結構中之節點。

This element describes a node within a structure.

(2)重複性(Multiplicity)

在<organization>中出現1或多次，在<item>中出現0或多次。

Occurs one or more times within <organization> and zero or more times within <item>.

(3)屬性(Attributes)

(a) identifier (必備)。內容清單之唯一的識別符。資料型式 = id。

identifier (required). An identifier that is unique within the Manifest. Data type = id.

(b) identifierref (選項)。參照<resource>之識別符(同一套裝之內)或用以解析檔案之完整位置之子內容清單。若未提供identifierref，則預設organization中沒有相關的內容。資料型式 = 字串。

identifierref (optional). A reference to a <resource> identifier (within the same package) or a sub-Manifest that is used to resolve the ultimate location of the file. If no identifierref is supplied, it is assumed that there is no content associated with this entry in the organization. Data type = string.

(c) isvisible (選項)。當課程實施時，標明資源是否顯示，若無，預設值爲“true”。資料型式 = boolean。

isvisible (optional). Indicates whether or not this resource is displayed when the unit of instruction is rendered. If not present, value is assumed to be ‘true’. Data type = boolean.

(d) parameters (選項)。啓動時傳遞至內容檔案之固定參數。資料型式 = 字串。

parameters (optional). Static parameters to be passed to the content file at launch time. Data type = string.

(4)元件(Elements)

(a) <title>

(b) <item>

(c) <metadata>

(5)範例(Example)

```
<item identifier="ITEM3" identifierref="RESOURCE3" isvisible="true">
        <title>Content 1</title>
</item>
```

5.2.3.4 <metadata>

(1)說明(Description)

此元件包含描述資源之詮釋資料。實作時可自由選用IMS Meta-Data標準或其他詮釋資料Schema定義之詮釋資料元件。

This element contains meta-data that describes the resource. Implementers are free to choose from any of the meta-data elements defined in the IMS Meta-Data specification or to define their own meta-data schema.

(2)重複性(Multiplicity)

在<organization>中出現0或1次。

Occurs zero or once within <organization>.

(3)範例(Example)

```
<organization identifier="TOC1">
  <metadata>
  <!-- schema and schemaversion not given as they apply at manifest level
  -->
    <imsmd:lom>
      <imsmd:educational>
        <imsmd:interactivitylevel>1</imsmd:interactivitylevel>
      </imsmd:educational>
    </imsmd:lom>
  </metadata>
</organization>
```

5.2.4 <resources>元件(<resources> Elements)

參照之資源的集合。此集合未預設順序或階層，資源可內嵌或從外部引用。

A collection of references to resources. There is no assumption of order or hierarchy. Resources can be described in-line or externally.

5.2.4.1 <resource>

(1)說明(Description)

此元件描述特定之內容檔案。

This element describes a specific content file.

(2)重複性(Multiplicity)

在<resources>中出現0或多次。

Occurs zero or more times within <resources>.

(3)屬性(Attributes)

(a) identifier (必備)。由編輯者或編輯工具提供的識別符，在內容清單中係唯一。

identifier (required). An identifier, provided by the author or authoring tool, that is unique within the Manifest.

(b) ‘type’ (必備)。表示資源型式之識別符。此標準定義“網頁內容(webcontent)”之型式，同時保留用於其他IMS標準定義之套裝內容的詞彙，包括學習設計(Learning Design)。此類標籤定義於‘Using IMS Content Packaging to

Package Instances of LIP and Other IMS Specifications’ [IMSBUND, 01] 實作手冊第七部份。IMS 標準第七部份有此表格之相關內容，藉由使用語法與規範性說明，陳述此標準影響之部份。

‘type’ (required). A string that identifies the type of resource. This specification defines the type “webcontent” plus reserved terms that are used to denote the packaging of content defined by other IMS specifications, including Learning Design. These labels are defined in Section 7 of the Implementation Handbook titled ‘Using IMS Content Packaging to Package Instances of LIP and Other IMS Specifications’ [IMSBUND, 01]. An IMS specification may extend the table in section 7 by using the syntax and including a normative statement to that effect in the specification.

(c) xml:base (選項)。提供內容檔案之相對路徑。此元件之使用定義於W3C之XML Base工作草案中。資料型式 = 字串。

xml:base (optional). This provides a relative path offset for the content file(s). The usage of this element is defined in the XML Base Working Draft from the W3C. Data type = string.

(d) href (選項)。此資源的“entry point”之參照。可容許完全符合條件的外部URIs。

href (optional). A reference to the “entry point” of this resource. External fully-qualified URIs are also permitted.

(4)元件(Elements)

(a) <metadata>

(b) <file>

(c) <dependency>

(5)範例(Example): 內部資源(In-line resource)

```
<resource identifier="RESOURCE2" type="webcontent"
href="topics/index.htm">
  <file href="topics/index.htm"/>
  <file href="images/pic1.gif"/>
  <file href="images/pic2.gif"/>
</resource>
```

5.2.4.1.1 <metadata>

(1)說明(Description)

此元件包含描述資源之詮釋資料。開發者可自由選用定義於IEEE

1484.12.1-2002學習物件詮釋資料標準(Standard for Learning Object Metadata)之詮釋資料元件(關於IEEE LOM標準之最佳實作與指導綱要請參照IMS Meta-Data v1.3 [MD, 04]），或自行定義詮釋資料Schema。

This element contains meta-data that describes the resource. Implementers are free to choose from any of the meta-data elements defined in the IEEE 1484.12.1-2002 Standard for Learning Object Metadata (see IMS Meta-Data v1.3 [MD, 04] for best practices and guidelines in implementing the IEEE LOM specification). or to define their own meta-data schema.

(2)重複性(Multiplicity)

在<resource>中出現0或1次。

Occurs zero or once within <resource>.

5.2.4.1.2 <file>

(1)說明(Description)

說明資源所屬之一或多個本地檔案，包含<resource>之href屬性所參照的資源。若資源參照至絕對URL(使用href），此<file>元件則非必備。

Identifies one or more local files that this resource is dependent on. This includes the resource being referenced in the href attribute of <resource>. If the resource references an absolute URL (using href), <file> element(s) are not required.

(2)重複性(Multiplicity)

在<resource>中出現0或多次。

Occurs zero or more times within <resource>.

(3)屬性(Attributes)

(a) href (必備)。檔案的URL。

href (required). URL of the file.

(4)元件(Element)

(a) <metadata>

(5)範例(Example)

```
<file href="topics/index.htm"/>
```

5.2.4.1.3 <dependency>

(1)說明(Description)

此元件用以識別單一資源，其可作爲多種檔案的資源承載容器。

This element identifies a single resource that can act as a container for multiple files that this resource depends upon.

(2)重複性(Multiplicity)

在<resource>中出現0或多次。

Occurs zero or more times within <resource>.

(3)屬性(Attributes)

(a) identifierref (必備)。供其他資源參照之識別符。

identifierref (required). An identifier for other resources to reference.

(4)範例(Example)

```
<resources>
  <resource identifier="R_A2" type="webcontent" href="sco1.html">
      <metadata/>
      <file href="sco1.html"/>
      <dependency indentiferref="R_A5"/>
  </resource>
  <resource identifier="R_A5" type="webcontent"
  href="pics/distress_sigs_add.jpg">
      <metadata/>
      <file href="pics/distress_sigs_add.jpg"/>
  </resource>
</resources>
```

5.2.5 延伸性(Extensibility)

IMS Content Packaging XML Binding 透過使用 XML 命名空間(XML Namespaces) XML 架構達到可延伸性。延伸機制應使用於描述其他詮釋資料之型式、組織及資源。IMS 內容包裝最佳實務指導綱要(IMS Content Packaging Best Practice Guide) [CP, 04c]有更詳細範例與說明。

The IMS Content Packaging XML Binding is extensible through the use of XML Namespaces and XML Schemas. It is expected that the extensibility mechanism will be used to describe additional types of meta-data, organizations, and resources. More information and examples of extensibility are contained in the IMS Content Packaging Best Practice Guide [CP, 04c].

5.3.範例(Samples)

一些伴隨著IMS Content Packaging標準的支援檔案文件以zip檔提供下載(imscp_v1p1p4.zip)，請參照附錄A。

A number of supporting files accompany the IMS Content Packaging specification documents and are available in the download .zip file (imscp_v1p1p4.zip), see Appendix A.

W3C ‘xml:’命名空間檔案可於網路上取得，但若處理器無法取得網路連線，使用者可先下載W3C同一份命名空間之‘xml.xsd’檔至本地電腦，再修改其綱要。在此種情況下，本地版本即爲控制文件，且需與內容清單檔案一同置於根目錄下。

The W3C ‘xml:’ namespace definition file is available online but in cases where the parser does not have internet access, users may modify the schema to reference a local version of ‘xml.xsd’ that has been retrieved from W3C and associated with the same namespace. In these cases the local version is a control document and as such it should be in the root directory along with the manifest file itself.

Content Packaging用以控管XSD之檔名僅標明第一或第二版，像是1.1.4版發布之XSD檔名‘ims_cpv1p1.xsd’。完整的版本號碼紀錄於Schema宣告中的‘version’屬性，如下面的例子：

The filename of the Content Packaging control XSD only identifies the primary and secondary versions of the version i.e., for the v1.1.4 release the XSD filename is ‘ims_cpv1p1.xsd’. The full version number is recorded in the ‘version’ attribute on the schema declaration e.g.,:

```
    <xsd:schema targetNamespace="http://www.imsglobal.org/xsd/imscp_v1p1"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://www.imsglobal.org/xsd/imscp_v1p1"
elementFormDefault="qualified" version="IMS CP 1.1.4">
```

6.內容包裝最佳實務及實作指南

6.1 概述

教學用內容經常需要去收集與包裝成某些電子格式，才可以進行聚合、散佈、管理、部署的動作。教材的製作者需要有效的工具與技術，來幫助他們創作內容；線上學習市場中的軟體業者

則需要生產能有效發佈與管理教學媒材的工具；最後，學習者有興趣的是能藉由良好的部署與接收工具，得到高品質的學習經驗。

6.IMS Content Packaging Best Practice and Implementation Guide

6.1 introduction

Instructional content often needs to be collected and packaged in some electronic form to enable efficient aggregation, distribution, management, and deployment. Producers of instructional materials want to have tools and technologies available to assist them in creating content. Software vendors in the online learning market want to create tools that enable efficient distribution and management of those instructional materials that have been created. Finally, learners are interested in high-quality learning experiences made possible by good deployment and delivery tools.

在線上學習社群中，內容物經過已知方式與檔案格式包裝、並附加足夠的資訊，比較能夠達成上述需求。對成長中的線上學習社群而言，需要制訂學習內容的方針與標準：

Content that is packaged in a known manner and file format, and with sufficient supporting information, can better satisfy the needs of the online learning community. This growing community needs guidelines and specifications for online learning content that will allow:

(1) 作者 建立線上學習內容。
(2) 管理者 管理和分發內容。
(3) 學習者 透過與內容互動，學習內容。

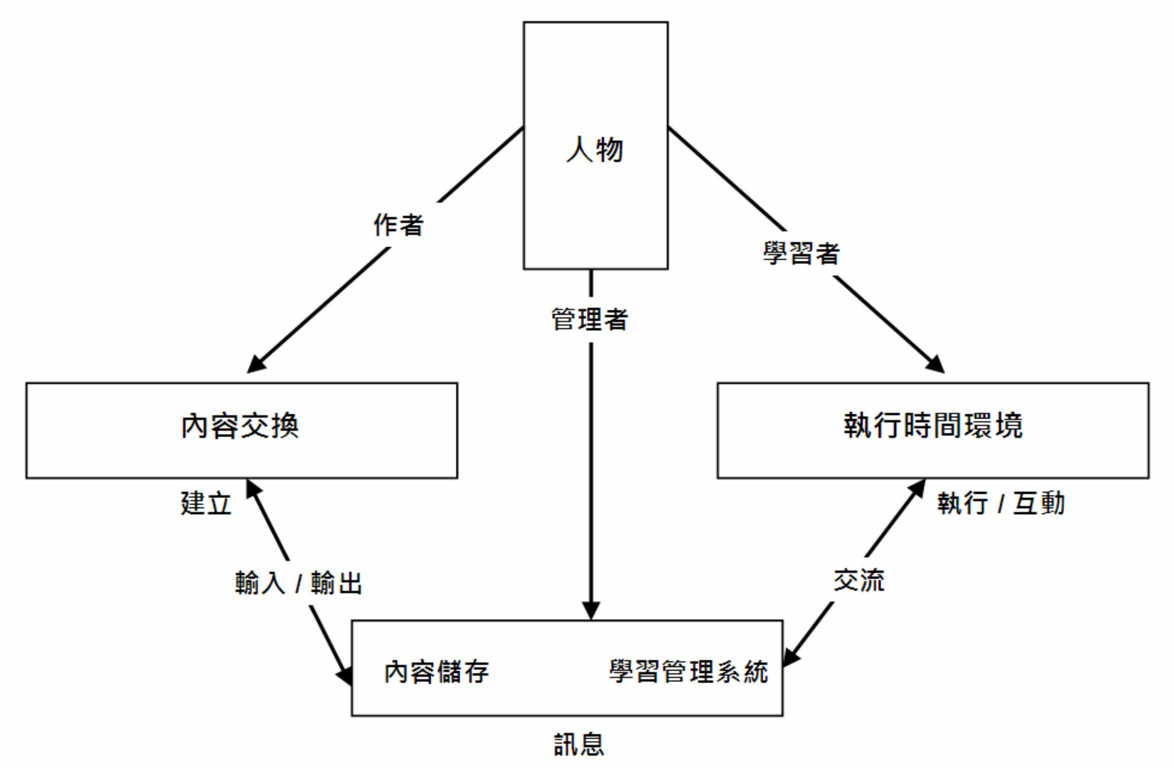
(1) Authors to build online learning content;
(2) Administrators to manage and distribute content;
(3) Learners to interact with and learn from the content.

達成此類精神目標的框架圖如圖6.1所示。IMS內容框架的目的就是可以簡單明瞭地將必要的內容資源、提供資訊形成封裝，以利封裝在網路上的溝通與線上學習使用。

A framework has been created with these goals in mind (Figure 6.1). The purpose of the IMS Content framework is to enable the encapsulation, in a concise and easily browsed manner, of all the required content resources, supporting information, and structure required to promote interoperable, online learning experiences.

圖6.1 IMS內容框架之目的

Figure 6.1 IMS Content framework goals



6.2 <MANIFEST>元件

套裝裡的檔案資源組織是獨立運作的。此 IMS 內容清單的<manifest>欄位，是用於以下目的：組織內容來表現一或多個表現結構或支援資源的詳細視圖。以此方法，<manifest>元件解說該套裝的集合與解開資源的內部結構。每個資源或一套資源支援一個顯示描述，包含組成內部檔案結構的每個內部資料夾或子目錄下的每個檔案的路徑。一內容清單可能提供一或多個靜態的內容呈現。

The organization of file resources within a Package is independent of their use. The <manifest> element in an IMS Manifest file serves the purpose of organizing the content for presentation in one or more presentation structures or views and of specifying the resource(s) supporting each view. In this way, a <manifest> element relieves the Package’s internal file structure from having to reflect the organization of resources for aggregation or disaggregation. Each resource or set of resources supporting a given presentation view is described for that view, including the path to each file through any internal folders or sub-directories comprising the internal file structure. A Manifest may provide one or more static views of the content.

單一<manifest>元件必須是 IMS 內容清單檔案的根節點元件。只能有唯一的<manifest>元件。在形成巢狀的<manifest>中其他的<manifest>皆要放在<resource>後面。此資訊模型沒有在<manifest>元件中加入特別的排序規則，而是遵守 XML 規則約束的簡單順序：依序為<metadata>、<organizations>、<resources>及<manifest>。內容清單包含了四個子元件：<metadata>、<organizations>、<resources>及更進一步的<manifest>元件。

A single <manifest> element is required as the root element of the IMS Manifest file. There can be one and only one top-level <manifest> element. All other instances of a <manifest> element are nested within the <manifest> element after the <resources> element. The information model does not impose a particular ordering within the <manifest> element however the corresponding XML binding does impose the implied order of: <metadata>, <organizations>, <resources> and <manifest>.A manifest contains four sub-elements: <metadata>, <organizations>, and <resources>, and any further <manifest> elements.

<metadata>（選項）：此標籤系描述詮釋資料。描述包含其內容清單。常用的詮釋資料元件包含標題、描述、關鍵字、貢獻者角色、內容用途（例如：教學目標、技術等級），和版權資訊。詮釋資料元件應優先選擇自 IEEE1484.12.1-2002 學習物件詮釋資料標準（請見 IMS Meta-Data v1.3 [MD, 04]，IEEE LOM 標準的最佳實務與實作指導綱要），之後再將所有不屬於 IEEE1484.12.1-2002 學習物件詮釋資料標準的詮釋資料元件以運用 XML 命名空間的機制包含在內容清單元件的詮釋資料元件中。全部的詮釋資料元件必須定義在 DTD 或 XSD 中，它們公布在此 IMS 內容清單檔案的頂部。直接參考的詮釋資料 DTD 或 XSD 檔案可以被包容在套裝之內部檔案結構根目錄的 imsmanifest.xml 裡。

<metadata> – (optional) this meta-data describes the manifest that contains it. Commonly used meta-data would include elements like title, description, keywords, a contributor’s role, a content’s purpose (e.g., educational objective, skill level), and copyright information. Meta-data elements should be drawn first from the IEEE 1484.12.1-2002 Standard for Learning Object Metadata (see IMS Meta-Data v1.3 [MD, 04] for best practices and guidelines in implementing the IEEE LOM specification). Any meta-data elements not found in the IEEE 1484.12.1-2002 Standard for Learning Object Metadata could then be included via an XML namespace in a manifest’s meta-data element(s).

All meta-data elements must be defined in a DTD or XSD, which are declared at the top of the IMS Manifest file. Directly referenced meta-data DTD or XSD files can be included with imsmanifest.xml at the root of a Package’s internal file structure.

<organizations>（必備）：此係爲描述組織之標籤。包括 0 個、1 或多個內容之靜態組織的描述，以便讓套裝裡的資源可以被移動來創作一或多個內容（例如：課程概要）的組織。課程的資源組織是否要描述係由內容製作者來判斷。如果內容製作者決定提出一個課程組織的單一或多個描述，他們也必須明確說明哪個是預設的。現行的內容包裝 XSD 需要單一的<organizations>元件做爲<manifest>元件的子項。如果內容製作者在此內容清單裡不需要組織段，它必須以空元件表示（例如：“<organizations/>”），以滿足控制文件（DTD、XSD）的控制法則。而且每個<manifest>元件中只允許有一個<organizations>。現行的標準定義<organizations>子元件作爲組織的分層體系；不過，以其他方式來描述組織結構或內容（例如：有條件/計畫性的）也是被允許的。

<organizations> – (required) contains zero, one, or multiple descriptions of the static organization of the content so that resources within the Package can be moved to create one or multiple organizations of content (such as course outlines). It is left to the discretion of content producers to decide whether to describe or not describe the organization of a course’s resources. If content producers choose to provide one or more descriptions of a course’s organization, they must also specify one as the default. The current Content Packaging XSD requires a single <organizations> element as a child of the <manifest> element. If content producers do not need an organizations section in the manifest, then it must appear as an empty element (i.e., “<organizations/>”) to satisfy the control rules expressed in the controlling documents (DTD, XSD). Also, only one <organizations> element is allowed within each <manifest> element. The current specification defines an <organization> sub-element as one that uses a hierarchical organization; however, other ways of describing the organizational structure or content (such as conditional/programmatic) are permitted.

<resources>（必備）：此係爲描述資源之標籤。包括套裝所包含之全部資源的參考。它至少要指引到檢視<organizations>元件中所指述之內容時所需要的全部檔案。透過相對或絕對的識別符來指引在套裝裡面的、或者是在外部的參考。舉例來說，引用外部 URL 的參考就可以不用將該資源包含在此套裝交換檔案中。資源對所引用的每個內容的元件亦可能包含<metadata>元件。而在最上層<manifest>元件中，只允許有唯一的<resources>元件。

<resources> – (required) includes references to all of the resources included in the package. At a minimum it should reference all those files that are needed in order to view the content as specified in the <organizations> element. References may either be made internally or externally of a Package to both relative and absolute identifiers. For example, a reference to an external URL is permitted without having to include that resource as part of the Package Interchange File. Resources may also contain a <metadata> element for each content item referenced. Only one <resources> element is allowed within the top-level <manifest> element.

<manifest>（選項）：此標籤係爲描述內容清單之標籤。指定 0 個以上的子內容清單。巢狀<manifest>元件可以詳述內容如何確實地聚合與解聚成其他套裝。
後面的章節會對此類做更詳盡的描述。

<manifest> – (optional) specifies zero or more sub-Manifests. Nested <manifest> elements specify how content may be reliably aggregated or disaggregated into other Packages.
The following sections describe these more fully.

6.3 <METADATA>元件

Metadata（詮釋資料）係爲選項，且允許用在內容清單中的多個地方，以更完整描述套裝之內容。搜尋引擎也可以查看 metadata 來爲學習者找出合適的內容。版權和其他智慧財產權也可以簡易地公告於 metadata 中。編輯軟體可讀取內容業者所規定的權利，以確認其是否有權限打開資源檔案並改變其內容。

Meta-data is optional and is allowed in various places in the manifest to more fully describe the contents of a Package. Search engines may look into the meta-data to find appropriate content for a learner or for content repackaging. Copyright and other intellectual property rights are easily declared within the meta-data. Authoring or editing tools could then read the rights stipulated by a content vendor to see if they have permission to open a resource file or files and change the contents.

IMS 內容套裝資訊模型定義 metadata 的五個部分，用以描述內容套裝不同的部分：
(1) Manifest。
(2) Organization。
(3) Item。
(4) Resource。
(5) File。
若需要描述 metadata 之元件，那麼每個元件應分別描述於 metadata 中分隔開的段落之中。此結構允許套裝的每個組成部分皆可以有詳細的說明。

The IMS CP Information model defines five places where meta-data can be used to describe different components of a content package:
(1) Manifest
(2) Organization
(3) Item
(4) Resource
(5) File

If there are requirements to describe any or all of these components with meta-data, then each of these respective components shall be described with separate instances of Meta-data. This construct allows a fine-grained description of each component of a package.

不過要注意，從一邏輯節點到另一個之間並沒有繼承關係。每個構成元件如果有需要，可以有自己的 metadata 段來描述它自己。舉例說明，如果此 metadata 和 resource X 相連，其作者是 Jane Smith，另有 file Y 沒有跟隨在此 metadata 後面，它是 resource X 的子節點而沒有 metadata，它也是 Jane Smith 創作的。在該例中，如果 Jane 要註記爲 file Y 的作者，就需要一段個別的 metadata 段和 file Y 連繫。

Beware, however, that there is no assumption of inheritance from one logical node to another. Each component, if desired, is represented by its own meta-data instance. If the meta-data associated with a resource X, for example, identifies Jane Smith as the author, it does not follow that file Y, a child node of resource X without meta-data, is also authored by Jane Smith. In this case, if Jane needs to be identified as the author of file Y, a separate meta-data instance needs to be associated with file Y.

描述、編目內容包裝可使用的完整全套的 metadata 元件並沒有包含在這份標準中。本標準建議採用 IEEE1484.12.1-2002 學習物件詮釋資料標準（參照 IMS Meta-Data v1.3 [MD, 04]，IEEE LOM 標準的最佳實務及實作指導綱要），它包括了 86 個 metadata 元件，給套裝的作者使用以描述和編目內容包裝。

The complete set of meta-data elements available for describing and cataloging a content Package is not included withthis specification. This specification recommends the best practice of using the IEEE 1484.12.1-2002 Standard for Learning Object Metadata (see IMS Meta-Data v1.3 [MD, 04] for best practices and guidelines in implementing the IEEE LOM specification), which contains approximately 86 individual meta-data elements that may be used to describe and catalog Content Packages, as the Package author sees fit.

6.3.1 外部詮釋資料(External Meta-Data)

實行內容包裝可能（但不必須）包括延伸引用外部的詮釋資料檔案。本版本的內容包裝標準並不指定或推薦這樣的機制，但內容包裝標準將來的版本將可以處理此問題。下面就是以外部 metadata 檔案延伸引用的例子：

```
<metadata>
   <schema>ADL SCORM</schema>
   <schemaversion>CAM 1.3</schemaversion>
   <adlcp:location>Lesson01.xml</adlcp:location>
</metadata>
```

本例取自於 ADL SCORM 2004 簡介。

A Content Packaging implementation may, but does not need to, include an extension that references an external meta-data file. This version of the Content Packaging specification does not specify or recommend such a mechanism, but future versions of the Content Packaging specification may address the issue. An example of an in-line extension that refers to an external meta-data file is demonstrated by the following fragment:

```
<metadata>
   <schema>ADL SCORM</schema>
   <schemaversion>CAM 1.3</schemaversion>
   <adlcp:location>Lesson01.xml</adlcp:location>
</metadata>
```

This example is taken from the ADL SCORM 2004 profile.

6.4 <ORGANIZATIONS>元件

目前有許多組織課程或是套裝內容的方法，包括沒有任何組織的情形。在內容清單檔案裡，<organizations>元件包含此類資訊。

There are many ways to organize course or Package content, including no organization at all. In a manifest file, the <organizations> element contains this information.

規畫組織讓其能有分階層的分枝、索引、利用條件分枝自定學習路徑與複合式物件分類皆是可以辦到的。如果課程或套裝並不需要特別的組織，<organizations>元件仍然是必要的，而且必須以<organizations/>呈現，以滿足 XSD 中所定義的控制規則。前述例子<organizations>元件便是空值。

It is possible to imagine organizations that will take into account such approaches as hierarchical branching, indexes, custom learning paths utilizing conditional branching, and complex objective hierarchies. If the course or Package presentation does not require a specific organization, the <organizations> element is still necessary and must appear as follows to satisfy the control rules expressed in the XSD: <organizations/>. However, in this case the <organizations> element is left empty.

儘管有許多內容組織的方法已被發展，在本標準中仍包括一種預設的方法，作爲本標準的一部分。此內容組織的預設方法以類似樹狀視圖或分層視圖的方式包含在<organization>元件中。此<organization>元件是在<organizations>元件中唯一允許使用的元件。內容可以有補充的組織架構，透過使用類型屬性將其設成非預設值。可以有多種組織架構，甚至有一個以上的組織架構是屬於同一型式，但只能規定一個爲預設。

While many content organization approaches may be developed, a default approach is included as part of this specification. This default approach to content organization, similar to a tree view or hierarchical representation, is encompassed in the <organization> element. The <organization> element is the only element allowed under <organizations>. Content may have additional organization schemas, through the use of the type attribute by setting it to a non-default value. There can be multiple organizations and more than one of the same type, but only one specified as the default.

6.4.1 <ORGANIZATION>元件

<organization>元件含有關於資料詳細組織的資訊。<organization>元件採取一種預設的結構標誌來表現分層的數値，譬如通常使用資料的樹狀圖或結構圖。本標準的未來版本可能會爲結構標誌增加一些附加數値來對應一些附加的結構組織或模型，例如：指向圖、語意網絡圖等。但除非使用附加數値改寫，否則<organization>元件預設皆爲<organization structure=hierarchical>。

The <organization> element contains information about one particular, passive organization of the material. The <organization> element assumes a default structure attribute value of hierarchical, such as is common with a tree view or structural representation of data. Future versions of the specification will likely include additional values for the structure attribute to correspond with additional structural organizations or shapes, such as a directed graph, a semantic network, or others. Until additional values are agreed upon, the <organization> element, by default, effectively reads: <organization structure=hierarchical>.

如果<organizations>元件中有一個以上<organization>元件，那麼它們應該是實質上會有相同學習成果的相異組織。而實質上目標不相同的資料就應該在分開的套裝裡出現。在<manifest>元件層級裡的詮釋資料中還是需要描述此套裝的整體目的。

If there is more than one <organization> element within the same <organizations> element, then it is expected that they should be variant organizations with substantially the same learning outcomes. Material with substantially different objectives should appear in separate Packages. It should always be the case that the meta-data at the <manifest> element level describe the purpose of the Package as a whole.

當一<organizations>元件中包含了多個<organization>元件時，建議依下述程序，選用<organization>的理由：

(1) 如果有給<organizations>預設屬性數値，就判定使用此組織。這是特定<organization>的選用判定順序；

(2) 若無預設値，則採用第一個遇到的<organization>元件。

Where an <organizations> element contains multiple <organization> elements, the following procedure is recommended, if one <organization> is to be selected for any reason:

(1) If there is a value given for the default attribute of <organizations>, then this identifies the organization to be used. This is the preferred method for identifying a particular <organization>;

(2) If there is no default given, then the first <organization> element encountered should be used.

製作內容包裝的軟體可以使用上述程序或是可以：

(1) 使用組織層級的詮釋資料來選擇，使用屬於它自己的規則；

(2) 容許使用者選擇組織；

(3) 使用任何其他的適當方法。

Software that processes a Content Package may use the above procedure or it may:

(1) Use organization-level meta-data to make a selection, using its own rules;

(2) Allow users to select the organization;

(3) Use any other suitable approach.

<organization>的顯示結構是透過<item>子元件來描述的。<item>可能包含子<item>元件（以分層級的方式呈現）或是可以顯示同層級的其他<item>（以同層級的方式）。巢狀層級<item>元件可以定義出樹狀瀏覽或分層結構圖。標題是可選擇的，但是推薦使用。<item>元件也可設定顯示或隱藏，預設爲顯示。

The presentation structure of <organization> is described through <item> sub-elements. An <item> may contain subordinate <item> elements (a hierarchical approach to presentation) or may appear on the same level as other <item>s (a flat approach). A tree view, or hierarchical representation, may be defined by the nesting levels of the <item> elements. Content developers can mix and match nesting levels as appropriate for their content. An <item> always has an identifier, and is linked to resources through an identifierref attribute. Titles are optional, but encouraged. The <item> element may also be visible or hidden, the default presence is visible.

作者可能也包含在<organization>與<item>元件的詮釋資料之中，它們描述的附加資訊以運用在儲存時的索引搜尋。

Authors may also include meta-data within the <organization> and <item> elements allowing them to describe additional information meaningful for searching or for indexing in a repository.

範例：內容清單的層級組織結構，按照<organization>元件與其包含的巢狀<item>元件順序，如下：

Example: A hierarchical organizational scheme for a manifest can be determined by the order and nesting of the <item> elements contained within the <organization> element, similar to the following:

```
<organization identifier="TOC1">
  <title>Default Organization</title>
  <item identifier="ITEM1" identifierref="RESOURCE1">
          <title>Lesson 1</title>
  </item>
  <item identifier="ITEM2" identifierref="RESOURCE2">
          <title>Lesson 2</title>
  </item>
  <item identifier="ITEM3" identifierref="RESOURCE3">
          <title>Lesson 3</title>
  </item>
</organization>
```

當學習管理系統或內容的觀眾遇到此內容的結構化組織或是階層樹狀圖時可以理解爲：

• Lesson 1

  • Lesson 2

• Lesson 3

An LMS or content viewer encountering this structural organization or hierarchical tree view of the content could interpret it conceptually as:

- Lesson 1
  - Lesson 2
- Lesson 3

6.4.2 使用巢狀< MANIFEST >元件(Using Nested < MANIFEST > Elements)

參考<item>元件的資源的機制是名叫“ identifierref”的屬性，它被用以引用資源。爲了確保內容清單保有將來仍然可以解開的能力，需要在引用上設定幾個強制的限制，包括了：

The mechanism for referencing an <item> element’s resource is the ‘identifierref’ attribute, which is used to reference resources. Certain restrictions are placed on the kinds of references that can be made in order to maintain the capability for future disaggregation of a compound Manifest, including:

(1) <item>元件的 identifierref 能引用的資源，可以在包容它的子<manifest>元件中找到。它也可以引用任何巢狀<manifest>的資源；

(2) 反過來說，不行：<item> 元件的 identifierref 不能引用比包容其<manifest>元件更高層的<manifest>，或是引用任何來自較高層<manifest>元件的資源。如果這麼作，當包含的內容清單解聚和建立不同的套裝時就找不到參考了。如果內容製作者需要引用分開的、外部的套裝，他們必須先把它聚合起來然後把它指定出來。

(3) <item>的 identifierref 可以引用子內容清單。

(1) An <item> element’s identifierref can reference resources found in a subordinate <manifest> element in which it is contained. It can also reference the resources of any nested <manifest>;

(2) The reverse is not true: An <item> element’s identifierref cannot refer to a <manifest> element that is higher than the <manifest> element that contains it, or to any resource referred to by a higher-level <manifest> element. If it were to do so, such references could not be resolved should the contained Manifest be disaggregated and used to create a

different Package. If content producers need to reference a separate, external Package, they must first aggregate it and then point down to it;

(3) An <item> element’s identifierref can reference a sub-Manifest.

6.5 <RESOURCES>元件

<resources>元件用以註明內容的集合與其檔案。個別的資源則宣告於以<resources>爲巢的個別<resource>元件裡。<resource>不一定是單一檔案。它可能是支援相關呈現結構（<item>元件）之檔案集合。此類檔案可能爲內部參考或是透過 URL 從外部參考。內部參考的檔案必須包含於套裝交換檔案中。

The <resources> element identifies a collection of content and its files. Individual resources are declared as a <resource> element nested within the <resources> element. A <resource> is not necessarily a single file. It may be a collection of files that support the presentation of the associated presentation structure (<item> element). These files may be internally referenced or externally referenced via a URL. Internally referenced files must be included in the Package Interchange File.

<resource>元件也可以有<metadata>子元件。從屬於<resource>下的<metadata>元件，無論它是單獨的檔案或是一群檔案集合。

A <resource> element may also have a <metadata> sub-element. The <metadata> element is for the <resource>, whether it is a single file or a collection of files.

<file>元件可以包含<metadata>子元件，它允許作者去描述附加的<file>資訊以應用在儲存時的搜尋索引。<resource>可以用相對路徑 URL 如 href 來參考內部（或本地）的檔案，或是以完全合法的遠端 URL 來參考外部的檔案或服務。被資源使用的內部檔案，可直接用<file>元件列舉，或是使用<dependency>元件間接參考另外的資源。舉例來說，在套裝中列舉出來全部的檔案集合（除了繫結控制文件和 imsmanifest.xml 檔案）在內容包裝的傳遞中皆必須傳送。而外部參考就不是套裝的部分，也不會出現在<file>元件裡。

A <file> element may contain a <metadata> sub-element allowing authors to describe additional <file> information meaningful for searching or for indexing in a repository. A <resource> may reference an internal (or local) file by a relative URL as the href or as an external file or service by a fully-qualified remote URL. Internal files used by the resource are either directly enumerated by <file> elements or indirectly enumerated by using the <dependency> element to reference another resource. For example, the union of all file enumerations in a Package identifies all files (excluding binding control documents and imsmanifest.xml files) that must be communicated on transmission of a
content Package. External referents do not form part of the Package and do not appear in <file> elements.

<resource>元件也可以包含<dependency>子元件。<dependency>元件註明能作爲多個檔案的容器的單個資源。並不需要在編列資源項目時逐項填寫，<dependency>容許作者定義資源的容器，簡單地指定<dependency>元件來代替個別單獨的資源。對<dependency>與<item>使用的 identifierref 屬性值有同樣的限制要求（進一步的指引請參照第 4.4.2 節），但在子內容清單中指引資源這點例外；<item>可以這麼做，但<dependency>不行。下面便是使用<dependency>的例子：

A <resource> element may also contain a <dependency> sub-element. The <dependency> element identifies a single resource which can act as a container for multiple files that this resource depends upon. Rather than having to list all resources item by item each time they are needed, <dependency> allows authors to define a container of resources and to simply refer to that <dependency> element instead of individual resources. The same restrictions on the values of the identifierref attribute apply to <dependency> as apply to <item> (see Section 4.4.2 for further guidance), with the exception of referring to resources in sub-Manifests. An <item> can do this, a <dependency> can’t. Below is an example of using <dependency>.

```
<resources>
   <resource identifier="R_A1" type="webcontent" href="sco06.html">
      <metadata/>
      <file href="sco06.html" />
      <file href="scripts/APIWrapper.js" />
      <file href="scripts/Functions.js" />
      <dependency identifierref="R_A4" />
      <dependency identifierref="R_A5" />
      <dependency identifierref="R_A6" />
   </resource>
   <resource identifier="R_A2" type="webcontent" href="sco1.html">
      <metadata/>
      <file href="sco1.html" />
      <file href="scripts/APIWrapper.js" />
      <file href="scripts/Functions.js" />
      <dependency identifierref="R_A5" />
   </resource>
   <resource identifier="R_A4" type="webcontent" href="pics/distress_sigs.jpg">
      <metadata/>
      <file href="pics/distress_sigs.jpg" />
   </resource>
   <resource identifier="R_A5" type="webcontent" href="pics/distress_sigs_add.jpg">
      <metadata/>
      <file href="pics/distress_sigs_add.jpg" />
   </resource>
   <resource identifier="R_A6" type="webcontent" href="pics/nav_aids.jpg">
      <metadata/>
      <file href="pics/nav_aids.jpg" />
   </resource>
</resources>
```

當描述的素材經由網路瀏覽器來展現時，“type”屬性通常被設爲“webcontent”。然而當內容包裝用以容納像 QTI-XML 的資料時，此“type”屬性應該要照實行手冊第 7 章-標題“使用 IMS 內容包裝 LIP 與其他 IMS 標準包裝的實例” [IMSBUND, 01]的建議作法。當用於容納學習用途資訊時，“type”屬性設爲“imsldcontent”。若情形是不符合任何前面提到的模式，那麼此屬性應該調整爲“other”。

The ‘type’ attribute is usually set to ‘webcontent’ when describing material that is to be launched through a Web browser. However, when a Content Package is being used to contain data such as a QTI-XML based assessment then the value of the ‘type’ attribute should be set as recommended in

Section 7 of the Implementation Handbook titled ‘Using IMS Content Packaging to Package Instances of LIP and Other IMS Specifications’ [IMSBUND, 01]. The ‘type’ attribute is set to ‘imsldcontent’ when used to contain learning design information. In situations where none of the previous methods are appropriate, then this attribute should be set to ‘other’.

實行手冊裡除了建議詞彙的用法之外並沒有擴充詞彙表，此標準未來的版本將會增加詞彙的擴充。

Apart from the usage of the vocabulary in the Implementation Handbook it is recommended that this vocabulary is NOT extended. Later versions of this specification will address the extension of this vocabulary.

6.6 <RESOURCES>與巢狀<MANIFEST>元件的範例(Examples of <RESOURCES> and Nested <MANIFEST>Elements)

IMS網站可下載到說明描述內嵌之子內容清單方式的範例。在 http://www.imsglobal.org/content/packaging/可找到此內容清單及其他範例。至於本標準未來的版本可能如何描述外部子內容清單的範例，請參照附錄D。

There is an example available for download from the IMS website that illustrates how to describe in-line sub-Manifests. You can find this sample and others at http://www.imsglobal.org/content/packaging/. For an example of how external sub-Manifests may be described in a future version of the specification, see Appendix D.

6.7 建立IMS套裝或套裝交換檔案(Building an IMS Package or Package Interchange File)

(1) 套裝中必備的命名空間皆應該在最上層<manifest>元件的屬性中宣告。

(2) 內部參考的 imsmanifest.xml 檔案和支持命名空間的檔案(DTD、XSD) 必須放置在套裝或套裝交換壓縮檔的根節點目錄。

(3) 全部的內部參考檔案必須存放在套裝的<resource>元件宣告的路徑位置。

(1) Any namespaces required within a Package should be declared as attributes of the top-level <manifest> element;

(2) The imsmanifest.xml file and any files supporting namespaces (DTD, XSD) that are referenced internally must be placed at the root of the Package or compressed Package Interchange File;

(3) All internally referenced files must be stored in the paths declared in all <resource> elements in a Package.

6.8 套裝的聚合與解聚(Aggregation and Disaggregation of Packages)

若要將某個單純（非聚合）的套裝聚合成新的（大範圍的）套裝，首先必須取用其內容清單以獲得與其<resource>元件列表。遍歷這些<resource>元件列表，並檢驗每個<file>元件來確認它們參考的是外部的還是內部的檔案(請注意<resources>元件的任何基點位置屬性與每個<resource>元件的置換基點位置屬性，皆要加在<file>元件的檔案參考）。這是用以建立被聚合的本地的套裝中全部檔案的列表清單。此清單輪流被檢視以取用以取用每個檔案並在新的套裝中建立其複本。接下來，被聚合之套裝的內容清單必須整合旗下的子<manifest>元件，包進正被建立之套裝的內容清單中。當此新套裝完成建立時，所包含的內容清單會放在新套裝交換檔案的根目錄，儲存於 imsmanifest.xml 檔案中。

If a simple (non-aggregated) Package is to be aggregated into a new (super-)Package, first its manifest must be accessed and its list of <resource> elements obtained. These are traversed and each of their <file> elements examined to determine whether they reference external or internal files (note that any base address attribute in the <resources> elements and any overriding base address attributes in each <resource> element, need to be prefixed to a <file> element’s file references). This is used to build a list of all the files contained locally in the Package that is being aggregated. This list in turn, is then used to access each file and to create a copy of it in the new Package. Next, the manifest of the Package being aggregated must be integrated as a subordinate <manifest> element into the manifest that is being created for the containing Package. When the construction of the new Package is complete, the containing manifest is saved as a file with the name imsmanifest.xml at the root of the new Package Interchange File.

若要將套裝從原先的大套裝中解開成較小的子套裝，那麼首先子套裝的<manifest>元件必須能自從包容的 imsmanifest.xml 檔中取用。然後藉由讀取被取用之內容清單的<resources>段來判定原本包含在這段的實體檔案。然後這份清單用來找出這些檔案在較大的套裝裡的位置，再把它們複製到新的較小套裝去。被取用的內容清單接著會存成命名爲 imsmanifest.xml 的檔案，一樣放在新套裝交換檔案的根目錄。

If a Package is to be disaggregated from a containing Package into a smaller, sub-Package, first that sub-Package’s <manifest> element must be accessed in the containing imsmanifest.xml file. The <resources> section of the accessed manifest is then read to determine the physical files that were originally contained in that section. This list is then used to locate these files in the larger Package and these are then copied to the new, smaller Package. The accessed manifest is then saved as a file with the name imsmanifest.xml and also included at the root of the new Package Interchange File.

若一集合套裝，含有聚合的子套裝，它自己要再被集合，請依循一樣的步驟：爲了建立全部的子內容清單參考的完整檔案列表，也需要執行集合套裝的子內容清單元件的附加。當這合併之套裝的內容清單已經包括全部的巢狀子內容清單時，此內容清單只需要被合併到新的集合內容

清單。同樣地，如果要解聚集合的子套裝，其子內容清單樹狀需要被執行以建立需要複製到解聚套裝的完整檔案列表。

If a compound Package, containing aggregated sub-Packages, is itself to be aggregated, then the same procedure is followed; with the addition that the compound Package’s sub-Manifest elements also have to be walked in order to build a complete list of files referenced in all the sub-Manifests. As the aggregated Package’s manifest already contains all the nested sub-Manifests, only this manifest needs to be merged into the new containing manifest. Similarly, if a compound sub-Package is to be disaggregated, its sub-Manifest tree needs to be walked in order to build the complete list of files that need to be copied into the disaggregated Package.

套裝，特別是組織的項目，不能引用此套裝範圍之外的套裝元件（<resource>元件）。被引用的元件必須被包含在引用它們出來的同一個套裝裡，包含該套裝之子套裝的元件。本標準並不包括關於引用的元件應該怎麼使用聚合與解開的工具來維護的說明。關於智慧財產權，以及資源如何保持它們原創、獨一無二的識別符等議題也不在本內容包裝標準之範圍。

Packages, specifically organizational items, may not reference Package elements (<resource> elements) that are outside the Package scope. Referenced elements must be contained in the same Package from which they were referenced, including elements that are in sub-Packages within the Package. This specification does not contain rules as to how such referenced elements should be maintained by aggregation and disaggregation tools. The issue of intellectual property rights, concerning how resources preserve their original, unique identifiers is beyond the scope of this version of the Content Packaging specification.

6.8.1 識別符(Identifiers)

當建立或處理套裝時，需要考慮識別符的範圍。爲了有效的內容包裝的內容清單，識別符必須是唯一的。如果一套裝被聚合成另一個套裝，藉由使用通用唯一識別符於內容清單，可以避免識別符衝突（對應 IMS Persistent、Location-Independent Resource Identifier Handbook [IMSPLID, 01]，如何產生或獲得識別符）。如果系統的儲存規畫沒有使用通用唯一識別符，就不應該在沒有支援套裝集合建立唯一識別符的工具的情況下，和其他系統交換套裝。

When creating or manipulating Packages, the scope of identifiers needs to be considered. In order to be a valid Content Packaging Manifest, identifiers must be unique. If a Package is aggregated into another Package, identifier collisions could be avoided or resolved by using universally unique identifiers across manifests (such as identifiers generated or obtained according to the IMS Persistent, Location-Independent Resource Identifier Handbook [IMSPLID, 01]). If universally unique identifiers are not used in a system’s own storage scheme, Packages should not be exchanged with other systems without building unique identifier generation into tools that support Package aggregation.

XML 繫結使用 XML 結構“xsd:ID”和“xsd:IDREF”來驗證識別符的唯一性。這確保識別符在該 XML 文件中係唯一，任何識別符使用“IDREF”必須在 XML 文件中有對應的“ID”宣告。“ID”和“IDREF”的用法並不保證是全球唯一識別符，所以當用於套裝集合時需要注意。將“ID”宣告傳送到新的聚合套裝也很重要，否則將會發生語法驗證的錯誤。

The XML binding uses the XML structures ‘xsd:ID’ and ‘xsd:IDREF’ to validate the uniqueness of the identifiers. This ensures that the identifiers are unique within the XML document and that any identifier referenced using ‘IDREF’ must have a corresponding ‘ID’ declaration within the XML document. The usage of ‘ID’ and IDREF’ do not ensure globally unique identifiers and so care needs to be taken when using package aggregation. It is also important that the ‘ID’ declaration for the identifier is passed into a newly aggregated package otherwise a parser validation error will occur.

6.8.2 XInclude

IMS 內容工作群建議 XInclude 機制，透過 W3C 完全贊成與支援時，可以作爲集合與解開套裝資源的有效方法。雖然如此，在 W3C 定案將 XInclude 當作建議與 XML 團體普遍支持它之前，作者還是不應該把 XInclude 用在包裝內容上。

The IMS Content Working Group expects that the XInclude mechanism, when fully approved and supported by the W3C, may prove a powerful way to support the aggregation and disaggregation of Package resources. However, authors should not use XInclude in packaging content until the W3C finalizes XInclude as a Recommendation and the XML community generally supports it.

備考：此處提到的 XInclude，是新出現的標準，IMS 有可能影響內容包裝標準的未來版本而非發展包含外部 XML 檔案的其他方法。參照附錄 D，舉例說明 XInclude 可能如何使用於本標準的未來版本。

Note: XInclude is mentioned here as an emerging standard that IMS will likely leverage in future versions of the Content Packaging specification rather than invent another way of including external XML files. See Appendix D for examples of how XInclude might be used in future versions of this specification.

6.8.3 xml：base 屬性

xml:base 是一種用以明確地記載解析外部檔案鏈結相關 URI 文件中的基礎 URI。在‘imsmanifest.xml’檔案中，內部與外部參考可能是絕對的或相對的。相對位址可加上‘xml:base’屬性。‘xml:base’屬性允許外部和本地的基礎位置皆可被指定。至於那些缺乏‘xml:base’的相對 URL，則是指相對於套裝根節點（‘imsmanifest.xml’的位置）而言。而有‘xml:base’出現的路徑，相對 URL 是相對於‘xml:base’指定的路徑而言。相對於‘xml:base’本身之絕對路徑，需

由套裝文件的位置來判定。意即當它被讀取時，輸入系統中‘imsmanifest.xml’檔案的位置，提供缺少的絕對路徑，此類規則表述於 RFC2396。

‘xml:base’ is a construct used to explicitly specify the base URI of a document in resolving relative URIs in links to external files. In the ‘imsmanifest.xml’ file, internal and external references may be absolute or relative. Relative addresses can be prefixed by an ‘xml:base’ attribute. The ‘xml:base’ attribute allows both external and local base addresses to be specified. Relative URLs, in the absence of ‘xml:base’, are relative to the Package root (location of ‘imsmanifest.xml’). In the presence of an ‘xml:base’ path, relative URLs are relative to the path specified in ‘xml:base’. When an ‘xml:base’ path is relative itself, the absolute path is then resolved to the location of the containing document. That is, the location of the ‘imsmanifest.xml’ file in an importing system, when it is read, supplies the missing absolute segment, per the rules expressed in RFC 2396.

在子內容清單中宣告的相對‘xml:base’路徑是相對於套裝根節點而言。假如附帶被宣告‘xml:base’路徑的內容清單包含子內容清單，此子內容清單也宣告了‘xml:base’路徑，在運行時多個‘xml:base’路徑就不應該被連結，相反地，該附屬清單中之 URI 只能相對於子內容清單宣告的 xml:base。當然在聚合時開發者可自由的建立相對‘xml:base’路徑之連結或採用其他方法。

Relative ‘xml:base’ paths that are declared in a sub-manifest are relative to the Package root. In cases where a manifest with a declared ‘xml:base’ path contains a sub-manifest, and the sub-manifest also declares an ‘xml:base’ path, the multiple ‘xml:base’ paths should not be concatenated at runtime. Instead, the URIs within such a sub-manifest are relative to the declared xml:base of the sub-manifest only. Implementors are, of course, free to construct a relative sub-manifest ‘xml:base’ path by concatenation or any other means at aggregation time.

在參考外部位置的‘xml:base’路徑存在的情況，相對 URL 是指相對於此位置而言。絕對（外部）URL 被認定爲被完全指定而不由附加路徑提供。

In the presence of an xml:base path, which references an external location, the relative URLs are relative to that location. Absolute (external) URLs are considered to be fully-specified without the provision of additional pathing.

使用‘xml:base’屬性時，必須注意不要超過任何相關的‘href’的長度限制。‘href’加‘xml:base’的最大長度定義爲 2000 個位元組。假如多個‘xml:base’值需要連續建立完整路徑，那麼須確保總長度不要超過‘href’，路徑長度若大過 2000 個位元組，則系統的行爲沒有定義。

When the ‘xml:base’ attribute is used, care must be taken not to exceed the length of any associated ‘href’. The maximum length of both ‘href’ and ‘xml:base’ is defined as 2000 octets. In cases where multiple ‘xml:base’ values need to be concatenated to create the full path then care must be taken to ensure that the total length does not exceed that of the ‘href’. If the path length is greater than 2000 octets then the system behavior is undefined.

當包裝時使用 xml:base，xml:base 路徑不應該以斜線 (leading forward slash)開始。其定義在 RFC 2396，一個路徑用斜線表示一個資源的絕對路徑。使用斜線很容易被曲解成宣告文件在本機位置。因爲上述理由，xml:base 屬性最多只能使用在指定內容包裝資源的子目錄的相對路徑。下面就是一個使用 xml:base 來指定內部資源的相對路徑的例子：

When using xml:base in packaging, the xml:base path should not begin with a leading forward slash. As defined in RFC 2396, a path with a leading forward slash indicates the absolute path of that resource. Using a leading forward slash can easily be misinterpreted as declaring the document as the local host. With this in mind, the xml:base attribute is most useful for specifying relative paths to sub-directories containing content Package resources. Below is an example of using xml:base to specify the path to resources that are internal and relative.

```
<manifest xmlns = "http://www.imsglobal.org/xsd/imscp_v1p1"
    xmlns:imsmd = "http://www.imsglobal.org/xsd/imsmd_v1p2"
    xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation = "http://www.imsglobal.org/xsd/imscp_v1p1
      http://www.imsglobal.org/xsd/imscp_v1p1.xsd
      http://www.imsglobal.org/xsd/imsmd_v1p2
      http://www.imsglobal.org/xsd/imsmd_v1p2.xsd "
      identifier="Manifest1-CEC3D3-3201-DF8E-8F42-3CEED12F4198"
      version="IMS CP 1.1.4">
    <metadata>
       <schema>IMS Content</schema>
       <schemaversion>1.1</schemaversion>
       <imsmd:lom>
          <imsmd:general>
             <imsmd:title>
                <imsmd:langstring xml:lang="en_US">IMS Content Packaging Sample - A
Relative xml:base</imsmd:langstring>
             </imsmd:title>
          </imsmd:general>
       </imsmd:lom >
    </metadata>
    <organizations default="TOC1">
       <organization identifier="TOC1">
```

```
                <title>default</title>
                <item identifier="ITEM1" identifierref="RESOURCE1">
                    <title>Lesson 1</title>
                    <item identifier="ITEM2" identifierref="RESOURCE2">
                        <title>Introduction 1</title>
                    </item>
                    <item identifier="ITEM3" identifierref="RESOURCE3">
                        <title>Content 1</title>
                    </item>
                    <item identifier="ITEM4" identifierref="RESOURCE4">
                        <title>Summary 1</title>
                    </item>
                </item>
                <item identifier="ITEM5" identifierref="RESOURCE5">
                    <title>Lesson 2</title>
                    <item identifier="ITEM6" identifierref="RESOURCE6">
                        <title>Introduction 2</title>
                    </item>
                    <item identifier="ITEM7" identifierref="RESOURCE7">
                        <title>Content 2</title>
                    </item>
                    <item identifier="ITEM8" identifierref="RESOURCE8">
                        <title>Summary 2</title>
                    </item>
                </item>
            </organization>
        </organizations>
        <resources>
            <resource identifier="RESOURCE1" type="webcontent" href="lesson1.htm"
    xml:base="lesson1/">
                <file href="lesson1.htm"/>
                <file href="picture1.gif"/>
            </resource>
            <resource identifier="RESOURCE2" type="webcontent" href="intro1.htm"
    xml:base="lesson1/">
                <file href="intro1.htm"/>
                <file href="picture2.gif"/>
            </resource>
            <resource identifier="RESOURCE3" type="webcontent" href="content1.htm"
    xml:base="lesson1/">
```

```
            <file href="content1.htm"/>
            <file href="picture3.gif"/>
        </resource>
        <resource identifier="RESOURCE4" type="webcontent" href="summary1.htm"
xml:base="lesson1/">
            <file href="summary1.htm"/>
            <file href="picture4.gif"/>
        </resource>
        <resource identifier="RESOURCE5" type="webcontent" href="lesson2.htm"
xml:base="lesson2/">
            <file href="lesson2.htm"/>
            <file href="picture1.gif"/>
        </resource>
        <resource identifier="RESOURCE6" type="webcontent" href="intro2.htm"
xml:base="lesson2/">
            <file href="intro2.htm"/>
            <file href="picture2.gif"/>
        </resource>
        <resource identifier="RESOURCE7" type="webcontent" href="content2.htm"
xml:base="lesson2/">
            <file href="content2.htm"/>
            <file href="picture3.gif"/>
        </resource>
        <resource identifier="RESOURCE8" type="webcontent" href="summary2.htm"
xml:base="lesson2/">
            <file href="summary2.htm"/>
            <file href="picture4.gif"/>
        </resource>
    </resources>
</manifest>
```

下面則是使用 xml:base 來指定外部資源的絕對路徑的例子：

The following is an example of using xml:base to specify the path to resources that are external and absolute.

```
<?xml version="1.0"?>
<manifest identifier="MANIFEST1"
xmlns="http://www.imsglobal.org/xsd/ims_cp_rootv1p1">
    <metadata>
```

```
            <schema>IMS Content</schema>
            <schemaversion>1.1</schemaversion>
            <imsmd:lom>
                <imsmd:general>
                    <imsmd:title>
                        <imsmd:langstring xml:lang="en_US">IMS Content Packaging Sample - A
Remote xml:base</imsmd:langstring>
                    </imsmd:title>
                </imsmd:general>
            </imsmd:lom>
        </metadata>
        <organizations default="TOC1">
            <organization identifier="TOC1">
                <title>Big Title</title>
                <item identifier="ITEM1" identifierref="RESOURCE1">
                    <title>Lesson 1</title>
                    <item identifier="ITEM2" identifierref="RESOURCE2">
                        <title>Introduction 1</title>
                    </item>
                    <item identifier="ITEM3" identifierref="RESOURCE3">
                        <title>Content 1</title>
                    </item>
                    <item identifier="ITEM4" identifierref="RESOURCE4">
                        <title>Summary 1</title>
                    </item>
                </item>
            </organization>
        </organizations>
        <resources xml:base="http://repository.imsglobal.org/foo/bar/">
            <resource identifier="RESOURCE1" type="webcontent" href="lesson1.htm"/>
            <resource identifier="RESOURCE2" type="webcontent" href="intro1.htm"/>
            <resource identifier="RESOURCE3" type="webcontent" href="content1.htm"/>
            <resource identifier="RESOURCE4" type="webcontent" href="summary1.htm"/>
        </resources>
    </manifest>
```

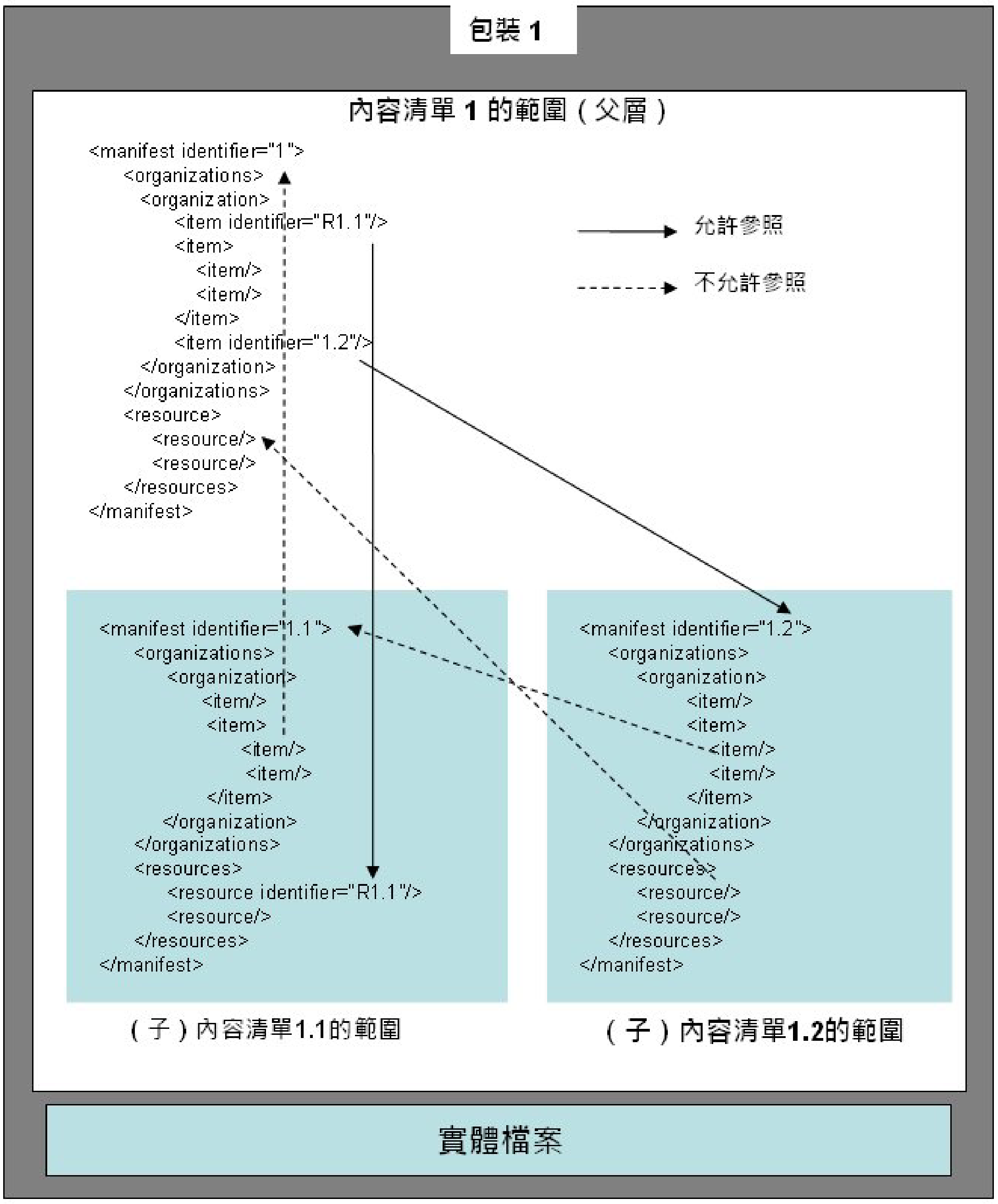
6.8.4 套裝範圍(Package Scope)

內容清單與子內容清單的範圍規則展示如圖 6.2。

The scoping rules for manifests and sub-manifests is shown in Figure6.2.

圖 6.2 子內容清單範圍規則

Figure 6.2 Scoping rules for sub-manifests.



Package1 的內容清單的範圍包含被定義在 Package1 的子內容清單及其本身，亦即 Package1 的內容清單(Manifest1)和依存在子套裝內的子內容清單-Manifest1.1 和 Manifest1.2。

The scope of Package 1’s manifest is considered to be itself and any sub-Manifests defined within Package 1. This includes the manifest of Package 1 (Manifest 1) and any sub-manifests found in the sub-packages - Manifest 1.1 and Manifest 1.2.

Package1.1 之內容清單的範圍爲自己與被定義在 Package1.1 的子內容清單。包括 Package1.1 的內容清單(Manifest1.1)和定義在子套裝內的子內容清單。圖 6.2 中 Package1.1 裡沒有定義子內容清單，所以範圍就是它自己。

The scope of Package 1.1’s manifest is itself and any sub-Manifests defined within Package 1.1. This includes the manifest of Package 1.1 (Manifest 1.1) and any sub-Manifests defined within Package 1.1. In Figure 6.2 there are no sub-Manifests defined in Package 1.1, so the scope is itself.

Package1.2 的內容清單的範圍爲自己與被定義在 Package1.2 的子內容清單。此包括了 Package1.2 的內容清單(Manifest1.2)和定義在子套裝內的子內容清單。在圖 6.2 中 Package1.2 裡沒有定義子內容清單，所以範圍就是它自己。

The scope of Package 1.2’s manifest is itself and any sub-Manifests defined within Package 1.2. This includes the manifest of Package 1.2 (Manifest 1.2) and any sub-Manifests defined within Package 1.2. In Figure 6.2 there are no sub-Manifests defined in Package 1.2, so the scope is itself.

套裝，特別是組織項目，不能引用內容清單範圍之外的內容清單元件（<resource>元件）。被引用的元件必須被包含在引用他們的同一個內容清單裡，包括套裝裡的子內容清單元件。上面的例子中，Package1 內容清單的元件可以引用依存於 sub- Manifest1.1 與 sub-Manifest1.2 的元件，因爲它們皆在 Package1 內容清單的範圍內。sub- Manifest1.1 與 sub-Manifest1.2 則只能引用它們自己的內容清單元件。子內容清單禁止引用（上圖中虛線）包含它的該內容清單元件（子內容清單元件不准引用父內容清單的元件）。

Packages, specifically organizational items, may not reference manifests elements (<resource> elements) that are outside the scope of manifest. Referenced elements must be contained in the same manifest from which they were referenced, including elements that are in sub-manifests within the Package. In the case above, Package 1 manifest’s elements can reference elements found in sub-Manifest 1.1 and sub-Manifest 1.2, since these are in scope of Package 1’s manifest. sub-Manifest 1.1 and sub-Manifest 1.2 can only reference manifest elements within itself. Sub-Manifests elements are prohibited from referencing (dashed lines above) manifest elements in any manifests in which they are contained (child manifests elements are not permitted to reference elements in any parent manifests).

目前允許一個父內容清單的‘item’引用子內容清單的特定‘resource’（圖 6. 2 R1.1 ‘resource’的引用），此類是否算是好作法尚有爭論。此係因爲有可替代的 XML 體系可以提供這樣的引用機制，因此在本標準之後的版本有可能禁止這樣的直接引用。

While it is currently permitted for an ‘item’ in the parent manifest to reference a specific ‘resource’ in a sub-Manifest (the reference to ‘resource’ R1.1 in Figure 6.2), it is debatable whether or not this is good practice. This is because alternative XML schemes are available to support such a referencing mechanism and so it is possible that this form of direct reference will be disallowed in later versions of this specification.

6.8.5<identifierref>參考元件(<identifierref> Referenced Elements)

下列元件可被 item 的 identifierref 的屬性參考使用：

(1) manifest 的識別符屬性（引用在參考的內容清單範圍內的全體 manifest）；

(2) resource 的識別符屬性（引用依存在參考的內容清單範圍內的子內容清單中的 resource）；

(3) item 的識別符屬性（引用依存在參考的內容清單範圍內的子內容清單中的 item）；

(4) organization 的識別符屬性（引用依存在參考的內容清單範圍內的子內容清單中的 organization）。

The following elements can be referenced using the identifierref attribute of an item:

(1) Identifier attribute of a manifest (references the entire manifest that is in scope of the referencing manifest);

(2) Identifier attribute of a resource (references the resource found in a sub-Manifest that is in scope of the referencing manifest);

(3) Identifier attribute of an item (references the item found in a sub-Manifest that is in scope of the referencing manifest);

(4) Identifier attribute of an organization (references the organization found in a sub-Manifest that is in scope of the referencing manifest).

6.9 Min/Max 封裝限制(Min/Max Binding Constraints)

在內容包裝資訊模型中，使用了 min/max 的概念。此概念假定了資料大小的限制，因此實作時必須要能支援該最大值的最低限度。在一個識別符的大小爲 1000 字元的例子中，每次實行時此識別符的大小即爲 1000 字元。某些開發情況雖可支援較大的尺寸、但互運性受到限制，因此只能保證前 1000 字元能確實地交換。

In the Content Packaging Information Model the concept of min/max was adopted. This concept assumes that the size constraints are defined such that an implementation must support the given value as the smallest possible maximum size. In the case of an identifier with a min/max size of 1000 characters then the smallest maximum size of the identifier supported in each and every implementation is 1000 characters. Some implementations may support larger sizes but interoperability is defined such that only the first 1000 characters is guaranteed to be exchanged consistently.

在 XML 架構中此限制並不具有強迫性。因此語法驗證的剖析器無法使用資訊模型中所定義的限制。也因此系統實作必須明確的支援 min/max 參數限制。

This constraint is not enforced in the XML Schema (XSD). Therefore a validating parser cannot enforce the constraint as defined by the Information Model. Therefore, system implementations must be such that all min/max constraints are explicitly supported.

6.10 使用 isvisible 屬性(Using the ‘isvisible’ Attribute)

‘isvisible’此屬性是用以表示當組織樹呈報給系統的‘user’使用者時‘item’ 是否會顯示。‘isvisible’的預設值爲‘true’並在‘item’沒使用此參數時採用。該特性不會被‘item’的子項繼承。一些表現的例子顯示在下面的表 6.1。

The ‘isvisible’ attribute is used to denote if the ‘item’ is to be visible when the organization tree is rendered for the‘user’ by the system. The default value for ‘isvisible’ is ‘true’ and this must be assumed if the attribute is not used on the ‘item’. This property is not inherited by the children of an ‘item’. Some examples of the rendering is shown in the following Table 6.1.

表 6.1 – 使用‘isvisible’屬性的範例

Table 6.1 - Examples of using the ‘isvisible’ attribute.

XML範例碼(Example XML Code)	執行項目(Rendered Items)

<item identifier="1">
   <title>A</title>
   <item identifier="2">
      <title>B</title>
      <item identifier="3">
         <title>C</title>
      </item>
   </item>
   <item identifier="4">
      <title>D</title>
   </item>
</item>
<item identifier="5">
   <title>E</title>
</item>
	A
	   B
	      C
	    D
	E

<item identifier="1" isvisible="false">
   <title>A</title>
   <item identifier="2">
      <title>B</title>
      <item identifier="3">
         <title>C</title>
      </item>
   </item>
   <item identifier="4">
      <title>D</title>
   </item>
</item>
<item identifier="5">
   <title>E</title>
</item>
	B
	   C
	D
	E

<item identifier="1" isvisible="false">
  <title>A</title>
  <item identifier="2" isvisible="true">
    <title>B</title>
    <item identifier="3" isvisible="true">
      <title>C</title>
    </item>
  </item>
  <item identifier="4" isvisible="true"> <title>D</title>
  </item>
</item><item identifier="5" isvisible="true">
  <title>E</title>
</item>
	B
	  C
	D
	E

<item identifier="1" isvisible="true">
  <title>A</title>
  <item identifier="2" isvisible="false">
    <title>B</title>
    <item identifier="3" isvisible="false">
      <title>C</title>
    </item>
  </item>
  <item identifier="4" isvisible="false">
    <title>D</title>
  </item>
</item>
<item identifier="5" isvisible="false">
  <title>E</title>
</item>
	A

<item identifier="1" isvisible="true">
  <title>A</title>
  <item identifier="2" isvisible="false">
    <title>B</title>
    <item identifier="3" isvisible="true">
      <title>C</title>
    </item>
  </item>
  <item identifier="4" isvisible="false">
    <title>D</title>
  </item>
</item>
<item identifier="5" isvisible="false">
  <title>E</title>
</item>
	A
	  C

<item identifier="1" isvisible="true">
  <title>A</title>
  <item identifier="2" isvisible="true">
    <title>B</title>
    <item identifier="3" isvisible="false">
      <title>C</title>
    </item>
  </item>
  <item identifier="4" isvisible="false">
    <title>D</title>
  </item>
</item>
<item identifier="5" isvisible="false">
  <title>E</title>
</item>
	A
	  B

7.驗證

W3C 的 XML1.0 格式容許兩種語法分析的類型：驗證與非驗證。非驗證的分析只注意文件的 well-formed 狀態，即確保遵守 XML 的語法規則。另一方面，驗證語法分析是執行完全 XML1.0 標準所必須的，即驗證語法分析必須遵守關於結構、資料類型和 Schema 指定的外部參考的全部規則。

7.Validation

The XML 1.0 Specification from the W3C allows for two types of parsers: validating and non-validating. Non-validating parsers are only concerned with the well-formedness of a document—that is, ensuring that the syntactic rules of XML have been followed. Validating parsers, on the other hand, are required to implement the full XML 1.0 Specification. This means that validating parsers must follow all of the rules concerning structure, data types, and external references that are specified by a schema.

Schema 記述文件裡有哪些元件存在、及該元件如何被組織。IMS 內容包裝標準對使用 XML 架構定義(XSD)提供了有限的指導。然而此類架構中，每個皆有不同的功能，任一架構皆可提供基本的文件驗證。建議基於 IMS 內容包裝標準編寫的任何內容清單文件皆能透過本標準所提供的 XSD 來驗證。

Schemas describe which elements may exist in a document and how those elements may be structured. The IMS Content Packaging specification provides limited guidance on the use of XML Schema Definition (XSD). While each of these schemas has different capabilities, any of these schemas can provide basic document validation. It is expected that any Manifest document in a Package that is written according to the IMS Content Packaging specification can be validated using the XSD schema available with this specification.

IMS 內容包裝標準件隨著一個 XSD(imscp_v1p1.xsd)。雖然在技術上用 DTD 來驗證文件是可行的，但卻無法用 DTD 來區分兩個使用同樣元件名稱的不同路徑元件（舉例來說，IMS 詮釋資料和 IMS 內容包裝皆有使用<resource>，但路徑不同，還有 IMS 內容包裝和 IMS 問題測試皆有使用<item>，但路徑不同）。比起修改 IMS 內容包裝資訊模型去符合 DTD 驗證的需求，內容工作團隊決定優先注目在 XML 架構的驗證上。

The IMS Content Packaging specification is accompanied by one XSD (imscp_v1p1.xsd). While it is technically feasible to validate documents that use DTDs, it is not possible to use a DTD to differentiate between two elements that use an element name in incompatible ways (for example IMS Meta-Data and IMS Content Packaging both use <resource> in meaningful, but incompatible ways, and IMS Content Packaging and IMS Question and Test both use <item> in meaningful, but incompatible ways). Rather than alter the IMS Content Packaging Information Model to adjust to the requirements of DTD validation, the Content Working Group made a decision to be forward-looking, towards XML Schemas, with respect to validation.

7.1 W3C SCHEMA 驗證(Validation)

IMS 已經更新內容包裝架構以支援 W3C XML 架構標準（日期）2001/05/02 的最終建議。目前有幾種商業的工具支援架構驗證包括了：Xerces、 XML Authority、XML Spy 及 Oracle parsers。

IMS has updated the Content Packaging Schema to support the Final Recommendation of the W3C XML Schema specification (dated) 2 May 2001. Currently, several commercial tools support Schema validation including: Xerces, XML Authority, XML Spy, and Oracle parsers.

‘xml:’命名空間屬性被定義在“http://www.w3.org/2001/xml.xsd”檔案。這是參考用以當作線上驗證或是將該檔案複製然後放在內容包裝的根節點目錄用以本地驗證。

The ‘xml:’ namespaced attributes are defined in file ‘http://www.w3.org/2001/xml.xsd’. This is the reference to be used for on-line validation or a copy of this file must be placed in the root of the content package for local validation.

請注意“http://www.w3.org/2001/xml.xsd”檔案由 W3C 維護持續更新，最新的版本為“http://www.w3.org/2001/03/xml.xsd”。

Note that the file ‘http://www.w3.org/2001/xml.xsd’ is always the most up to date version of the file ‘http://www.w3.org/2001/03/xml.xsd’ as maintained by W3C.

8.符合性

一個包裝標準的符合性對與 IMS 內容包裝標準有關的贊助者來說是重要的。符合性可以確保內容的可溝通性。內容業者與其客戶對它的期望是當內容在系統內、系統間、與跨網路時內容皆可順利地再被包裝與被承接的學習管理系統使用，而且電腦平台能支援教學內容與學習服務提供者。它也幫助學習管理系統業者、電腦平台與學習服務控制他們的資料儲存和工具的規模或是幫忙子系統處理內容包裝。

8. Conformance

Conformance to a packaging specification is an important issue for stakeholders involved with the IMS Content Packaging specification. Conformance clarifies content interoperability. It sets an expectation for content vendors and their customers about how that content will be repackaged, and possibly used by compliant LMSs, computing platforms supporting instructional content, and learning service providers as content moves about within systems, between systems, and across the Web. It also helps LMS vendors, computing platforms, and learning services to control the scope of their data stores and tools or sub-systems required to operate on content Packages.

此標準指出了二種層級的符合性來引導內容開發者、電腦平台、或學習服務怎樣去處理內容開發者在 IMS 內容清單檔案中元件與延伸的內容。符合性的相同層級應該指引那些需要在系統中、系統間或是透過網路再次包裝內容的使用者。

This specification addresses two levels of conformance to guide content developers in how LMS vendors, computing platforms, or learning services may deal with the elements and extensions content developers place within an IMS Manifest file. These same levels of conformance should guide those who repackage content for redistribution within their systems, across systems, or across the Web.

8.1 套裝符合性(Package Conformance)

爲了達到符合性，imsmanifest.xml 檔案和全部的資源皆會直接或間接地參照 IMS 內容包裝文件（即爲套裝交換檔案）。

For the purposes of conformance, an IMS Content Package is the relevant imsmanifest.xml file and all resources directly or indirectly referenced by this document (also known as the Package Interchange File).

8.1.1 套裝符合性 Level 0

(1) 此套裝在發佈媒材（檔案、CD-ROM 等）的根節點必須包含一個名爲 imsmanifest.xml 的檔案。

(2) 此套裝在發佈媒材（檔案、CD-ROM 等）的根節點必須包含任何直接參考的控制檔案(DTD、XSD)。

(3) imsmanifest.xml 檔必須包含 well-formed XML，它遵守描述於 IMS 內容包裝 XML 繫結標準第 3 章的 XML 格式。

(4) 如果 imsmanifest.xml 檔包含 IMS 詮釋資料，它必須包含命名空間的延伸性以包括基於 IMS 詮釋資料標準 v1.2.1 的詮釋資料。

(5) imsmanifest.xml 檔不可參考使用 XInclude 的元件。（此限制可能因它被 XML 語法普遍支援後解除）

(6) 依附於一個本地資源（例如：一個完全包容在套裝交換檔案裡的資源）的檔案皆必須以<file>元件在 imsmanifest.xml 檔的<resources>段中定義，而且必須放在置放 imsmanifest.xml 的目錄或子目錄中。

8.1.1 Package Conformance Level 0 (no extensions)

a) The Package must contain a file called imsmanifest.xml in the root of the distribution medium (archive file, CD-ROM, etc.);

b) The Package must contain any directly referenced controlling files used DTD, XSD) in the root of the distribution medium (archive file, CD-ROM, etc.);

c) The imsmanifest.xml file must contain well-formed XML that adheres to the XML format described in section 3 of the IMS Content Packaging XML Binding specification;

d) If the imsmanifest.xml file contains IMS Meta-Data, it must contain a namespace extension to include meta-data according to the IMS Meta-Data Specification v1.2.1;

e) The imsmanifest.xml file must not reference any elements using XInclude. (This requirement may be relaxed when it is generally supported in XML parsers);

f) All files that a local resource (i.e., a resource that is contained entirely within the Package Interchange File) is dependent on must be identified by <file> elements in the <resources> section of the imsmanifest.xml file and must be contained within the directory or sub-directories that contain imsmanifest.xml.

8.1.2 套裝符合性 Level 1

(1) 符合 Level 0 全部的要求（除了第 5 點之外）。

(2) imsmanifest.xml 檔可容納額外的命名空間延伸。如果補充的命名空間延伸使用 Schema 或修正 DTD 來描述與控制，那麼所有被直接參考的控制檔案皆必須包括在此套裝裡。

8.1.2 Package Conformance Level 1 (urilizes extensions)

(1) All level 0 conformance requirements (except ‘e’) apply;

(2) The imsmanifest.xml file may contain additional namespace extensions. If additional namespace extensions are described and controlled using a schema or modified DTD, then any directly referenced control files must be included in the Package.

8.2 系統與工具符合性(System and Tool Conformance)

為了達到符合性，向輸入、輸出、建立和處理 IMS 內容包裝的系統與工具要求系統與工具符合性。

For the purposes of conformance, system and tool conformance refers to the systems and tools that import, export, create, and manipulate IMS Content Packages.

8.2.1 系統與工具符合性 Level 0（可不保留延伸性）

(1) 一個遵守標準的系統或工具必須辨識與處理符合 level 0 或 level 1 的 IMS 內容包裝。處理 IMS 內容包裝的系統和工具的特徵與功能故意不指定。

(2) imsmanifest.xml 中以 IMS 內容包裝 XML 繫結標準 v1.1.4 與 IMS 詮釋資料標準 v1.2.1 或 IEEE P1484.12.3 延伸性標誌語言(XML)Schema 定義語言繫結學習物件詮釋資料標準草案呈現的全部元件必須保存以便再次傳遞。

(3) 命名空間延伸，除了 IMS 詮釋資料標準 v1.2.1 命名空間或是 IEEE P1484.12.3 延伸性標誌語言(XML)Schema 定義語言繫結學習物件詮釋資料命名空間標準草案，可能會被忽略與不能被再次傳遞。

8.2.1 System and Tool Conformance Level 0 (may not preserces extensions)

(1) A conforming system or tool must recognize and process any conforming IMS Content Package that conforms to level 0 or level 1. The features and functionality of systems and tools that process IMS Content Packages are purposely not specified;

(2) All elements of the IMS Content Packaging XML Binding Specification v1.1.4 and IMS Meta-Data Specification v1.2.1 or the IEEE P1484.12.3 Draft Standard for Extensible Markup Language (XML) Schema Definition Language Binding for Learning Object Metadata that are present in imsmanifest.xml must be preserved upon re-transmittal;

(3) Name-spaced extensions, other than the IMS Meta-Data Specification v1.2.1 namespace or the IEEE P1484.12.3 Draft Standard for Extensible Markup Language (XML) Schema Definition Language Binding for Learning Object Metadata namespace, may be ignored and may not be re-transmitted.

8.2.2 系統與工具符合性 Level 1（保留延伸性）

(1) 符合 Level 0 的第 1 點與第 2 點。

(2) 全部的命名空間延伸性皆必須保存以再次傳遞。

8.2.2 System and Tool Conformance Level 1 (preserces extensions)

(1) Level 0 conformance requirements (a) and (b) apply;

(2) All name-spaced extensions must be preserved upon re-transmittal.

8.3 IMS 套裝符合性的最佳實務建議(Best Practice Recommendations for IMS Package Conformance Levels)

本節包含支持 IMS 內容套裝的功能性與互運性的補充建議。

This section contains additional recommendations to support the functionality and interoperability of IMS Content Packages.

(1) 建議建立、傳送或重新套裝內容者，應在他們的公共網站公告他們所支援的 IMS 內容套裝符合性層級或系統與工具符合性層級。自創延伸性命名空間的組織或企業亦鼓勵公開其定義的 DTD 或 XSD；

A general recommendation to all who create, deliver, or repackage content is that they publish at their public Websites which level of the IMS Content Package Conformance Level or System and Tool Conformance Levels they support. An organization or enterprise that originates a namespace extension is encouraged to make public the DTD or XSD files that define it;

(2) 建議內容製作者將內容以符合預期之聚合或解聚的方式組織。亦即，如果內容製作者並不打算或預計讓其內容被解聚，則這些內容應該加密成一個整體的內容清單。相反的，子內容清單應該照他們所預期的聚合與解聚層級來組織內容；

It is expected that content producers will organize their content for expected aggregations or disaggregation. That is, if content producers do not expect, or desire their content to be disaggregated, it should be encoded in a monolithic manifest. Conversely, sub-Manifests should be used to organize content according to expected levels of aggregation and disaggregation;

(3) IMS 內容工作小組期待訓練系統、平台及學習空間之廠商能主動採用與其產品或服務之訓練社群相關之命名空間機制的元件。此外，內容建立者可能會想用專屬的命名空間，以便在其內容中支援更豐富的特色，且與上述廠商能互相溝通支援。因此，IMS內容工作小組強力建議，當內容在重新包裝並傳送至網路其他系統（或工具）時，系統與工具可重新建立原創的第三方命名空間及命名空間元件之IMS內容清單檔案。

The IMS Content Working Group expects that vendors of training systems, platforms, and learning spaces will actively use namespaced elements that are relevant to their product(s) or the training communities they serve. Additionally, content creators may want to use proprietary namespaces to support a richer set of features in their content than would otherwise be available, and negotiate support for those features with vendors of training systems, platforms, and learning spaces. Hence, the IMS Content Working Group strongly encourages systems and tools to recreate an originating IMS Manifest file’s use of third party namespaces and namespaced elements when such content is repackaged for transmission from their system or tool to elsewhere on the Web;

(4) 當聚合或解聚內容時，內容的再包裝應由原始套裝使用子內容清單或參考來引導到外部內容清單。亦即，準備聚合或解聚的課程的該部份將保留在一個子內容清單中。如此，系統或工具應該保留此原始子內容清單或外部參考內容清單，或是從它們的環境再包裝內容傳輸時能夠複製它們。最好子內容清單或外部參考內容清單裡的元件與屬性皆不需增加或刪除。

Content re-packagers should be guided by an original Package’s use of sub-Manifests or references to external manifests when aggregating or disaggregating content. That is, a portion of a course or curriculum that is a candidate for aggregation or disaggregation will be held in a sub-Manifest. So, a system or tool should preserve the original sub-Manifest(s) or externally referenced manifests or, be able to replicate them when repackaging content to export out of their environment. It is expected that there will be no additions or deletions to elements and attributes within a sub-Manifest or externally referenced manifest.

9.擴展性

爲了儘可能給開發者最高的延伸性，內容清單的 XML 繫結可自由擴充。作爲其它元件容器的元件皆可以再擴充以包容新的元件。包含資料類型（例如：字串、整數）的元件和封閉資料模型的元件不可以擴展。封閉資料模型元件的例子包括<schema>與<schemaversion>。擴展必須提供此擴展來源的參考（例如：經由命名空間）。

9.Extensibility

To allow developers the most flexibility possible, the XML binding of a manifest may be freely extended. All elements that serve as containers for other elements may be extended to include new elements. Elements that contain data types (e.g., string, integer) and elements with a ‘closed’ data model may not be extended. Examples of elements with a closed data model include <schema> and <schemaversion>. Extensions must provide references (e.g., via namespacing) to the source of the extensions.

至少會有兩種擴展的情況對開發者會發生問題。第一種情況是當需要其它內容包裝工具和業者之互運時。自訂的擴展之後若必須保持在全球的各個單獨團體間的互運性是很困難的。第二種情況是當開發者希望增加擴充以及提供或修改一個允許文件驗證的 schema 時。每個 schema（DTD 或 XSD）需要有不同驗證的處理擴充途徑。下面的章節就提供一些可以處理擴展途徑的簡要說明。

There are at least two cases where extensions can cause problems for developers. The first case is when interoperability with other content packaging tools and vendors is required. Custom extensions must then be agreed upon between individual parties making global interoperability very difficult. The second case is when a developer wishes to add extensions and also provide or alter a schema that will allow document validation. Each schema DTD or XSD) requires a different approach to handle extensions that can be validated. The following sections provide some brief explanations of approaches that may be used for handling extensions.

備考：下面的範例由XML片段組成來說明擴展的基礎概念。此例子不是well-formed和缺少一些資訊如控制文件（DTD或XSD）參考。完整的範例檔和它們的相關schema可在http://www.imsglobal.org/content/packaging/找到。

Note: The following examples consist of XML fragments to illustrate basic concepts of extensibility. These samples are not well formed and are missing some information such as any references to a control document (DTD or XSD). Complete sample files with their associated schemas can be found at http://www.imsglobal.org/content/packaging/.

9.1 擴展<metadata>(Extending <metadata>)

內容出版者或學習管理系統業者可能需要傳輸或儲存一些不被 IMS 詮釋資料標準 v1.2.1[MD,901]定義的詮釋資料。

A content publisher or LMS vendor may need to transport or store meta-data that is not defined by the IMS Meta-Data Specification v1.2.1 [MD, 901].

舉例來說，假如虛擬學習管理系統“LitWare Inc.”需要保存關於教學設計方法的詮釋資料用以產生課程。下面的步驟便說明如何使用基於 XML 架構定義語言的架構來簡單地達成此事。

For example, assume the fictitious LMS ‘LitWare Inc.,’ needs to maintain meta-data about the Instructional Design methodology used to create a course. The following steps illustrate how easily this can be done when using a schema based upon XML Schema Definition Language:

(1) 新增定義新元件的 XML 架構。對本例來說，XML 架構可由下述組成：

(1) Create an XML schema that defines the new element(s). For the given example, the XML schema could consist of the following:

```
<xsd:schema targetNamespace="http://www.litwareinc.net/xsd/litware"
     xmlns:xml="http://www.w3c.org/XML/1998/namespace"
     xmlns:xsd="http://www.w3c.org/2001/XMLSchema"
     xmlns:xsi="http://www.w3c.org/2001/XMLSchema-instance"
     xmlns="http://www.litwareinc.org/xsd/litware"
     elementFormDefault="qualified">
<xsd:element name="instructionaldesignmethodology" type="xsd:string"/>
</xsd:schema>
```

(2) 當輸出至學習管理系統時，imsmanifest.xml 裡的元件將顯示如下。下面的例子說明一種擴展 IMS 詮釋資料 v1.2[MD,501]的方法。

(2) When exporting to the learning management system, the element would appear as follows in imsmanifest.xml.

The following sample shows a way to extend the IMS Meta-Data v1.2 [MD, 501]:

```
<manifest identifier=MANIFEST1>
     <metadata>
          <schema>IMS Content</schema>
          <schemaversion>1.1</schemaversion>
          <imsmd:lom>
             <imsmd:general>
                <imsmd:title>
                          <imsmd:langstring xml:lang=en_US>Sample
                Manifest</imsmd:langstring>
                </imsmd:title>
                <imsmd:description>
                          <imsmd:langstring xml:lang=en_US>Metadata
                tensions</imsmd:langstring>
                </imsmd:description>
                <litware:instructionaldesignmethodology>LWI Mindmapping Methodology
                </litware:instructionaldesignmethodology>
```

```
            </imsmd:general>
          </imsmd:lom>
      </metadata>
      <organizations> . . .</organizations>
      <resources>. . .</resources>
  </manifest>
```

9.2 擴展<organizations>(Extending<organizations>)

未來預期會出現許多不同的內容組織的方法。ADL一直持續發展一種如此與他的SCORM版本發行連繫的方法。當此標準發表時，內容包裝網站(http://www.imsglobal.org/content/packaging/)之內容清單範例（包括繫結與範例）正在開發當中，而且可能不會是ADL訂定SCORM的最終版本。雖然在此範例內表達的一些想法可能尚未完成而且檔案不能完全驗證，但它仍然爲IMS內容包裝標準如何容許不同的內容組織體系實際外掛到套裝內容清單檔案，提供了一個好的概念。

It is expected that over time, many different approaches to content organization will emerge. The ADL has been developing one such approach in connection with its releases of SCORM versions. At the time of this specification release, the sample manifest, included with the Bindings and Examples from the Content Packaging website (http://www.imsglobal.org/content/packaging/) was a work in progress and may not be the final direction the ADL takes in future versions of SCORM. While some of the ideas expressed in this sample may not be complete and the file can not be properly validated, it still provides a good conceptual model of how the IMS Content Packaging specification allows different content organization schemes to essentially ‘plug-in’ to a Package Manifest file.

9.3 擴展<resources>(Extending<resources>)

使用外部與內嵌參考來擴展<resources>是內容包裝的顯著特徵。不過在 IMS 在最佳作法指南中提出擴展<resources>之前，目前 IMS 仍然持續在此部分做更多的測試與工作。

Extending <resources> using both external and in-line references is an important feature of Content Packaging. However, IMS is currently doing more testing and work in this area before providing samples of extending <resources> in this Best Practice Guide.

9.4 用 DTD 擴展(Extending with DTDs)

上面的例子，Schema 的內容模型必須被開放以便能夠擴展。爲了用 IMS 內容包裝 DTD 實現一樣的目標，必須產生一個包含擴展性的新 DTD。該 DTD 跟 IMS 內容包裝 DTD 不同。此途徑允許一個文件來驗證其擴展性，但它限制了內容套裝的互運性。

In the examples above, the content models of the schemas must be ‘open’ to enable extensibility. To accomplish the same goal using the IMS Content Packaging DTD, a new DTD must be created to include the extensions. Such a DTD would differ from the IMS Content Packaging DTD. This approach would allow a document to be validated with extensions in it, but it limits the interoperability of the content Package.

附錄 A

支援文件

Supporting Files

有許多支援文件伴隨 IMS 內容包裝標準文件，而且可從下載的.zip 檔(imscp_v1p1p4.zip)中取得。此 zip 檔中的檔案如下：

imscp_infov1p1p4.pdf	IMS 內容包裝資訊模型
\imscp_bindv1p1p4.pdf	IMS 內容包裝 XML 繫結
\imscp_bestv1p1p4.pdf	IMS 內容包裝最佳作法指南（即本文件）
\imscp_sumcv1p1p4.pdf	IMS 內容包裝修改沿革
\schema\ imscp_v1p1.xsd	IMS 內容 XML 架構，1.1.4 版
\samples\All_Elements	使用內容包裝元件說明一個簡單的內容清單。
\samples\QTI_Example	說明一個簡單的內容清單包裝 QTI 元件。
\samples\Full_Metadata	說明一個使用了 IMS 內容包裝標準定義的全部元件與屬性的內容清單。
\samples\Multiple_Organizations	說明多種<organizations>的使用，提供對一課程的不同路徑。
\samples\Simple_Manifest	說明一個簡單內容清單。
\samples\Sub_Manifests	說明子內容清單之使用以促進再利用。此處例子舉簡單內容清單爲例，並用子內容清單來實作。

A number of supporting files accompany the IMS Content Packaging specification documents and are available in the download .zip file (imscp_v1p1p4.zip). The files in the zip file are as follows:

imscp_infov1p1p4.pdf	IMS Content Packaging Information Model
\imscp_bindv1p1p4.pdf	IMS Content Packaging XML Binding
\imscp_bestv1p1p4.pdf	IMS Content Packaging Best Practice Guide (this document)
\imscp_sumcv1p1p4.pdf	IMS Content Packaging Summary of Changes
\schema\ imscp_v1p1.xsd	IMS Content XML Schema, version 1.1.4
\samples\All_Elements	Illustrates a simple manifest using Content Packaging elements.

\samples\QTI_Example	Illustrates a simple manifest packaging QTI elements.
\samples\Full_Metadata	Illustrates a manifest that uses all elements and attributes defined in the IMS Content Packaging specification.
\samples\Multiple_Organizations	Illustrates the use of multiple <organizations>, to provide different paths through a course.
\samples\Simple_Manifest	Illustrates a simple manifest.
\samples\Sub_Manifests	Illustrates the use of sub-Manifests to promote reuse. This example takes the Simple Manifest example, and implements it using sub-Manifests.

附錄 B

補充資源

Additional Resources

B1-各項文件

IMS 內容文件

IMS 內容包裝資訊模型：

http://www.imsglobal.org/content/packaging/

IMS 內容包裝XML繫結：

http://www.imsglobal.org/content/packaging/

IMS內容包裝最佳實作指導綱要：

http://www.imsglobal.org/content/packaging/

IMS 詮釋資料文件

IMS 詮釋資料最佳作法與實施指南：

http://www.imsglobal.org/metadata/

IMS 學習資源詮釋資料資訊模型：

http://www.imsglobal.org/metadata/

IMS 一般參考文獻

IMS 一致性獨立位置資源標識手冊：

http://www.imsglobal.org/implementationhandbook/imsrid_handv1p0.html

使用IMS內容包裝來包裝LIP和其他IMS標準的實例：

http://www.imsglobal.org/implementationhandbook/

ADL/AICC Documents

可分享的內容物件參考模型: http://www.adlnet.org/

Aviation Industry CBT Committee (AICC) API for Web Implementation: http://www.aicc.org/

Internet Engineering Task Force (IETF)

RFC 2396：一致性資源識別符 (URI)：http://www.ietf.org/rfc/rfc2396.txt

IEEE

IEEE LTSC 1484.12 學習物件詮釋資料：http://www.ltsc.ieee.org/wg12/

XML

W3C XML 1.0 版標準：http://www.w3.org/TR/1998/REC-xml-19980210

W3C XML 命名空間建議：http://www.w3.org/TR/1999/REC-xml-names-19990114

XML包含技術報告：http://www.w3.org/TR/xinclude

W3C XML架構建議：http://www.w3.org/TR/2001/REC-xmlschema-0-20010502/

B1-Various Documents

IMS Content Documents

IMS Content Packaging Information Model:
http://www.imsglobal.org/content/packaging/

IMS Content Packaging XML Binding:
http://www.imsglobal.org/content/packaging/

IMS Content Packaging Best Practice Guide:
http://www.imsglobal.org/content/packaging/

IMS Meta-Data Documents

The IMS Meta-Data Best Practice and Implementation Guide:
http://www.imsglobal.org/metadata/

The IMS Learning Resource Meta-Data Information Model:
http://www.imsglobal.org/metadata/

IMS General Reference

IMS Persistent, Location-Independent Resource Identifier Handbook:
http://www.imsglobal.org/implementationhandbook/imsrid_handv1p0.html

Using IMS Content Packaging to Package Instances of LIP and Other IMS Specifications:
http://www.imsglobal.org/implementationhandbook/

ADL /AICC Documents

Sharable Content Object Reference Model: http://www.adlnet.org/

Aviation Industry CBT Committee (AICC) API for Web Implementation:
http://www.aicc.org/

Internet Engineering Task Force (IETF)

RFC 2396: Uniform Resource Identifiers (URI): http://www.ietf.org/rfc/rfc2396.txt

IEEE

IEEE LTSC 1484.12 Learning Object Metadata: http://www.ltsc.ieee.org/wg12/

XML

XML Version 1.0 specification of the W3C: http://www.w3.org/TR/1998/REC-xml-19980210

XML Namespace Recommendation of W3C: http://www.w3.org/TR/1999/REC-xml-names-19990114

XML Inclusion Technical Report: http://www.w3.org/TR/xinclude

XML Schema Recommendation of W3C: http://www.w3.org/TR/2001/REC-xmlschema-0-20010502/

B2-命名空間與架構參考

使用在 XML 架構檔案的命名空間、檔案命名與命名空間字首的 XML 實例如下：

標準	命名空間	檔案名稱	字首
內容包裝	http://www.imsglobal.org/xsd/imscp_v1p1	imscp_v1p1.xsd	imscp:
詮釋資料	http://www.imsglobal.org/xsd/imsmd_v1p2	imsmd_v1p2.xsd	imsmd:
學習者資訊包裝標準	http://www.imsglobal.org/xsd/imslip_v1p0	imslip_v1p0.xsd	imslip:
問題與測驗互通性標準	http://www.imsglobal.org/xsd/imsqti_v1p1	imsqti_v1p1.xsd	imsqti:
簡易編序	http://www.imsglobal.org/xsd/imsss_v1p0	imsss_v1p0.xsd	imsss:
學習設計	http://www.imsglobal.org/xsd/imsld_v1p0	imsld_v1p0.xsd	imsld:

B2- Namespacing and Schema Reference

The namespaces, filenames, and namespace prefixes for XML instances using the XML Schema files are as follows:

	Namespace	Filename	Prefix
Content Packaging	http://www.imsglobal.org/xsd/imscp_v1p1	imscp_v1p1.xsd	imscp:
Meta-Data	http://www.imsglobal.org/xsd/imsmd_v1p2	imsmd_v1p2.xsd	imsmd:
LIP	http://www.imsglobal.org/xsd/imslip_v1p0	imslip_v1p0.xsd	imslip:
QTI	http://www.imsglobal.org/xsd/imsqti_v1p1	imsqti_v1p1.xsd	imsqti:
Simple Sequencing	http://www.imsglobal.org/xsd/imsss_v1p0	imsss_v1p0.xsd	imsss:
Learning Design	http://www.imsglobal.org/xsd/imsld_v1p0	imsld_v1p0.xsd	imsld:

本標準所提供的全部範例，全列表於附錄 A，所用的 Schema XSD 檔案位於 IMS 站點。靠著 IMS 站點上的 XSD 檔，標準編輯者可使用 XML Spy 5.1 版和 Turbo XML 2.3.1 版來驗證附錄 A 裡的每個範例。預期其他的 XML 架構用語法分析器也可以驗證範例檔案，只要此分析器能找到線上 XML 架構檔案。使用線上架構檔案參考（請參照下面的線上 XSD 檔案範例）是最佳作法，在 IMS 站點的 XSD 檔案是最新的。要使用此線上 XSD 檔案需要讓分析器有打開、實用的網際網路連線。如果無法使用網際網路連線或是使用者希望能在本地驗證，他們需要去改變在他們例子裡的命名空間宣告，以匹配下面的本地 XSD 檔案例子。

All of the samples provided with this specification, as listed in Appendix A, make use of the schema XSD) files located on the IMS website. The specification editors used XML Spy v5.1 and Turbo XML v2.3.1 to validate each of the samples listed in Appendix A, against the XSD files on the IMS website. It is expected that other XML Schema-capable parsers will also validate sample files as long as the parser is able to locate the online XML Schema files. It is best practice to use the online Schema file references (see Online XSD Files example below) as the XSD files on the IMS website will be the most up-to-date. Using the online XSD files requires the parser to have an open, functional connection to the Internet. If, however, an Internet connection is not available or users wish to validate files locally, they will need to change the namespace declarations in their samples to match the Local XSD Files example below.

線上XSD檔案

使用IMS站點提供的XSD檔的XML實例，在根節點<manifest>元件的宣言如下格式：

Online XSD Files

For those XML instances using the XSD files as located on the IMS website, the declaration in the root <manifest> element is of the form:

```
<manifest
    xmlns=http://www.imsglobal.org/xsd/imscp_v1p1xmlns:imsmd=http://www.ims
    global.org/xsd/imsmd_v1p2xmlns:xsi=http://www.w3.org/2001/XMLSchema-in
    stancexsi:schemaLocation="http://www.imsglobal.org/xsd/imscp_v1p1
                        http://www.imsglobal.org/xsd/imscp_v1p1.xsd
                        http://www.imsglobal.org/xsd/imsmd_v1p2
                        http://www.imsglobal.org/xsd/imsmd_v1p2.xsd"
    identifier="Manifest01" version="IMS CP 1.1.4">
```

本地XSD檔

可本地使用的XSD檔的XML實例，在實例相同的目錄，在根節點<manifest>元件的宣言如下格式：

Local XSD Files

For XML instances in which the XSD files are locally available, in the same directory as the instance, the declaration in the root <manifest> element is of the form:

```
<manifest
    xmlns="http://www.imsglobal.org/xsd/imscp_v1p1"xmlns:ims
    md="http://www.imsglobal.org/xsd/imsmd_v1p2"xmlns:xsi="h
```

ttp://www.w3.org/2001/XMLSchema-instance"xsi:schemaLoc

ation="http://www.imsglobal.org/xsd/imscp_v1p1.xsd

imscp_v1p1.xsdhttp://www.imsglo

bal.org/xsd/imsmd_v1p2imsmd_v1

p2.xsd"

identifier="Manifest01" version="IMS CP 1.1.4">

‘version’ 屬性是可選擇的。

附錄 C

一致化

Harmonization

IMS 標準間的一致化是重要的，IMS 堅持確保其標準對全部標準的詞彙、元件名稱和其它皆用同樣的對策。關於一致化或 IMS 內容包裝標準與其它 IMS 標準的實行範例的資訊，請看下面：

對LIP與其它IMS標準使用IMS內容包裝的包裝實例。一般使用手冊，說明如何包裝LIP實例也可用於包裝詮釋資料，QTI或Enterprise。要下載此份文件，請參考IMS站點的使用手冊部分：
http://www.imsglobal.org/implementationhandbook/

Harmonization between IMS specifications is important and IMS is committed to ensuring its specifications use similar strategies for vocabularies, GUIDs, element names, and others across all standards. For information about harmonization or sample implementations of the IMS Content Packaging specification and other IMS specifications,see the following:

Using IMS Content Packaging to Package Instances of LIP and other IMS specifications. A general implementation handbook illustrating how to package instances of LIP that could also be applied to packaging instances of Meta-Data, QTI, or Enterprise. To download this document, visit the Implementation Handbook portion of the IMS website: http://www.imsglobal.org/implementationhandbook/.

附錄 D

未來發展方向

Possible Future Directions

本章描述本內容包裝工作團隊考慮對未來發佈的版本建議的公開內容。

表格 D1 已定未來內容的列表

內容識別號	註解
CP113-49	以 URNs 做爲識別符或是使用 PLIDs 代替 xs:ID。
CP113-51	說明 ‘isvisible’ 屬性的用法。
CP113-55	[identifierref]必需定義的更嚴謹清楚。
CP113-56	sub-Manifests 的通用語義詮釋。
CP113-62	sub-Mmanifests 具體化。
CP113-63	提供非必備的描述建議。
CP113-64	建立本標準的 RDF 版本。
CP113-65	Title 的語法。
CP113-66	對繫結的布林值的解釋。
CP113-68	按‘meta-data’元件擴充 ADL ‘location’。
CP113-80	關於最佳作法指南中的包裝範圍不夠清楚的地方。
CP113-110	新增“variation”元件。
CP113-161	在 manifest 中的額外檔案。

This section describes open issues that the Content Packaging Working Group consider as recommendations for future version releases.

Table D1 List of issues to be resolved in the future.

Issue Identifier	Comment
CP113-49	Using URNs as identifiers or use PLIDs instead of xs:ID.
CP113-51	Clarification on the usage of the ‘isvisible’ attribute.
CP113-55	[identifierref] needs to be made more solid and clearly defined.

CP113-56	Common semantic interpretation of sub-Manifests.
CP113-62	Externalizing sub-Mmanifests.
CP113-63	Provide optional presentation hints.
CP113-64	Creating an RDF version of the spec.
CP113-65	Language for Title.
CP113-66	Ambiguity in boolean values in binding.
CP113-68	ADL ‘location’ extension for the ‘meta-data’ element.
CP113-80	Lack of clarity in package scope in BPG.
CP113-110	Adding “variation” element.
CP113-161	Extra files in manifest.

英中名詞對照表

	-A-
Amendment	修正
	-B-
Binding	繫結
	-C-
Character set	字元集
Conformance	符合性
Conformance statement	符合性聲明
Container	容器
Content	內容
Content Packaging, CP	***內容包裝***
	-D-
Dependency	***從屬物***
Document Type Definition, DTD	文件型式定義
Dynamic sequencing	***動態排序***
	-E-
Element	元件
	-F-
File	檔案
	-G-
	-H-
	-I-
Identifier	識別符
Information Model	資訊模型
Item	項目
	-J-
	-K-
	-L-
Learning Management System, LMS	學習管理系統
	-M-
Manifest	***內容清單***
Metadata	詮釋資料
	-N-
	-O-
Organization	組織
	-P-
Package	***套裝***
Parameter	參數
Persistent Location Independent Resource Identifier, PLIRI	*永久性定位獨立資源識別符*
Resource	資源
	-Q-
	-R-
	-S-

Schema	*架構*
Sub-manifest	*子內容清單*
	-T-
Title	標題
Traverse	遍歷
Type	型式
	-U-
	-V-
Version	版本
	-W-
World Wide Web Consortium, W3C	全球資訊網聯盟
	-X-
Extensible Mark-up Language, XML	可延伸標誌語言
	-Y-
	-Z-

內容包裝－ 英文草案

CNS	IMS Content Packaging	General No..	xxx-x
		Classified No.. .	xxxx-x

IMS Content Packaging

Contents

Data of Approval	**Bureau of Standards,Metrology and Inspection**	Data of Revision May,5,2007

1. Scope

1.1 Introduction

The IMS Content Packaging specification describes data structures, XML binding and accompanying best practices that are used to provide interoperability for Internet based content with content creation tools, learning management systems (LMS), and run time environments. The scope of the IMS Content Packaging specification is focused on defining interoperability between systems that wish to import, export, aggregate, and disaggregate Packages of content.

1.2 Relationship to Other Specifications

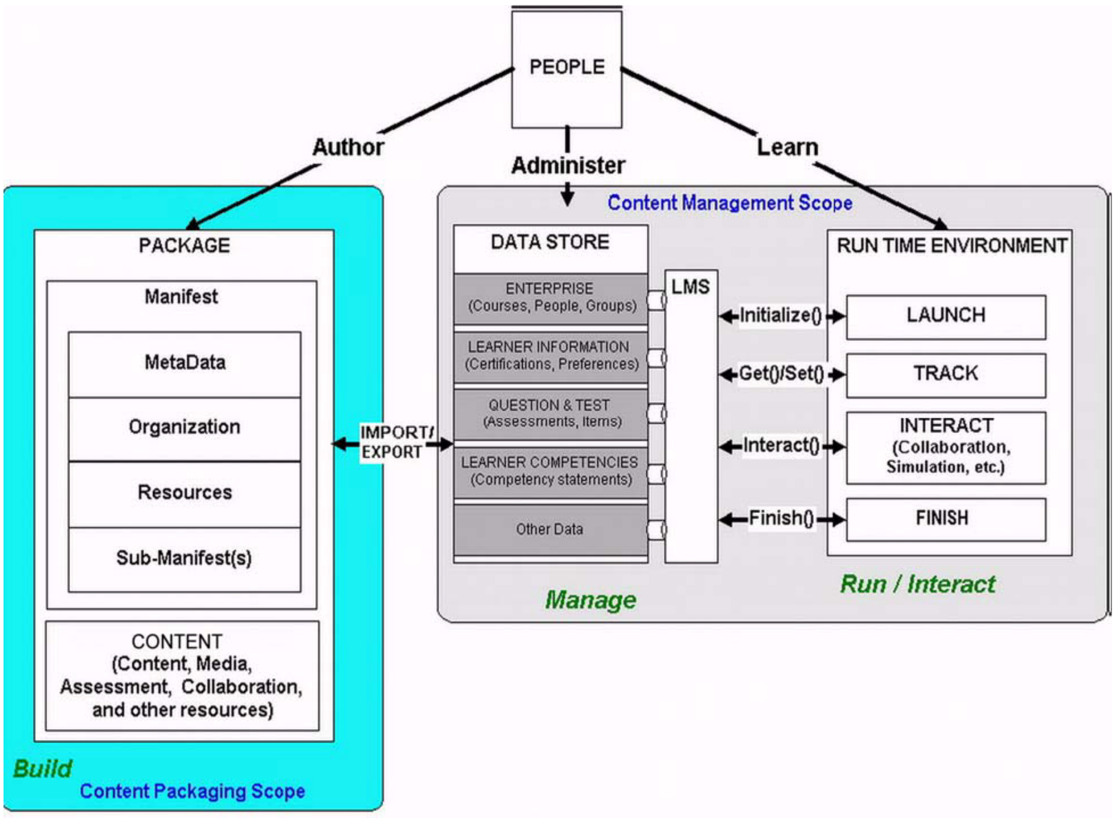
The entire, extended scope of the IMS Content Packaging specification is complemented by the overall goals of the IMS Content framework. Those goals are to provide enough guidance, through this specification, that people may build, manage, and interact with interoperable, online learning materials.

The following historical work was considered in the development of the original version 1.1 framework:

(1) IMS API draft specification version 0.6 (6/98);
(2) IMS Packaging draft specification version 0.6 (2/99);
(3) The Aviation Industry CBT Committee's (AICC) API for Web Implementation of AICC/IEEE CMI specification (9/99);
(4) The Advanced Distributed Learning Initiative's (ADL) Sharable Content Object Reference Model (11/99);
(5) The Microsoft Learning Resource Interchange (LRN) specification (01/00).

The scope of the IMS Content specification was captured in a diagram through a series of meetings and group discussions. This expanded view of the Content scope is depicted in Figure 1.1.

Figure 1.1 IMS Content framework

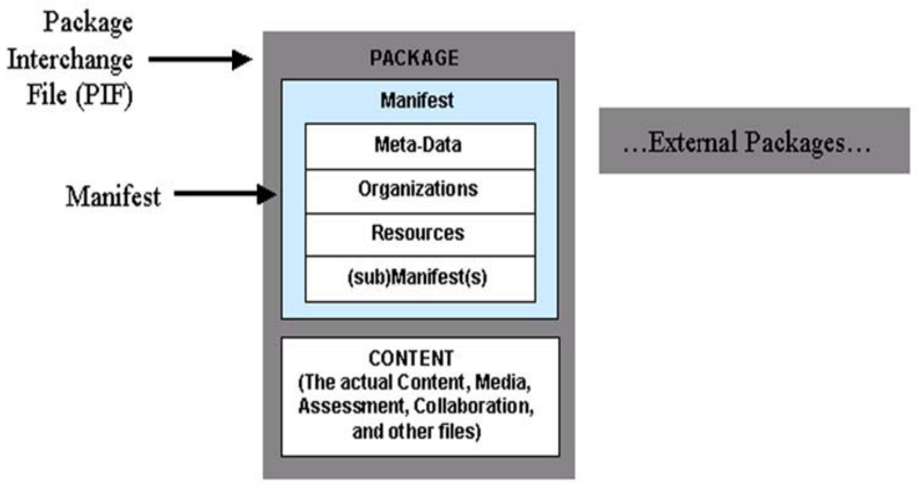

PEOPLE
Author
Administer
Learn
PACKAGE
Manifest
MetaData
Organization
Resources
Sub-Manifest(s)
CONTENT (Content, Media, Assessment, Collaboration, and other resources)
Build
Content Packaging Scope
IMPORT/ EXPORT
Content Management Scope
DATA STORE
ENTERPRISE (Courses, People, Groups)
LEARNER INFORMATION (Certifications, Preferences)
QUESTION & TEST (Assessments, Items)
LEARNER COMPETENCIES (Competency statements)
Other Data
LMS
Initialize()
Get()/Set()
Interact()
Finish()
RUN TIME ENVIRONMENT
LAUNCH
TRACK
INTERACT (Collaboration, Simulation, etc.)
FINISH
Manage
Run / Interact

The complete, identified scope of the IMS Content framework is large and complex. To reduce the complexity and decrease the amount of time needed to complete a first specification, the scope was broken down into three, main parts: Content Packaging, Data Model, and Run Time Environment. Each of these topics requires additional explanation and each is described in more detail in the following sections.

(1)Content Packaging

The IMS Content Packaging portion of the IMS Content framework represents the section that deals with the issues of content resource aggregation, course organization, and meta-data. All of the documents that comprise the IMS Content Packaging specification are focused on the scope represented in Figure 1.2.

Figure 1.2 IMS Content Package Scope


Package Interchange File (PIF)
Manifest
PACKAGE
Manifest
Meta-Data
Organizations
Resources
(sub)Manifest(s)
CONTENT (The actual Content, Media, Assessment, Collaboration, and other files)
…External Packages…

(2) Data Model

A future version of an IMS Content specification will address important, core issues of a general and extendable content data model. The data model represents that portion of the IMS Content framework where content is imported, stored, managed, and manipulated for instructional purposes. LMS vendors and computer platform vendors will play a key role in defining this portion of the specification.

A future IMS Content specification will also take into account how the IMS Enterprise, Question and Test Interoperability, and Learner Information Package specifications play a role in the data model. Other efforts such as the work that has been done within the ADL and AICC are being considered to determine which parts we can agree on that are common across all domains and which parts are specific to a particular community. The content team will also carefully determine a mechanism for how extensions to the data model may be represented so that different communities can use the IMS Content framework.

(3) Run Time Environment

A future IMS Content specification will deal also with the issues surrounding run time environments. The run time environment portion of the IMS Content framework represents the point where learners will interact with the content presented to them. One of the key requirements for this portion of the specification will be the identification of standard mechanisms to enable communication between a run time environment and an LMS.

2. Terms and definitions

2.1General Terms

2.1.1 ADL

Advanced Distributed Learning Initiative was started by the United States

White House in 1997, and aims to advance the use of online training.

2.1.2 AICC

Aviation Industry CBT Committee is a membership-based international forum that develops recommendations on interoperable learning technologies for the aviation industry.

2.1.3 character set

The characters used by a computer to display information.

2.1.4 choice

One of the possible responses that a test taker might select. Choices contain the correct answers and distracters.

2.1.5 conformance statement

A conformance statement provides a mechanism for customers to fairly compare vendors of assessment tools and content.

2.1.6 database

A collection of information/data, often organized within tables, within a computer’s mass storage system.Databases are structured in a way to provide for rapid search and retrieval by computer software. The following databases are used by testing systems: item, test definition, scheduling, and results.

2.1.7 DTD

Document Type Definition.

2.1.8 dynamic sequencing

The sequencing of items or sections is based upon previous responses from a test taker.

2.1.9 element

An XML term that defines a component within an XML document that has been identified in a way a computer can understand.

2.1.10 element contents

An XML term used to describe the content of the element.

2.1.11 element attributes

Provides additional information about an element.

2.1.12 IEEE

Institute of Electrical and Electronics Engineers that provides a forum for developing specifications and standards.

2.1.13 IMS

An organization dedicated to developing specification for distributed learning.

2.1.14 LTSC

Learning Technology Standards Committee.

2.1.15 LMS

Learning Management System which is the system responsible for the management of the learning experience.

2.1.16 Meta-data

Meta-data: Descriptive information about data. Can be thought of as data about data. IMS specifications typically use meta-data to describe learning resources.

2.1.17 W3C

World Wide Web Consortium.

2.1.18 XML

Extensible Mark-up Language is a specification, produced by the W3C.

2.1.19 XSD

XML Schema Definition.

2.2Content Packaging Elements and Attributes

2.2.1 default

Indicates which organization scheme is the default one.

2.2.2 dependency

Identifies the location of a resource that contains dependent files.

2.2.3 file

A reference to a file that a resource is dependent on.

2.2.4 CPI

Content & Packaging Interchange

2.2.5 href

A reference to a URL.

2.2.6 identifier

An identifier that is unique within the manifest.

2.2.7 identifierref

A reference to an identifier in the manifest or in a resource.

2.2.8 isvisible

Indicates whether or not an item is displayed when the package is displayed or rendered.

2.2.9 item

A node within the organization.

2.2.10 manifest

A reusable unit of instruction. Encapsulates meta-data, organizations, and resource references.

2.2.11 metadata

Meta-data describing the manifest.

2.2.12 organization

Defines a particular hierarchical organization.

2.2.13 organizations

Describes one or more structures, or organizations for this package.

2.2.14 parameters

Static parameters to be passed to the resource at launch time.

2.2.15 resource

A reference to a resource.

2.2.16 resources

A collection of references to resources. There is no assumption of order or hierarchy.

2.2.17 schema

Describes the schema that defines and controls the manifest.

2.2.18 schemaversion

Describes version of the above schema (e.g., 1,0, 1.1).

2.2.19 title

Title of the organization.

2.2.20 type

Indicates the type of resource.

2.2.21 version

Identifies the version of this manifest (e.g., 1.0).

2.2.22 xml base

Provides a relative path offset for relative URIs in the package.

3. Normative reference

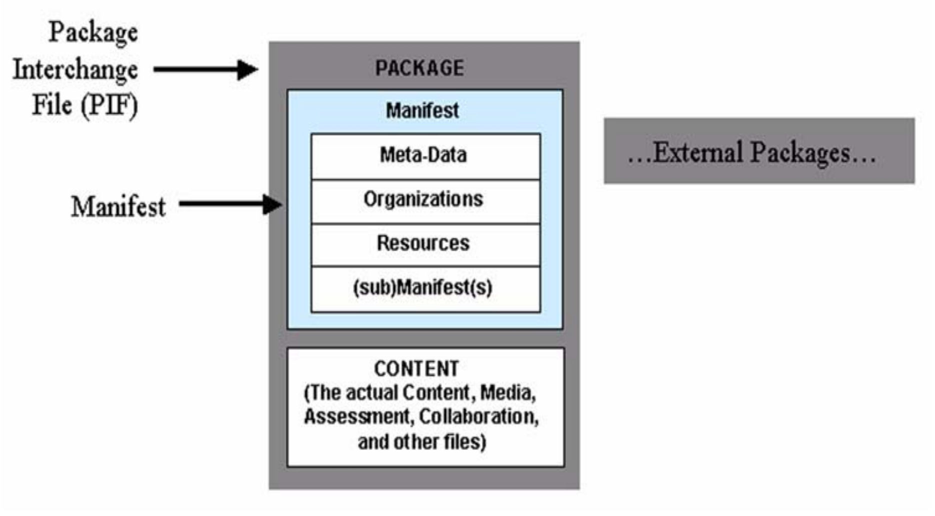
[CP, 04a]	IMS Content Packaging Information Model v1.1.4, C.Smythe, A.Jackl, IMS Global Learning Consortium, Inc., October 2004.
[CP, 04b]	IMS Content Packaging XML Binding v1.1.4, C.Smythe, A.Jackl, IMS Global Learning Consortium, Inc., October 2004.
[CP, 04c]	IMS Content Packaging Best Practice and Implementation Guide v1.1.4, C.Smythe, A.Jackl, IMS Global Learning Consortium, Inc., October 2004.
[CP, 04d]	IMS Content Packaging Summary of Changes v1.1.4, C.Smythe, A.Jackl, W.Kraan, IMS Global Learning Consortium, Inc., October 2004.
[IMSBUND, 01]	Using IMS Content Packaging to Package Instances of LIP and Other IMS Specifications v1.0, B.Olivier, M.McKell, IMS Global Learning Consortium, Inc., August 2001.
[IMSPLID, 01]	IMS Persistent, Location-Independent, Resource Identifier Implementation Handbook v1.0, M.McKell, IMS Global Learning Consortium, Inc., April 2001.

[ISO/IEC10646]	ISO (International Organization for Standardization). ISO/IEC 10646-1993 (E). Information technology - Universal Multiple-Octet Coded Character Set (UCS) - Part 1: Architecture and Basic Multilingual Plane. [Geneva]: International Organization for Standardization, 1993 (plus amendments AM 1 through AM 7).
[MD, 04]	IMS Meta-data Best Practice Guide for IEEE 1484.12.1-2002 Standard for Learning Object Metadata v1.3, P.Barker, L.Campbell, A.Roberts, IMS Global Learning Consortium, Inc., May 2004.
[MD, 901]	IMS Learning Resource Meta-data v1.2.1, S.Thropp, M.McKell, IMS Global Learning Consortium, Inc., September 2001.
[MD, 501]	IMS Learning Resource Meta-data v1.2, T.Anderson, M.McKell, IMS Global Learning Consortium, Inc., May 2001.
[Unicode, 96]	The Unicode Consortium. The Unicode Standard, Version 2.0. Reading, Mass.: Addison-Wesley Developers Press, 1996.
[XML, 98]	XML 1.0 Specification of the W3C: http://www.w3.org/TR/1998/REC-xml-19980210.
[XML, 99]	XML Namespace Recommendation of W3C: http://www.w3.org/TR/1999/REC-xml-names-19990114. XML Schema Recommendation of W3C: http://www.w3.org/TR/2001/REC-xmlschema-0-20010502/.

4. IMS Content Packaging Conceptual Model

Figure 4.1 is a conceptual diagram that illustrates the components of the IMS Content Packaging Information Model. As indicated in the IMS Content Packaging Best Practice Guide, this is part of the larger IMS Content Framework, which forms the basis for this and future specifications.

Figure 4.1 IMS Content Package Scope


Package Interchange File (PIF)
Manifest
PACKAGE
Manifest
Meta-Data
Organizations
Resources
(sub)Manifest(s)
CONTENT (The actual Content, Media, Assessment, Collaboration, and other files)
…External Packages…

4.1 Conceptual Model Discussion

The IMS Package depicted in Figure 4.1 consists of two major elements: a special XML file describing the content organization and resources in a Package, and the file resources being described by the XML. The special XML file is called the

IMS Manifest file, because course content and organization is described in the context of ‘manifests’. Once a Package has been incorporated into a single file for transportation, it is called a Package Interchange File. The relationship of these parts to the content container is described below:

(1) Package Interchange File – a single file, (e.g., ‘.zip’, ‘.jar’, ‘.cab’) which includes a top-level manifest file named “imsmanifest.xml” and all other files as identified by the Manifest. A Package Interchange File is a concise Web delivery format, a means of transporting related, structured information. PKZip v2.04g (.zip) is recommended as the default Package Interchange File format. Any ZIP file format MUST conform to RFC1951.

(2) Package – a logical directory, which includes a specially named XML file, any XML control documents it directly references (such as a DTD or XSD file), and contains the actual file resources. The file resources may be organized in sub-directories.

(2.1) Top-level Manifest – a mandatory XML element describing the Package itself. It may also contain optional sub-Manifests. Each instance of a manifest contains the following sections:

(a) Meta-data section – an XML element describing a manifest as a whole;

(b) Organizations section – an XML element describing zero, one, or multiple organizations of the content within a manifest;

(c) Resources section – an XML element containing references to all of the actual resources and media elements needed for a manifest, including meta-data describing the resources, and references to any external files;

(d) sub-Manifest – one or more optional, logically nested manifests;

(2.2) File Resources – these are the actual media elements, text files, graphics, and other resources as described by the manifest(s). The file resources may be organized in sub-directories.

(3) Package – A Package represents a unit of usable (and reusable) content. This may be part of a course that has instructional relevance outside of a course organization and can be delivered independently, as an entire course or as a collection of courses. Once a Package arrives at its destination to a run time service, such as an LMS vendor, the Package must allow itself to be aggregated or disaggregated into other Packages. A Package must be able to stand alone; that is, it must contain all the information needed to use the contents for learning when it has been unpacked.

Packages are not required to be incorporated into a Package Interchange File. A Package may also be distributed on a CD-ROM or other removable media without being compressed into a single file. An IMS Manifest file and any other supporting XML files directly referenced by it (DTD, XSD) must be at the root of the distribution medium.

(4) Manifest – A manifest is a description in XML of the resources comprising meaningful instruction. A

manifest may also contain zero or more static ways of organizing the instructional resources for presentation.

The scope of manifest is elastic. A manifest can describe part of a course that can stand by itself outside of the context of a course (an instructional object), an entire course, or a collection of courses. The decision is given to content developers to describe their content in the way they want it to be considered for aggregation or disaggregation. The general rule is that a Package always contains a single top-level manifest that may contain one or more sub-Manifests. The top-level manifest always describes the Package. Any nested sub-Manifests describe the content at the level to which the sub-Manifest is scoped, such as a course, instructional object, or other.

For example, if all content comprising a course is so tightly coupled that no part of it may be presented out of the course context, a content developer would want to create a single manifest to describe that course's resources and organization. However, content developers who create "instructional objects" that could be recombined with other instructional objects to create different course presentations would want to describe each instructional object in its own manifest, then aggregate those manifests into a higher-level manifest containing a course organization. Finally, a content developer who wants to move multiple courses in a single Package (a curriculum), would use a top-level manifest to contain each course-level manifest and any instructional object manifests that each course might contain.

(5) Resource – The resources described in the manifest are assets such as Web pages, media files, text files, assessment objects or other pieces of data in file form. Resources may also include assets that are outside the Package but available through a URL, or collections of resources described by sub-Manifests. The combination of resources is generally categorized as content. Each resource may be described in a <resource> element within a manifest's XML. This element includes a list of all the assets required to use the resource. The files included in the Package are listed as <file> elements within such <resource> elements.

While all content should be referenced in the resource section of the manifest, it is not necessary for all declared resources in a manifest to be referenced by <item> elements in the <organization> section of a manifest. This feature can be useful in two cases:

(a) When a file needs to be included that does not need to be presented to the learner;
(b) When the package is used as a content archive that is not designed to be presented to a learner.

4.2 Standard Name for the Manifest File

Content distributed according to the IMS Content Packaging specification must contain an IMS Manifest file. To ensure that the IMS Manifest file can always be found within a Package, it has a pre-defined name and location:

imsmanifest.xml

In the absence of this file, the package is not an IMS Package and cannot be processed. It is required that the name be kept, as above, in all lowercase letters.

The IMS Manifest file and any of its directly referenced XML control files (DTD, XSD) must be placed at the root of the Package Interchange File or any other packaging image (like a CD-ROM). XML control files that are indirectly referenced can be located as required by the namespace and path names. The usage of remote or local validation files is implementation dependent.

However, if local files are used then these must be identical to those online. If “local” validation is going to be performed using a local copy of the W3C xml.xsd and the validation process is going to be done in a “disconnected” environment, then “local” versions (i.e., copy of) of the following files will also be needed: *datatypes.dtd* and *XMLSchema.dtd*. These DTDs are used by the W3C provided *xml.xsd* and can be obtained from the W3C.

4.3 Extensibility

An important underpinning of the IMS Content Packaging specification is rich support for extensibility.

While the base Content Packaging Information Model leverages the rich set of meta-data elements defined in the IMS Meta-Data Specification v1.2.1 or the IEEE 1484.12.3 Standard for eXtensible Markup Language (XML) Schema Binding for Learning Object Metadata (see IMS Meta-Data v1.3 [MD, 04] for best practices and guidelines in implementing the IEEE LOM specification), it defines only the basic structures for organization and resources (Web Content).

It is expected that implementers of this specification will define new types of resources and organizations to describe and transport rich learning resources, and over time, it may be possible to incorporate widely used extensions into future versions of this specification.

4.4 Manifest Elements

This section provides a conceptual, informative description of the elements contained in a Manifest. Figure 4.2 illustrates the primary elements of a Manifest.

Figure 4.2 – Manifest elements.

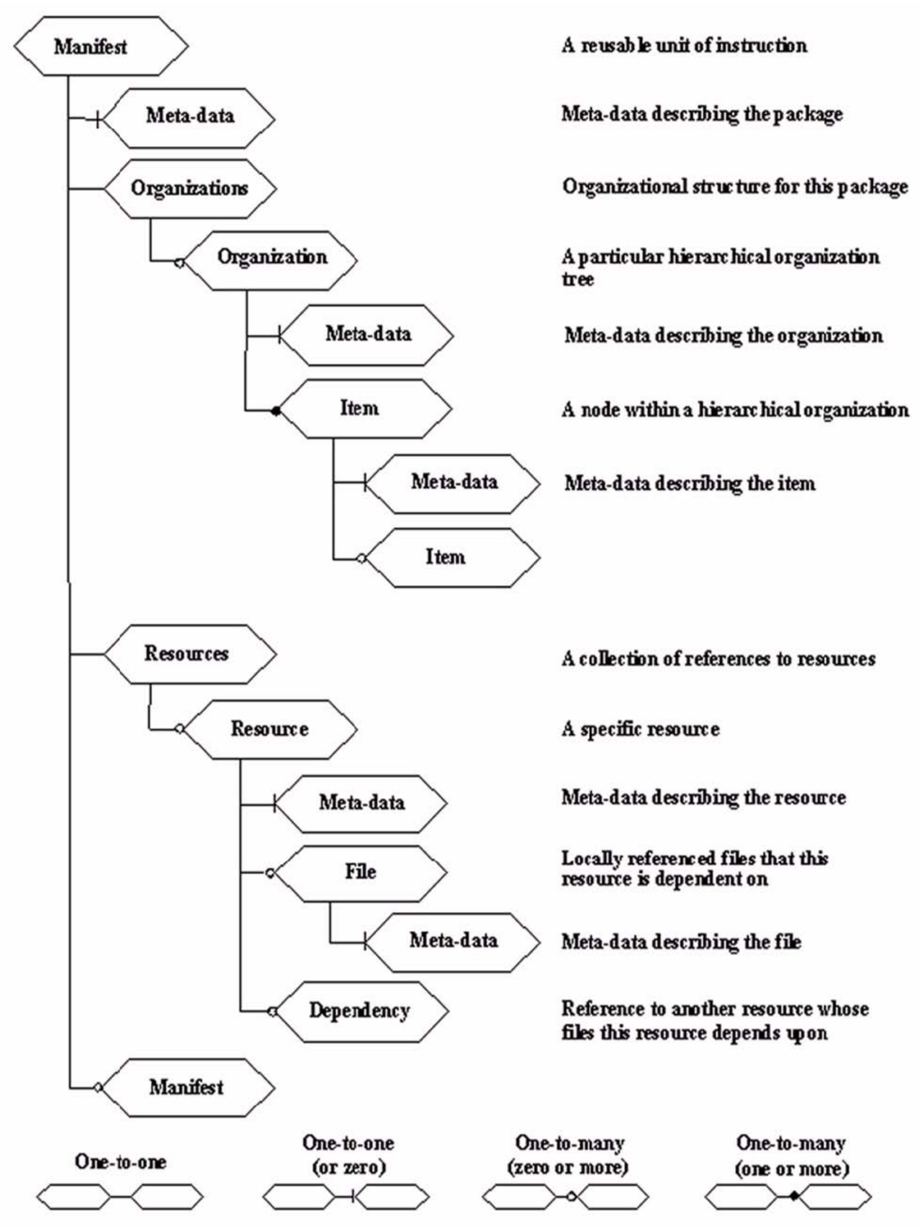

Manifest
A reusable unit of instruction
Meta-data
Meta-data describing the package
Organizations
Organizational structure for this package
Organization
A particular hierarchical organization tree
Meta-data
Meta-data describing the organization
Item
A node within a hierarchical organization
Meta-data
Meta-data describing the item
Item
Resources
A collection of references to resources
Resource
A specific resource
Meta-data
Meta-data describing the resource
File
Locally referenced files that this resource is dependent on
Meta-data
Meta-data describing the file
Dependency
Reference to another resource whose files this resource depends upon
Manifest
One-to-one
One-to-one (or zero)
One-to-many (zero or more)
One-to-many (one or more)

Table 4.1 provides a conceptual, informative description of the data objects. The columns used in the table refer to:

No:	The number of the data element. An element may be composed of sub-elements. The numbering scheme reflects these relationships.
Name:	The descriptive name of the element.
Explanation:	A brief functional description of the element.
Reqd:	Indicates if the element is required. M = mandatory element that must be included in the data object, if the element at the higher level is included. C = conditional element, existence is dependent on values of other elements. O = optional element.
Mult:	Multiplicity of the element. Repeatability of an Element implies that all sub-elements repeat with the Element. Blank (-) = single instance. Number = maximum number of times the element is repeatable. n = multiple occurrences allowed, no limit.
Type:	A description of formatting rules for the data element: Type includes the maximum length of the element. The international character set specified by ISO 10646 will be used for all fields. Container = 'tag' element, of fixed length. ID = element used to uniquely identify an object. IDRef = a reference to an ID. String (n) = descriptive element (smallest permitted maximum). Boolean = True | False.

Notes: Additional descriptive information about the element.

(1) In the table below, the Manifest elements contained in the Content Packaging Information Model are described using mixed case to enhance readability. Implementers of this specification should refer to particular binding specifications. For example, some XML bindings follow the W3C convention of using lowercase for all elements;

(2) Elements surrounded by braces ({}) indicate areas in the Information Model where elements from other information models or specifications are expected to be included;

(3) The order of elements described in Table 4.1 is not significant from the perspective of the information model. However, the corresponding XML binding imposes this implied order as a requirement for IMS manifests.

Table 4.1 - Content packaging data objects.

No	Name	Explanation	Reqd	Mult	Type	Note
1	Manifest	A reusable unit of instruction. Encapsulates meta-data, organizations, and resource references.	M	-	Container	
1.1	Identifier	An identifier that is unique within the Manifest.	M	-	ID	See the Best Practice Guide for guidelines on the use of identifiers.

1.2	Version	Identifies the version of this Manifest (e.g., 1.0).	O	-	String (20)	Used to identify if there have been any changes to the Package. Identifier is the same in two Manifest files.
1.3	Xml:base	Provides the relative path offset for the content file(s).	O	-	String (2000)	The maximum length of both 'href' and 'xml:base' is defined as 2000 octets. In cases where multiple 'xml:base' values need to be concatenated to create the full path then care must be taken to ensure that the total length does not exceed that of the 'href'. If the path length is greater than 2000 octets then the system behavior is undefined
1.4	Meta-data	Meta-data describing the Manifest.	O	-	Container	
1.4.1	Schema	Describes the schema that defines and controls the Manifest.	O	-	String (100)	If no schema element is present, it is assumed to be "IMS Content".
1.4.2	SchemaVersion	Describes the version of the above schema (e.g., 1,0, 1.1).	O	-	String (20)	If no version is present, it is assumed to be "1.1"
1.4.3	{Meta-Data}	This is where Meta-data is inserted using an appropriate information model.	O	n	-	By default, the information contained in this section is defined by the IMS Meta-Data specification.
1.5	Organizations	Describes one or more structures or organizations for this package.	M	-	Container	An organization can be selected by its meta-data.
1.5.1	Default	Indicates which organization scheme is the default one.	O	-	IDRef	When used, this must point to a direct child <organization> in the manifest i.e., it cannot point to an <organization> in a sub-manifest. If not supplied, the first organization element encountered is assumed to be the default.
1.5.2	Organization	Describes a particular hierarchical organization.	O	n	Container	Different views or organizational paths through the content can be described using multiple

						instances of organization.
1.5.2.1	Identifier	An identifier, for the organization, that is unique within the Manifest file.	M	-	ID	See the Best Practice Guide for guidelines on the use of identifiers.
1.5.2.2	Structure	Has a default value of “hierarchical” for describing the shape of an organization.	O	-	String (200)	Other values for structure will likely become part of a future specification.
1.5.2.3	Title	Describes the title of the organization.	O	-	String (200)	Used to help user decide which organization to choose.
1.5.2.4	Item	A node that describes the shape of the organization.	M	n	Containe r	Can be used in a hierarchical organizational scheme by ordering and nesting.
1.5.2.4. 1	Identifier	An identifier, for the Item, that is unique within the Manifest file.	M	-	ID	See the Best Practice Guide for guidelines on the use of identifiers.
1.5.2.4. 2	IdentifierRef	A reference to an identifier in the resources section or a sub-Manifest.	O	-	String (2000)	
1.5.2.4. 3	Title	Title of the item.	O	-	String (200)	
1.5.2.4. 4	IsVisible	Indicates whether or not this item is displayed when the structure of the Package is displayed or rendered.	O	-	Boolean	If not present, value is assumed to be “true”. This value only affects the item for which it is defined and not the children of the Item or a resource associated with an Item.

No	Name	Explanation	Reqd	Mult	Type	Note
1.5.2.4. 5	Paramete rs	Static parameters to be passed to the resource at launch time.	O	-	String (1000)	#<parameter> <name>=<value>(&<name>=<value>) *(#<parameter>) ?<name>=<value>(&<name>=<value>)*(#<parameter>) Where <parameter> is some implementation defined characterstring value Where <name> is some implementation defined name Where <value> is some implementation defined value The "=" is required to separate the <name> and <value> pair The "&" is required to separate multiple sets of <name> and <value> pairs The (&<name>=<value>)* indicates that the 0 or more <name> and <value> pairs can be concatenated together NOTE: The characters used in the

						parameters field may need to be URL encoded. RFC 2396 defines the rules and requirements for encoding URLs.
1.5.2.4.6	Item	A sub-node within this organization.	O	n	Container	This is a sub-item and repeats all the parts of <item>.
1.5.2.4.7	Meta-data	Meta-data describing this item.	O	-	Container	See item 1.4.3 above
1.5.2.4.7.1	{Meta-Data}	This is where Meta-data is inserted using an appropriate information model.	O	n	-	By default, the information contained in this section is defined by the IMS Meta-Data specification.
1.5.2.4.8	Meta-data	Meta-data describing this organization.	O	-	Container	See item 1.4.3 above
1.5.2.4.8.1	{Meta-Data}	This is where Meta-data is inserted using an appropriate information model.	O	n	-	By default, the information contained in this section is defined by the IMS Meta-Data specification.
1.6	Resources	A collection of references to resources. There is no assumption of order or hierarchy.	M	-	Container	
1.6.1	Xml:base	Provides the relative path offset for the content file(s).	O	-	String (2000)	
1.6.2	Resource	A reference to a resource.	O	n	Container	
1.6.2.1	Identifier	An identifier, of the resource, that is unique within the scope of its containing manifest file.	M	-	ID	See the Best Practice Guide for guidelines on the use of identifiers.

No	Name	Explanation	Reqd	Mult	Type	Note
1.6.2.2	Type	Indicates the type of resource.	M	-	String (1000)	The only current types are: - “webcontent”, defined as content that can be hosted in or launched by an Internet browser (this includes HTML-based content, content that requires plug-ins e.g., Flash, Real Media and executables that are launched by a browser); - The labels defined in Section 7 of the document 'Using IMS Content Packaging to Package Instances of LIP and Other IMS Specifications, Version 1.0 Implementation Handbook, August 2001'; Note that IMS specifications and versions of IMS specifications that had not yet been released at the time of writing of this document, should follow the syntax as specified in section 7.’ NOTE: An IMS specification may extend the table in section by using the syntax and including a normative statement to that effect in the specification; - “imsldcontent” when the content is used to support the IMS Learning Design specification; - “other”, which should be used when no other term is appropriate.
1.6.2.3	Href	A reference to a URL.	O	-	String (2000)	If the URL is also based upon the contents of the ‘parameter’ attribute of the <item> referencing the <resource> then the algorithm defined in Section 4.2 should be used to construct the full

						URL. The syntax for the 'Href' value must conform to RFC2396.
1.6.2.4	Xml:base	Provides the relative path offset for the content file(s).	O	-	String (2000)	See item 1.3 above.
1.6.2.5	Meta-data	Meta-data describing this resource.	O	-	Container	See item 1.4.3 above
1.6.2.5.1	{Meta-Data}	This is where Meta-data is inserted using an appropriate information model.	O	n	-	By default, the information contained in this section is defined by the IMS Meta-Data specification.
1.6.2.6	File	A listing of files that this resource is dependent on.	O	-	Container	An element identifying a single file this resource is dependent on. Repeat as needed for each file for a given resource.
1.6.2.6.1	Href	Identifies the location of the file.	M	n	String (2000)	The syntax for the 'Href' value must conform to RFC2396.
1.6.2.6.2	Meta-data	Meta-data describing this file.	O	-	Container	See item 1.4.3 above.
1.6.2.6.2.1	{Meta-Data}	This is where Meta-data is inserted using an appropriate information model.	O	n	-	By default, the information contained in this section is defined by the IMS Meta-Data specification.
1.6.2.7	Dependency	Identifies a resource whose files this resource depends upon.	O	n	IDref	This element identifies a single resource which can act as a container for multiple files that this resource depends upon.
1.6.2.7.1	IdentifierRef	A reference to an identifier in the resources section.	M	-	String (2000)	This indentifierref cannot reference an identifier in a sub-manifest.
1.7	Manifest	A reusable unit of instruction. Encapsulates meta-data, organizations, and resource references.	O	n	Container	See section 4.1 sub-Manifests below for more information.

4.4.1 sub-Manifests

When the identifierref of an <item> in an <organization> references a sub-Manifest rather than another type of resource, this shall be interpreted as follows:

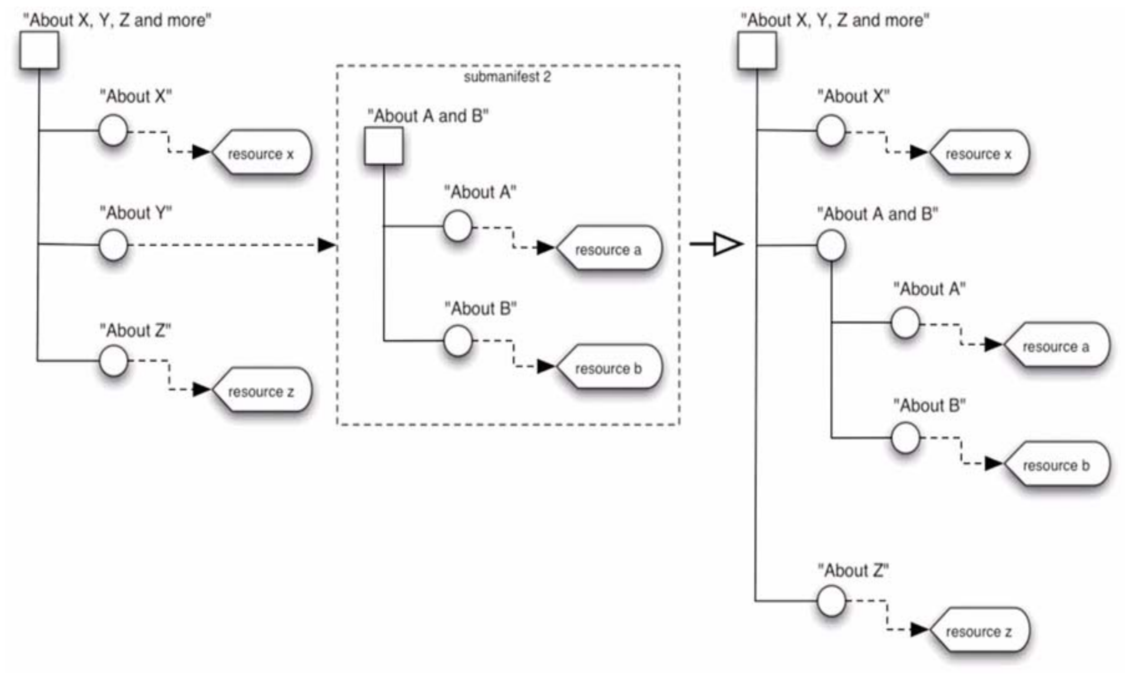
If the sub-Manifest does not include any <organization>, the reference cannot be resolved. This shall be treated as a null identifierref;

If the sub-Manifest includes an <organization>, the root node of that organization (i.e., the

<organization> element itself) shall merge with the <item> that references the sub-Manifest. If the same attribute is specified for both the <item> and the <organization> it references, but with different values, the value defined for the referred <organization> manifest shall override the value defined for the referring <item>. That is, child attributes take precedence over parent attributes. This behavior is expected of the rendering of the navigation tree, but does not need to affect the XML of the manifest(s).

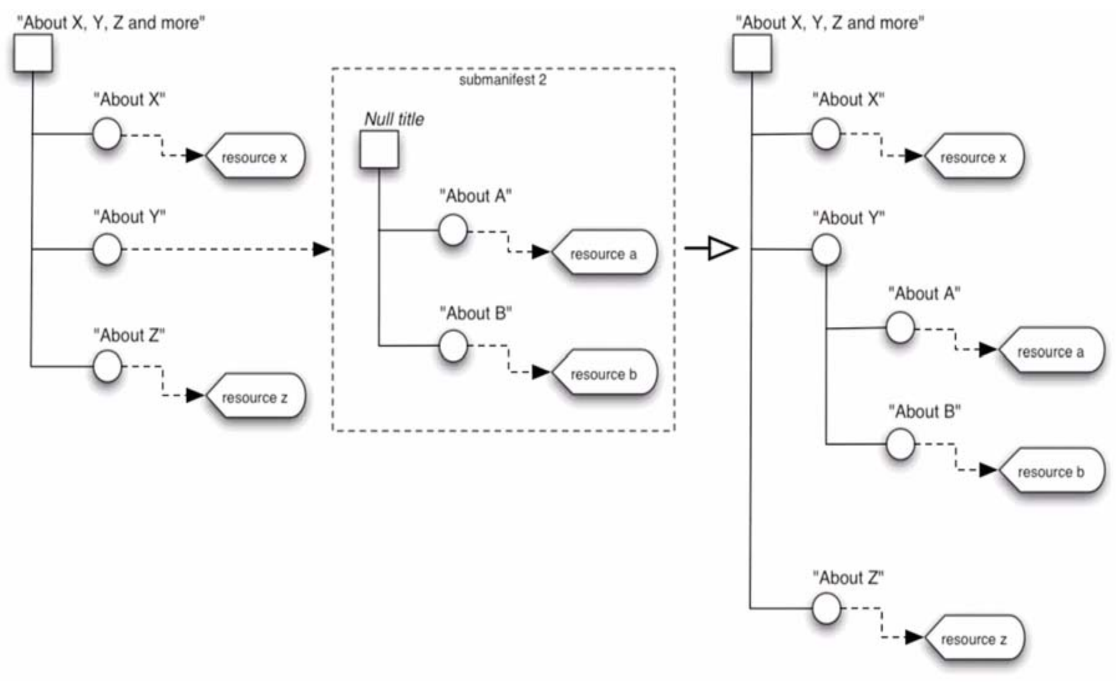
The diagram in Figure 4.2 explains how the content of a sub-Manifest is merged with the content of a referencing manifest in the rendering of a navigation tree. The circles represent items in an organization structure. In this example, the organization of the sub-manifest does not have a title attribute.

Figure 4.2 - Merging <organization> from a sub-Manifest.



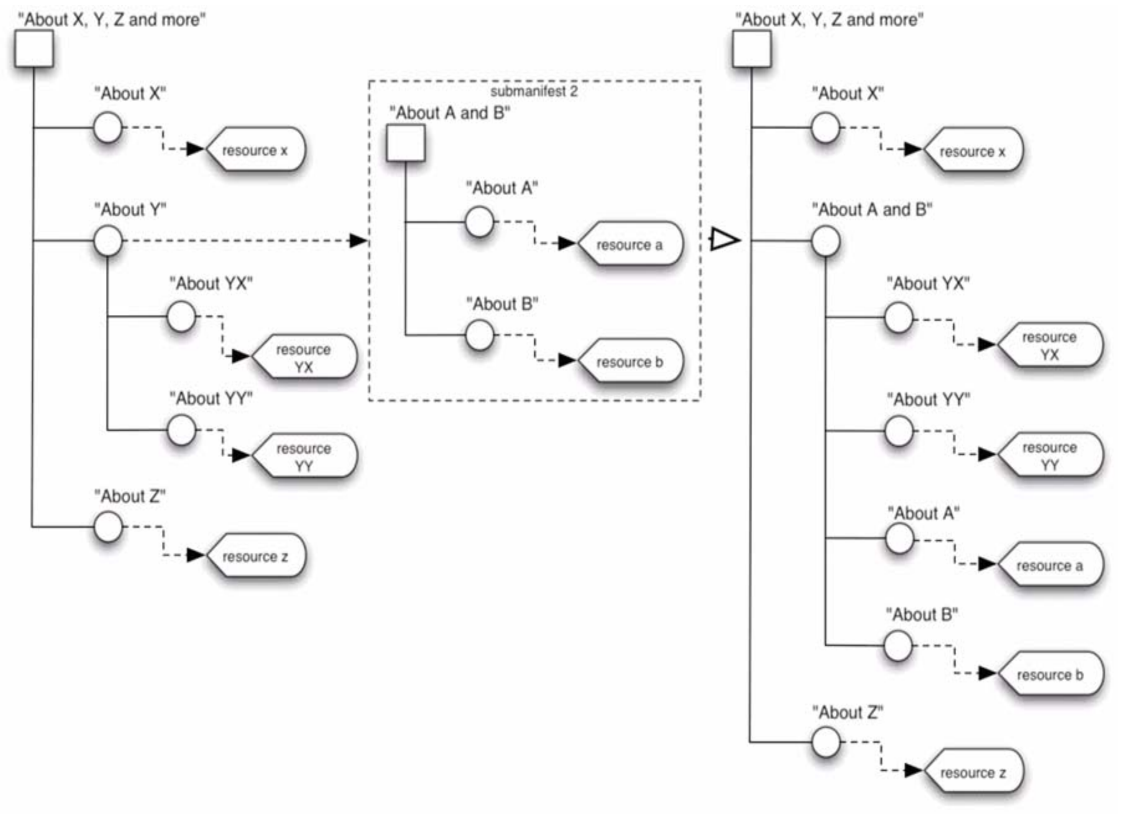
The diagram in Figure 4.3 shows an example where the <organization> of the sub-Manifest does have a title attribute.

Figure 4.3 Merging an attributeless <organization> from a sub-Manifest.



In Figure 4.4, the diagram explains how the content of a sub-Manifest is merged with an item that has children of its own, and how the referencing item's children take precedence over those they merge with. In this example, the organization of the sub-Manifest does not have a title attribute.

Figure 4.4 Merging <organization> from a sub-Manifest when the referencing item has children.



4.4.2 Href URL Construction Algorithm

In the cases where full URL referenced in the ‘Href’ value is also based upon the parameters passed in the ‘parameter’ attribute of the <item> referencing the <resource> then the following algorithm is to be used to construct the full URL:

```
 While    first char of parameters is in"?&"
        Clear first char of parameters
 If    first char of parameters is "#"
    If the URI contains "#"
         Discard parameters
    Else
        Append parameters to the URI
     Done processing URI
If the URI contains "?"
     Append "&" to the URI
Else
     Append "?" to the URI
Append parameters to the URIDone processing the URI
```

The definition of this algorithm is normative.

5. IMS Content Packaging XML Binding

5.1 XML Basics

The Content Packaging data model can be defined as a hierarchy. Hierarchical models are convenient for representing data consisting of many elements and sub-elements. XML is perfectly suited for representing hierarchical models. An XML document is a hierarchy comprised of elements that have contents and attributes.

5.1.1 Elements

An element is a component of a document that has been identified in a way a computer can understand. Each element has a tag name. When a tag name is shown as “<TAGNAME>”, with less-than and greater-than symbols before and after the tag name, it serves as the start-tag to mark the beginning of an element. When that same tag name has a forward slash “/” added, it serves as an end-tag such as “</TAGNAME>”. An element may have contents between its start and end-tags, and may have one or more attributes. When an XML element has a start and end-tag (also called an opening and closing tag) with a common name, it is considered to be “well-formed” XML. The contents of an element are placed between the start and end-tags as shown below:
<TAGNAME>contents</TAGNAME>

(1) Element Contents

An element may contain other elements, Parsed Character Data (PCDATA), Character Data (CDATA), or a mixture of PCDATA and elements. The allowable contents of an element are its content model. PCDATA really means any character string that does not contain elements. PCDATA is what the bulk of elements will use between their start and end-tags. CDATA is different in that it is a method for adding any character data that should not be processed. For example, you could add some JavaScript code instructions using a CDATA section. A CDATA section tells the parser not to look for any mark-up until after it locates the end of the CDATA section.

(2) Element Attributes

An attribute provides additional information about an element. Attributes are a way of attaching characteristics or properties to the elements of a document. An element may have more than one attribute. Attributes are contained within the start tag of an element. Attributes are represented by an attribute name followed by an equal sign and the attribute value in quotation marks:

```
<timeframe>
   <begin restrict="1"> 1999-07-23 </begin>
</timeframe>
```

In this example, the <timeframe> element contains another element: the <begin> element. The <begin> element has one attribute ‘restrict’, with the value “1”. The value for the element <begin> is “1999-07-23”. These two elements then make up a timeframe begin date.

(3) Element Names

Each element has a unique name, referred to as the tag name. XML is case-sensitive in its processing of tag names.

The IMS Content Packaging XML Binding adheres to the following tag name rules:

(a) All tag names will conform to the rules for element naming as given within the XML 1.0 Specification.

(b) Names beginning in XML in any case or mix of cases are not permitted.

(c) The IMS binding will use only lowercase tag and element names.

(d) Element names may not include words reserved by the XML specification. These include:

DOCTYPE

ELEMENT

ATTLIST

ENTITY

(e) Tag names defined by the IMS binding may not be redefined.

5.1.2 Document Type Definitions

The tag name, content model, and attributes of elements are defined in a Document Type Definition (DTD) statement. This may exist as an external file or a block of text internal to an XML document. Internal DTDs are used to override elements defined in external DTD files, so an internal DTD should be used with care. The DTD defines the elements that may be used and may define the contents of the elements.

This specification defines a DTD (imscp_rootv1p1.dtd) as a non-normative reference. Some XML editors may make use of a DTD to help guide the developer in creating the proper elements at the proper locations in an XML file. Other developers will make use of DTDs to validate their XML documents to ensure their document is consistent with all of the element names and locations defined in the DTD. Details of the construction of DTDs are outside the scope of this document, but links to the XML 1.0 Specification are included in section 1.5 of this document.

5.1.3 XML Schemas

A schema is a formal specification of element names that indicates which elements are allowed in an XML document, and in which combinations. New schema languages, such as those defined in the XML-Schemas Working Group, provide the same baseline functionality as a DTD. However, because these schema languages are extensible, developers can augment them with additional information, such as data types, inheritance, and presentation rules. This makes these new schema languages far more powerful than DTDs. For more information about XML schemas, there is a link to the W3C XML Schema Recommendation in section 1.5.

This specification defines a W3C XML Schema as a non-normative reference. Some XML editors may make use of schemas to help guide the developer in creating the proper elements at the proper locations in an XML file. Other developers will make use of schemas to validate their XML documents and/or to define extensions to the IMS Content Packaging Binding. Details of the construction of schemas are outside the scope of this document.

5.1.4 Valid Character Sets

A Content Packaging record must use UTF-8 or UTF-16 encoding of the character sets as defined in ISO 10646. See the XML Version 1.0 for more details on the specification of well-formed XML.

5.1.5 Special Handling Requirements

(1) XML Reserved Characters

Some characters used in XML must be escaped when used outside of their XML-defined usage as found in section 2.4 of the XML 1.0 Specification. These characters are ampersand (&), less than (<), greater than (>), apostrophe (‘), and the double-quote character (“).

These characters may be represented using either numeric character references or the strings “&”, “<”, “>”, “&apos;”, and “””.

Below is a more complete quote from the W3C XML standards:

Quote from Extensible Markup Language (XML) 1.0
W3C Recommendation 10-February-1998
2.4 Character Data and Markup

“Text consists of intermingled character data and markup. Markup takes the form of start-tags, end-tags,empty-element tags, entity references, character references, comments, CDATA section delimiters, document type declarations, and processing instructions.

“All text that is not markup constitutes the character data of the document.
“The ampersand character (&) and the left angle bracket (<) may appear in their literal form only when used as markup delimiters, or within a comment, a processing instruction, or a CDATA section. They are also legal within the literal entity value of an internal entity declaration; see "4.3.2 Well-Formed Parsed Entities". If they are needed elsewhere, they must be escaped using either numeric character references or the strings "&" and "<" respectively. The right angle bracket (>) may be represented using the string ">", and must, for compatibility, be escaped using ">" or a character reference when it appears in the string "]]>" in content, when that string is not marking the end of a CDATA section.

“In the content of elements, character data is any string of characters, which does not contain the start-delimiter of any markup. In a CDATA section, character data is any string of characters not including the CDATA-section-close delimiter, "]]>".
“To allow attribute values to contain both single and double quotes, the

apostrophe or single-quote character (') may be represented as "&apos;", and the double-quote character (") as """."

(2) White Space Handling

Questions often arise as to whether Web-based data transmission tools might inadvertently strip off or transform some of the white space characters embedded in data transmitted between systems using XML. To eliminate concern about this issue, refer to the following quote from the W3C XML standards, which indicate that all white space must be preserved where it is part of the data.

Quote from Extensible Markup Language (XML) 1.0
W3C Recommendation 10-February-1998
2.10 White Space Handling

"In editing XML documents, it is often convenient to use "white space" (spaces, tabs, and blank lines, denoted by the non-terminal S in this specification) to set apart the mark-up for greater readability. Such white space is typically not intended for inclusion in the delivered version of the document. On the other hand, "significant" white space that should be preserved in the delivered version is common, for example in poetry and source code.

"An XML processor must always pass all characters in a document that are not mark-up through to the application. A validating XML processor must also inform the application which of these characters constitute white space appearing in element content.

"A special attribute named xml:space may be attached to an element to signal an intention that in that element, white space should be preserved by applications. In valid documents, this attribute, like any other, must be declared if it is used. When declared, it must be given as an enumerated type whose only possible values are
"<!ATTLIST poem xml:space (default|preserve) 'preserve'>
"The value "default" signals that applications' default white-space processing modes are acceptable for this element; the value "preserve" indicates the intent that applications preserve all the white space. This declared intent is considered to apply to all elements within the content of the element where it is specified, unless overridden with another instance of the xml:space attribute."

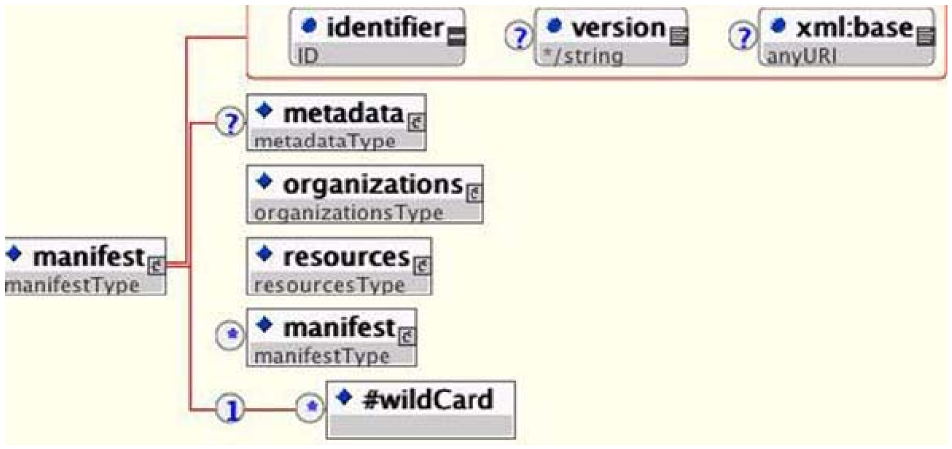
5.3.Narrative Description of XML Binding

This specification defines the XML format using narrative. XML DTDs and XML Schemas that implement this *abstract* format are referenced as non-normative parts of this specification.

5.3.1 <manifest> Elements

Description. The first, top-level <manifest> element in the Manifest encloses all the reference data. Subsequent occurrences of the <manifest> elements inside the top-level <manifest> are used to compartmentalize files, meta-data, and organization structure for aggregation, disaggregation, and reuse. The best-practice use of the IMS Content
Packaging specification will result in each “learning object” or “atomic unit of learning” being placed within its own <manifest> element.

Figure 5.1 <manifest> elements.



Multiplicity.

The top-level <manifest> occurs once and only once within the IMS Manifest file.

Attributes.

(a) identifier (required). An identifier, provided by an author or authoring tool, that is unique within the Manifest.
 Data type = string;

(b) version (optional). Identifies the version of the Manifest. Is used to distinguish between manifests with the same identifier.
 Data type = string;

(c) xml:base (optional). This provides a relative path offset for the content file(s). The usage of this element is defined in the XML Base Working Draft from the W3C. Data type = string.

Elements.

(a) <metadata>

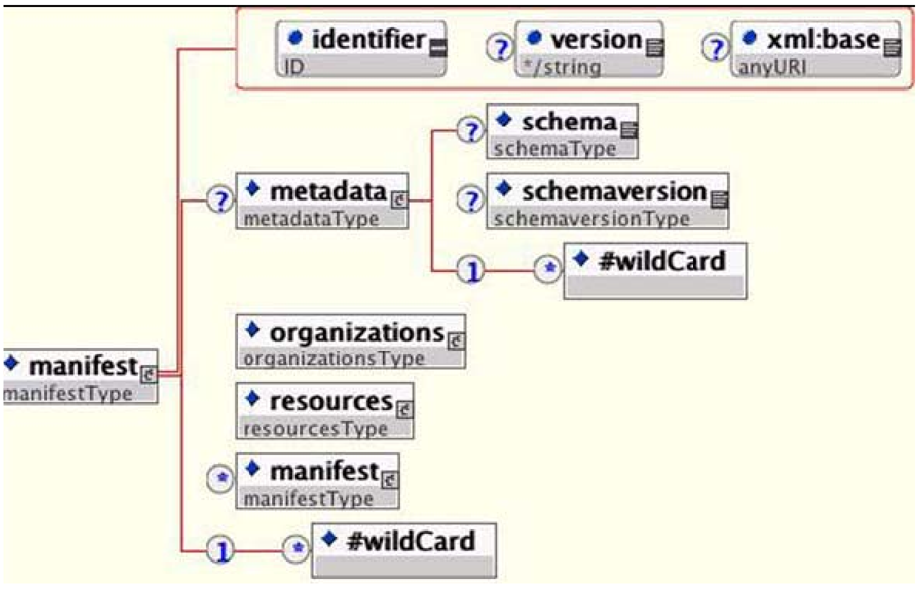
(b) <organizations>

(c) <resources>

(d) <manifest>

5.3.1.1 <metadata>

Figure 5.2 <metadata> elements.



Multiplicity.

Occurs zero or once within a <manifest> element.

Elements.

(a) <schema>

(b) <schemaversion>

(c) Meta-Data: Implementers are free to choose from any of the meta-data elements defined in the IMS Meta-Data specification or other meta-data standards.

Example.

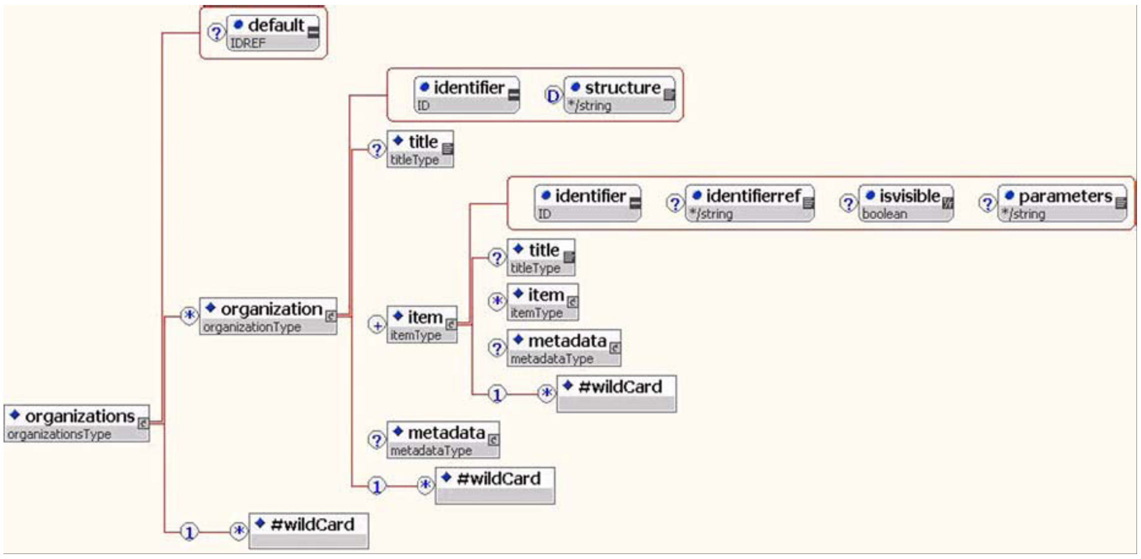
```
<metadata>
  <schema>IMS Content</schema>
  <schemaversion>1.1</schemaversion>
  <imsmd:lom>
    <imsmd:general>
      <imsmd:title>
        <imsmd:langstring xml:lang="en-US">Simple
      Manifest</imsmd:langstring>
      </imsmd:title>
    </imsmd:general>
  </imsmd:lom>
</metadata>
```

5.3.1.2 <organizations>

Description.

Describes zero, one, or more structures or organizations (i.e., <organization> elements) for this package.

Figure 5.3 <organizations> elements.



Multiplicity.

Occurs once within a <manifest> element.

Attributes.

(a) default (optional). Identifies the default organization to use. Data type = idref.

Elements.

(a) <organization>

Example.

```
<organizations default="TOC1">
  <organization identifier="TOC1" structure="hierarchical">
    <title>default</title>
    <item identifier="ITEM1" identifierref="RESOURCE1" isvisible="true">
      <title>Lesson 1</title>
      <item identifier="ITEM2" identifierref="RESOURCE2" isvisible="true">
        <title>Introduction 1</title>
      </item>
      <item identifier="ITEM3" identifierref="RESOURCE3" isvisible="true">
        <title>Content 1</title>
      </item>
      <item identifier="ITEM4" identifierref="RESOURCE4" isvisible="true">
        <title>Summary 1</title>
      </item>
    </item>
    <item identifier="ITEM5" identifierref="RESOURCE5" isvisible="false">
      <title>Lesson 2</title>
      <item identifier="ITEM6" identifierref="RESOURCE6" isvisible="false">
        <title>Introduction 2</title>
      </item>
      <item identifier="ITEM7" identifierref="RESOURCE7" isvisible="false">
        <title>Content 2</title>
      </item>
      <item identifier="ITEM8" identifierref="RESOURCE8" isvisible="false">
```

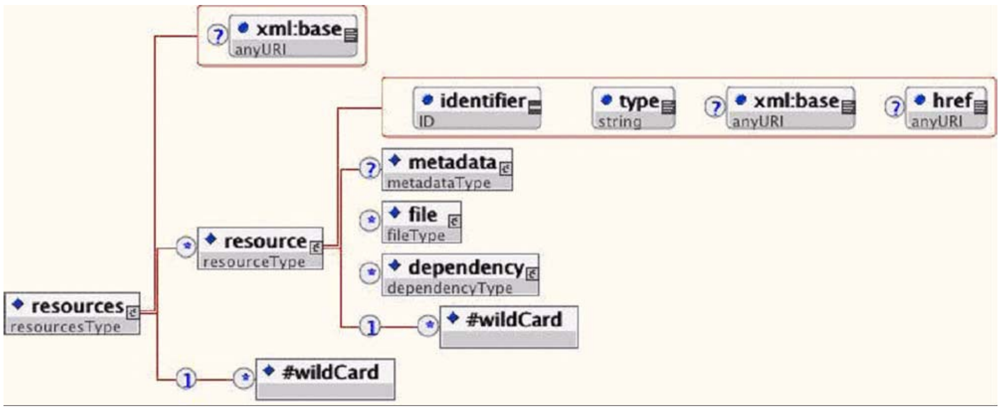
```
        <title>Summary 2</title>
      </item>
    </item>
    <item identifier="ITEM9" identifierref="RESOURCE9" isvisible="true">
        <title>Lesson 3</title>
      <item identifier="ITEM10" identifierref="RESOURCE10" isvisible="true"
      parameters="foo">
        <title>Introduction 3</title>
      </item>
      <item identifier="ITEM11" identifierref="RESOURCE11" isvisible="true">
        <title>Content 3</title>
      </item>
      <item identifier="ITEM12" identifierref="RESOURCE12" isvisible="true">
        <title>Summary 3</title>
      </item>
    </item>
  </organization>
</organizations>
```

5.3.1.3 <resources>

Description.

This element identifies a collection of content files.

Figure 5.4 - <resources> elements.



Multiplicity.

Occurs once and only once within a <manifest> element.

Attributes.

(a) xml:base (optional). This provides a relative path offset for the content file(s). The usage of this element is defined in the XML Base Working Draft from the W3C. Data type = string.

Elements.

(a) <resource>

Example.

```
<resources>
  <resource identifier="RESOURCE1" type="webcontent" href="lesson1.htm">
        <file href="lesson1.htm"/>
  </resource>
  <resource identifier="RESOURCE2" type="webcontent" href="intro1.htm">
        <file href="intro1.htm"/>
  </resource>
  <resource identifier="RESOURCE3" type="webcontent" href="content1.htm">
        <file href="content1.htm"/>
  </resource>
  <resource identifier="RESOURCE4" type="webcontent" href="summary1.htm">
        <file href="summary1.htm"/>
  </resource>
</resources>
```

5.3.2 <metadata> Elements

5.3.2.1 <schema>

Description.

Describes the schema used (e.g., IMS Content). If no schema element is present, it is assumed to be “IMS Content”. Data type = string.

Multiplicity.

Occurs zero or once within <metadata>.

Example.

```
<schema>IMS Content</schema>
```

5.3.2.2 <schemaversion>

Description.

Describes version of the above schema (e.g., 1,0, 1.1). If no version is present, it is assumed to be “1.1”. Data type = string.

Multiplicity.

Occurs zero or once within <metadata>.

Example.

```
<schemaversion>1.1</schemaversion>
```

5.3.2.3 Meta-data

Description.

See the IMS Meta-Data specification for more detail on the meta-data that are available for describing and cataloguing content packages. The IMS Meta-Data v1.2.1 is the default specification but other specifications/standards are permitted.

Multiplicity.

Defined in the relevant meta-data specification.

Example: In-line meta-data

```
<metadata>
  <imsmd:lom>
    <imsmd:general>
      <imsmd:title>
        <imsmd:langstring xml:lang="en-US">Simple
      Manifest</imsmd:langstring>
      </imsmd:title>
    </imsmd:general>
  </imsmd:lom>
</metadata>
```

5.3.3 <organizations> Elements

5.3.3.1 <organization>

Description.

This element describes a particular, passive organization of the material.

Multiplicity.

Occurs zero or more times within <organizations>.

Attributes.

(a) identifier (required). An identifier, provided by an author or authoring tool, that is unique within the Manifest.
Data type = id.

(b) structure (optional). Assumes a default value of “hierarchical”, such as is common with a tree view or structural representation of data.
Data type = string.

Elements.

(a) <title>

(b) <item>

(c) <metadata>

Example.

```
<organization identifier="TOC1">
  <title>default</title>
  <item identifier="ITEM1" identifierref="RESOURCE1" isvisible="true">
        <title>Lesson 1</title>
  </item>
  <item identifier="ITEM2" identifierref="RESOURCE2" isvisible="true">
        <title>Introduction 1</title>
  </item>
</organization>
```

5.3.3.2 <title>

Description.

This element describes the title of an <item>.

Multiplicity.

Occurs zero or more times within <item>.

5.3.3.3 <item>

Description.

This element describes a node within a structure.

Multiplicity.

Occurs one or more times within <organization> and zero or more times within <item>.

Attributes.

(a) identifier (required). An identifier that is unique within the Manifest. Data type = id.
(b) identifierref (optional). A reference to a <resource> identifier (within the same package) or a sub-Manifest that is used to resolve the ultimate location of the file. If no identifierref is supplied, it is assumed that there is no content associated with this entry in the organization. Data type = string.
(c) isvisible (optional). Indicates whether or not this resource is displayed when the unit of instruction is rendered. If not present, value is assumed to be ‘true’. Data type = boolean.
(d) parameters (optional). Static parameters to be passed to the content file at launch time. Data type = string.

Elements.

(a) <title>
(b) <item>
(c) <metadata>

Example.

```
<item identifier="ITEM3" identifierref="RESOURCE3" isvisible="true">
        <title>Content 1</title>
</item>
```

5.3.3.4 <metadata>

Description.

This element contains meta-data that describes the resource. Implementers are free to choose from any of the meta-data elements defined in the IMS Meta-Data specification or to define their own meta-data schema.

Multiplicity.

Occurs zero or once within <organization>.

Example.

```
<organization identifier="TOC1">
  <metadata>
  <!-- schema and schemaversion not given as they apply at manifest level -->
    <imsmd:lom>
      <imsmd:educational>
        <imsmd:interactivitylevel>1</imsmd:interactivitylevel>
      </imsmd:educational>
    </imsmd:lom>
  </metadata>
</organization>
```

5.3.4 <resources> Elements

A collection of references to resources. There is no assumption of order or hierarchy. Resources can be described in-line or externally.

5.3.4.1 <resource>

Description

This element describes a specific content file.

Multiplicity.

Occurs zero or more times within <resources>.

Attributes.

(a) identifier (required). An identifier, provided by the author or authoring tool, that is unique within the Manifest.

(b) ‘type’ (required). A string that identifies the type of resource. This specification defines the type “webcontent” plus reserved terms that are used to denote the packaging of content defined by other IMS specifications, including Learning Design. These labels are defined in Section 7 of the Implementation Handbook titled ‘Using IMS Content Packaging to Package Instances of LIP and Other IMS Specifications’ [IMSBUND, 01]. An IMS specification may extend the table in section 7 by using the syntax and including a normative statement to that effect in the specification.

(c) xml:base (optional). This provides a relative path offset for the content file(s). The usage of this element is defined in the XML Base Working Draft from the W3C. Data type = string.

(d) href (optional). A reference to the “entry point” of this resource. External fully-qualified URIs are also permitted.

Elements.

(a) <metadata>

(b) <file>

(c) <dependency>

Example: In-line resource

```
<resource identifier="RESOURCE2" type="webcontent" href="topics/index.htm">
  <file href="topics/index.htm"/>
  <file href="images/pic1.gif"/>
  <file href="images/pic2.gif"/>
</resource>
```

5.3.4.1.1 <metadata>

Description

This element contains meta-data that describes the resource. Implementers are free to choose from any of the meta-data elements defined in the IEEE 1484.12.1-2002 Standard for Learning Object Metadata (see IMS Meta-Data v1.3 [MD, 04] for best practices and guidelines in implementing the IEEE LOM specification). or to define their own meta-data schema.

Multiplicity.

Occurs zero or once within <resource>.

5.3.4.1.2 <file>

Description

Identifies one or more local files that this resource is dependent on. This includes the resource being referenced in the href attribute of <resource>. If the resource references an absolute URL (using href), <file> element(s) are not required.

Multiplicity.

Occurs zero or more times within <resource>.

Attributes.

(a) href (required). URL of the file.

Element

(a) <metadata>

Example.

```
<file href="topics/index.htm"/>
```

5.3.4.1.3 <dependency>

Description

This element identifies a single resource that can act as a container for multiple files that this resource depends upon.

Multiplicity.

Occurs zero or more times within <resource>.

Attributes.

(a) identifierref (required). An identifier for other resources to reference.

Example.

```
<resources>
  <resource identifier="R_A2" type="webcontent" href="sco1.html">
      <metadata/>
      <file href="sco1.html"/>
      <dependency indentiferref="R_A5"/>
  </resource>
  <resource identifier="R_A5" type="webcontent"
  href="pics/distress_sigs_add.jpg">
      <metadata/>
      <file href="pics/distress_sigs_add.jpg"/>
  </resource>
</resources>
```

5.3.5 Extensibility

The IMS Content Packaging XML Binding is extensible through the use of XML Namespaces and XML Schemas. It is expected that the extensibility mechanism will be used to describe additional types of meta-data, organizations, and resources. More information and examples of extensibility are contained in the IMS Content Packaging Best Practice Guide [CP, 04c].

5.4.Samples

A number of supporting files accompany the IMS Content Packaging specification documents and are available in the download .zip file (imscp_v1p1p4.zip), see Appendix A.

The W3C ‘xml:’ namespace definition file is available online but in cases where the parser does not have internet access, users may modify the schema to reference a local version of ‘xml.xsd’ that has been retrieved from W3C and associated with the same namespace. In these cases the local version is a control document and as such it should be in the root directory along with the manifest file itself.

The filename of the Content Packaging control XSD only identifies the primary and secondary versions of the version i.e., for the v1.1.4 release the XSD filename is ‘ims_cpv1p1.xsd’. The full version number is recorded in the ‘version’ attribute on the schema declaration e.g.,:

```
  <xsd:schema targetNamespace="http://www.imsglobal.org/xsd/imscp_v1p1"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://www.imsglobal.org/xsd/imscp_v1p1"
elementFormDefault="qualified" version="IMS CP 1.1.4">
```

6. IMS Content Packaging Best Practice and Implementation Guide

6.1 introduction

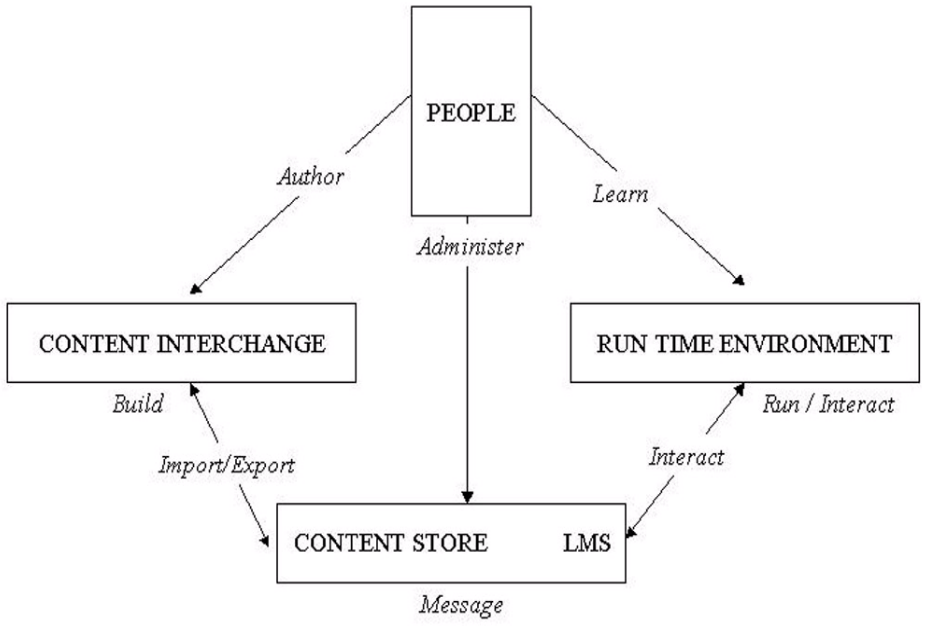
Instructional content often needs to be collected and packaged in some electronic form to enable efficient aggregation, distribution, management, and deployment. Producers of instructional materials want to have tools and technologies available to assist them in creating content. Software vendors in the online learning market want to create tools that enable efficient distribution and management of those instructional materials that have been created. Finally, learners are interested in high-quality learning experiences made possible by good deployment and delivery tools.

Content that is packaged in a known manner and file format, and with sufficient supporting information, can better satisfy the needs of the online learning community. This growing community needs guidelines and specifications for online learning content that will allow:

(a) Authors to build online learning content.
(b) Administrators to manage and distribute content.
(c) Learners to interact with and learn from the content.

A framework has been created with these goals in mind (Figure 6.1). The purpose of the IMS Content framework is to enable the encapsulation, in a concise and easily browsed manner, of all the required content resources, supporting information, and structure required to promote interoperable, online learning experiences.

Figure 6.1 IMS Content framework goals



6.2 <MANIFEST> Element

The organization of file resources within a Package is independent of their use. The <manifest> element in an IMS Manifest file serves the purpose of organizing the content for presentation in one or more presentation structures or views and of specifying the resource(s) supporting each view. In this way, a <manifest> element relieves the Package’s internal file structure from having to reflect the organization of resources for aggregation or disaggregation. Each resource or set of resources supporting a given presentation view is described for that view, including the path to each file through any internal folders or sub-directories comprising the internal file structure. A Manifest may provide one or more static views of the content.

A single <manifest> element is required as the root element of the IMS Manifest file. There can be one and only one top-level <manifest> element. All other instances of a <manifest> element are nested within the <manifest> element after the <resources> element. The information model does not impose a particular ordering within the <manifest> element however the corresponding XML binding does impose the implied order of: <metadata>, <organizations>, <resources> and <manifest>.A manifest contains four sub-elements: <metadata>, <organizations>, and <resources>, and any further <manifest> elements.

(1) <metadata> – (optional) this meta-data describes the manifest that contains it. Commonly used meta-data would include elements like title, description, keywords, a contributor’s role, a content’s purpose (e.g., educational objective, skill level), and copyright information. Meta-data elements should be drawn first from the IEEE 1484.12.1-2002 Standard for Learning Object Metadata (see IMS Meta-Data v1.3 [MD, 04] for best practices and guidelines in implementing the IEEE LOM

specification). Any meta-data elements not found in the IEEE 1484.12.1-2002 Standard for Learning Object Metadata could then be included via an XML namespace in a manifest’s meta-data element(s). All meta-data elements must be defined in a DTD or XSD, which are declared at the top of the IMS Manifest file. Directly referenced meta-data DTD or XSD files can be included with imsmanifest.xml at the root of a Package’s internal file structure.

(2) <organizations> – (required) contains zero, one, or multiple descriptions of the static organization of the content so that resources within the Package can be moved to create one or multiple organizations of content (such as course outlines). It is left to the discretion of content producers to decide whether to describe or not describe the organization of a course’s resources. If content producers choose to provide one or more descriptions of a course’s organization, they must also specify one as the default. The current Content Packaging XSD requires a single <organizations> element as a child of the <manifest> element. If content producers do not need an organizations section in the manifest, then it must appear as an empty element (i.e., “<organizations/>”) to satisfy the control rules expressed in the controlling documents (DTD, XSD). Also, only one <organizations> element is allowed within each <manifest> element. The current specification defines an <organization> sub-element as one that uses a hierarchical organization; however, other ways of describing the organizational structure or content (such as conditional/programmatic) are permitted.

(3) <resources> – (required) includes references to all of the resources included in the package. At a minimum it should reference all those files that are needed in order to view the content as specified in the <organizations> element. References may either be made internally or externally of a Package to both relative and absolute identifiers. For example, a reference to an external URL is permitted without having to include that resource as part of the Package Interchange File. Resources may also contain a <metadata> element for each content item referenced. Only one <resources> element is allowed within the top-level <manifest> element.

(4) <manifest> – (optional) specifies zero or more sub-Manifests. Nested <manifest> elements specify how content may be reliably aggregated or disaggregated into other Packages.
The following sections describe these more fully.

6.3 <METADATA> Element

Meta-data is optional and is allowed in various places in the manifest to more fully describe the contents of a Package. Search engines may look into the meta-data to find appropriate content for a learner or for content repackaging. Copyright and other intellectual property rights are easily declared within the meta-data. Authoring or editing tools could then read the rights stipulated by a content vendor to see if they have permission to open a resource file or files and change the contents.

The IMS CP Information model defines five places where meta-data can be used to describe different components of a content package:

(1) Manifest
(2) Organization
(3) Item
(4) Resource
(5) File

If there are requirements to describe any or all of these components with meta-data, then each of these respective components shall be described with separate instances of Meta-data. This construct allows a fine-grained description of each component of a package.

Beware, however, that there is no assumption of inheritance from one logical node to another. Each component, if desired, is represented by its own meta-data instance. If the meta-data associated with a resource X, for example, identifies Jane Smith as the author, it does not follow that file Y, a child node of resource X without meta-data, is also authored by Jane Smith. In this case, if Jane needs to be identified as the author of file Y, a separate meta-data instance needs to be associated with file Y.

The complete set of meta-data elements available for describing and cataloging a content Package is not included withthis specification. This specification recommends the best practice of using the IEEE 1484.12.1-2002 Standard for Learning Object Metadata (see IMS Meta-Data v1.3 [MD, 04] for best practices and guidelines in implementing the IEEE LOM specification), which contains approximately 86 individual meta-data elements that may be used to describe and catalog Content Packages, as the Package author sees fit.

6.3.1 External Meta-Data

A Content Packaging implementation may, but does not need to, include an extension that references an external meta-data file. This version of the Content Packaging specification does not specify or recommend such a mechanism, but future versions of the Content Packaging specification may address the issue. An example of an in-line extension that refers to an external meta-data file is demonstrated by the following fragment:

```
<metadata>
   <schema>ADL SCORM</schema>
   <schemaversion>CAM 1.3</schemaversion>
   <adlcp:location>Lesson01.xml</adlcp:location>
</metadata>
```

This example is taken from the ADL SCORM 2004 profile.

6.4 <ORGANIZATIONS> Element

There are many ways to organize course or Package content, including no organization at all. In a manifest file, the <organizations> element contains this information.

It is possible to imagine organizations that will take into account such approaches as hierarchical branching, indexes, custom learning paths utilizing conditional branching, and complex objective hierarchies. If the course or Package presentation does not require a specific organization, the <organizations> element is still necessary and must appear as follows to satisfy the control rules expressed in the XSD: <organizations/>. However, in this case the <organizations> element is left empty.

While many content organization approaches may be developed, a default approach is included as part of this specification. This default approach to content organization, similar to a tree view or hierarchical representation, is encompassed in the <organization> element. The <organization> element is the only element allowed under <organizations>. Content may have additional organization schemas, through the use of the type attribute by setting it to a non-default value. There can be multiple organizations and more than one of the same type, but only one specified as the default.

6.4.1 <ORGANIZATION> Element

The <organization> element contains information about one particular, passive organization of the material. The <organization> element assumes a default structure attribute value of hierarchical, such as is common with a tree view or structural representation of data. Future versions of the specification will likely include additional values for the structure attribute to correspond with additional structural organizations or shapes, such as a directed graph, a semantic network, or others. Until additional values are agreed upon, the <organization> element, by default, effectively reads: <organization structure=hierarchical>.

If there is more than one <organization> element *within the same <organizations> element*, then it is expected that they should be variant organizations with substantially the same learning outcomes. Material with substantially different objectives should appear in separate Packages. It should always be the case that the meta-data at the <manifest> element level describe the purpose of the Package as a whole.

Where an <organizations> element contains multiple <organization> elements, the following procedure is recommended, if one <organization> is to be selected for any reason:

(a) If there is a value given for the default attribute of <organizations>, then this identifies the organization to be used. This is the preferred method for identifying a particular <organization>;

(b) If there is no default given, then the first <organization> element encountered should be used.

Software that processes a Content Package may use the above procedure or it may:

(a) Use organization-level meta-data to make a selection, using its own rules;

(b) Allow users to select the organization;

(c) Use any other suitable approach.

The presentation structure of <organization> is described through <item> sub-elements. An <item> may contain subordinate <item> elements (a hierarchical approach to presentation) or may appear on the same level as other <item>s (a flat approach). A tree view, or hierarchical representation, may be defined by the nesting levels of the <item> elements. Content developers can mix and match nesting levels as appropriate for their content. An <item> always has an identifier, and is linked to resources through an identifierref attribute. Titles are optional, but encouraged. The <item> element may also be visible or hidden, the default presence is visible.

Authors may also include meta-data within the <organization> and <item> elements allowing them to describe additional information meaningful for searching or for indexing in a repository.

Example: A hierarchical organizational scheme for a manifest can be determined by the order and nesting of the <item> elements contained within the <organization> element, similar to the following:

```
<organization identifier="TOC1">
   <title>Default Organization</title>
   <item identifier="ITEM1" identifierref="RESOURCE1">
           <title>Lesson 1</title>
   </item>
   <item identifier="ITEM2" identifierref="RESOURCE2">
           <title>Lesson 2</title>
   </item>
   <item identifier="ITEM3" identifierref="RESOURCE3">
           <title>Lesson 3</title>
   </item>
</organization>
```

An LMS or content viewer encountering this structural organization or hierarchical tree view of the content could interpret it conceptually as:

- Lesson 1
  - Lesson 2
- Lesson 3

6.4.2 Using Nested < MANIFEST > Elements

The mechanism for referencing an <item> element’s resource is the ‘identifierref’ attribute, which is used to reference resources. Certain restrictions are placed on the kinds of references that can be made in order to maintain the capability for future disaggregation of a compound Manifest, including:

(a) An <item> element’s identifierref can reference resources found in a subordinate <manifest> element in which it is contained. It can also reference the resources of any nested <manifest>;

(b) The reverse is not true: An <item> element’s identifierref cannot refer to a <manifest> element that is higher than the <manifest> element that contains it, or to any resource referred to by a higher-level <manifest> element. If it were to do so, such references could not be resolved should the contained Manifest be disaggregated and used to create a different Package. If content producers need to reference a separate, external Package, they must first aggregate it and then point down to it;

(c) An <item> element's identifierref can reference a sub-Manifest.

6.5 <RESOURCES> Element

The <resources> element identifies a collection of content and its files. Individual resources are declared as a <resource> element nested within the <resources> element. A <resource> is not necessarily a single file. It may be a collection of files that support the presentation of the associated presentation structure (<item> element). These files may be internally referenced or externally referenced via a URL. Internally referenced files must be included in the Package Interchange File.

A <resource> element may also have a <metadata> sub-element. The <metadata> element is for the <resource>, whether it is a single file or a collection of files.

A <file> element may contain a <metadata> sub-element allowing authors to describe additional <file> information meaningful for searching or for indexing in a repository. A <resource> may reference an internal (or local) file by a relative URL as the href or as an external file or service by a fully-qualified remote URL. Internal files used by the resource are either directly enumerated by <file> elements or indirectly enumerated by using the <dependency> element to reference another resource. For example, the union of all file enumerations in a Package identifies all files (excluding binding control documents and imsmanifest.xml files) that must be communicated on transmission of a content Package. External referents do not form part of the Package and do not appear in <file> elements.

A <resource> element may also contain a <dependency> sub-element. The <dependency> element identifies a single resource which can act as a container for multiple files that this resource depends upon. Rather than having to list all resources item by item each time they are needed, <dependency> allows authors to define a container of resources and to simply refer to that <dependency> element instead of individual resources. The same restrictions on the values of the identifierref attribute apply to <dependency> as apply to <item> (see Section 4.4.2 for further guidance), with the exception of referring to resources in sub-Manifests. An <item> can do this, a <dependency> can't. Below is an example of using <dependency>.

```
<resources>
   <resource identifier="R_A1" type="webcontent" href="sco06.html">
      <metadata/>
      <file href="sco06.html" />
      <file href="scripts/APIWrapper.js" />
      <file href="scripts/Functions.js" />
      <dependency identifierref="R_A4" />
      <dependency identifierref="R_A5" />
      <dependency identifierref="R_A6" />
   </resource>
   <resource identifier="R_A2" type="webcontent" href="sco1.html">
```

```
      <metadata/>
      <file href="sco1.html" />
      <file href="scripts/APIWrapper.js" />
      <file href="scripts/Functions.js" />
      <dependency identifierref="R_A5" />
   </resource>
   <resource identifier="R_A4" type="webcontent" href="pics/distress_sigs.jpg">
      <metadata/>
      <file href="pics/distress_sigs.jpg" />
   </resource>
   <resource identifier="R_A5" type="webcontent" href="pics/distress_sigs_add.jpg">
      <metadata/>
      <file href="pics/distress_sigs_add.jpg" />
   </resource>
   <resource identifier="R_A6" type="webcontent" href="pics/nav_aids.jpg">
      <metadata/>
      <file href="pics/nav_aids.jpg" />
   </resource>
</resources>
```

The ‘type’ attribute is usually set to ‘webcontent’ when describing material that is to be launched through a Web browser. However, when a Content Package is being used to contain data such as a QTI-XML based assessment then the value of the ‘type’ attribute should be set as recommended in Section 7 of the Implementation Handbook titled ‘Using IMS Content Packaging to Package Instances of LIP and Other IMS Specifications’ [IMSBUND, 01]. The ‘type’ attribute is set to ‘imsldcontent’ when used to contain learning design information. In situations where none of the previous methods are appropriate, then this attribute should be set to ‘other’.

Apart from the usage of the vocabulary in the Implementation Handbook it is recommended that this vocabulary is NOT extended. Later versions of this specification will address the extension of this vocabulary.

6.6 Examples of <RESOURCES> and Nested <MANIFEST>Elements

There is an example available for download from the IMS website that illustrates how to describe in-line sub-Manifests. You can find this sample and others at http://www.imsglobal.org/content/packaging/. For an example of how external sub-Manifests may be described in a future version of the specification, see Appendix D.

6.7Building an IMS Package or Package Interchange File

(1) Any namespaces required within a Package should be declared as attributes of the top-level <manifest> element;

(2) The imsmanifest.xml file and any files supporting namespaces (DTD, XSD) that are referenced internally must be placed at the root of the Package or compressed Package Interchange File;

(3) All internally referenced files must be stored in the paths declared in all <resource> elements in a Package.

6.8 Aggregation and Disaggregation of Packages

If a simple (non-aggregated) Package is to be aggregated into a new (super-)Package, first its manifest must be accessed and its list of <resource> elements obtained. These are traversed and each of their <file> elements examined to determine whether they reference external or internal files (note that any base address attribute in the <resources> elements and any overriding base address attributes in each <resource> element, need to be prefixed to a <file> element’s file references). This is used to build a list of all the files contained locally in the Package that is being aggregated. This list in turn, is then used to access each file and to create a copy of it in the new Package. Next, the manifest of the Package being aggregated must be integrated as a subordinate <manifest> element into the manifest that is being created for the containing Package. When the construction of the new Package is complete, the containing manifest is saved as a file with the name imsmanifest.xml at the root of the new Package Interchange File.

If a Package is to be disaggregated from a containing Package into a smaller, sub-Package, first that sub-Package’s <manifest> element must be accessed in the containing *imsmanifest.xml* file. The <resources> section of the accessed manifest is then read to determine the physical files that were originally contained in that section. This list is then used to locate these files in the larger Package and these are then copied to the new, smaller Package. The accessed manifest is then saved as a file with the name imsmanifest.xml and also included at the root of the new Package Interchange File.

If a compound Package, containing aggregated sub-Packages, is itself to be aggregated, then the same procedure is followed; with the addition that the compound Package’s sub-Manifest elements also have to be walked in order to build a complete list of files referenced in all the sub-Manifests. As the aggregated Package’s manifest already contains all the nested sub-Manifests, only this manifest needs to be merged into the new containing manifest. Similarly, if a compound sub-Package is to be disaggregated, its sub-Manifest tree needs to be walked in order to build the complete list of files that need to be copied into the disaggregated Package.

Packages, specifically organizational items, may not reference Package elements (<resource> elements) that are outside the Package scope. Referenced elements must be contained in the same Package from which they were referenced, including elements that are in sub-Packages within the Package. This specification does not contain rules as to how such referenced elements should be maintained by aggregation and disaggregation tools. The issue of intellectual property rights, concerning how resources preserve their original, unique identifiers is beyond the scope of this version of the Content Packaging specification.

6.8.1 Identifiers

When creating or manipulating Packages, the scope of identifiers needs to be considered. In order to be a valid Content Packaging Manifest, identifiers must be unique. If a Package is aggregated into another Package, identifier collisions could be avoided or resolved by using universally unique identifiers across manifests (such as identifiers generated or obtained according to the IMS Persistent, Location-Independent Resource Identifier Handbook [IMSPLID, 01]). If universally unique identifiers are not used in a system’s own storage scheme, Packages should not be exchanged with other systems without building unique identifier generation into tools that support Package aggregation.

The XML binding uses the XML structures ‘xsd:ID’ and ‘xsd:IDREF’ to validate the uniqueness of the identifiers. This ensures that the identifiers are unique within the XML document and that any identifier referenced using ‘IDREF’ must have a corresponding ‘ID’ declaration within the XML document. The usage of ‘ID’ and IDREF’ do not ensure globally unique identifiers and so care needs to be taken when using package aggregation. It is also important that the ‘ID’ declaration for the identifier is passed into a newly aggregated package otherwise a parser validation error will occur.

6.8.2 XInclude

The IMS Content Working Group expects that the XInclude mechanism, when fully approved and supported by the W3C, may prove a powerful way to support the aggregation and disaggregation of Package resources. However, authors should not use XInclude in packaging content until the W3C finalizes XInclude as a Recommendation and the XML community generally supports it.

Note: XInclude is mentioned here as an emerging standard that IMS will likely leverage in future versions of the Content Packaging specification rather than invent another way of including external XML files. See Appendix D for examples of how XInclude might be used in future versions of this specification.

6.8.3 xml：base

‘xml:base’ is a construct used to explicitly specify the base URI of a document in resolving relative URIs in links to external files. In the ‘imsmanifest.xml’ file, internal and external references may be absolute or relative. Relative addresses can be prefixed by an ‘xml:base’ attribute. The ‘xml:base’ attribute allows both external and local base addresses to be specified. Relative URLs, in the absence of ‘xml:base’, are relative to the Package root (location of ‘imsmanifest.xml’). In the presence of an ‘xml:base’ path, relative URLs are relative to the path specified in ‘xml:base’. When an ‘xml:base’ path is relative itself, the absolute path is then resolved to the location of the containing document. That is, the location of the ‘imsmanifest.xml’ file in an importing system, when it is read, supplies the missing absolute segment, per the rules expressed in RFC 2396.

Relative ‘xml:base’ paths that are declared in a sub-manifest are relative to the Package root. In cases where a manifest with a declared ‘xml:base’ path contains a sub-manifest, and the sub-manifest also declares an ‘xml:base’ path, the multiple ‘xml:base’ paths should not be concatenated at runtime. Instead, the URIs within such a sub-manifest are relative to the declared xml:base of the sub-manifest only. Implementors are, of course, free to construct a relative sub-manifest ‘xml:base’ path by concatenation or any other means at aggregation time.

In the presence of an xml:base path, which references an external location, the relative URLs are

relative to that location. Absolute (external) URLs are considered to be fully-specified without the provision of additional pathing.
When the ‘xml:base’ attribute is used, care must be taken not to exceed the length of any associated ‘href’. The maximum length of both ‘href’ and ‘xml:base’ is defined as 2000 octets. In cases where multiple ‘xml:base’ values need to be concatenated to create the full path then care must be taken to ensure that the total length does not exceed that of the ‘href’. If the path length is greater than 2000 octets then the system behavior is undefined.

When using xml:base in packaging, the xml:base path should not begin with a leading forward slash. As defined in RFC 2396, a path with a leading forward slash indicates the absolute path of that resource. Using a leading forward slash can easily be misinterpreted as declaring the document as the local host. With this in mind, the xml:base attribute is most useful for specifying relative paths to sub-directories containing content Package resources. Below is an example of using xml:base to specify the path to resources that are internal and relative.

```
<manifest xmlns = "http://www.imsglobal.org/xsd/imscp_v1p1"
    xmlns:imsmd = "http://www.imsglobal.org/xsd/imsmd_v1p2"
    xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation = "http://www.imsglobal.org/xsd/imscp_v1p1
      http://www.imsglobal.org/xsd/imscp_v1p1.xsd
      http://www.imsglobal.org/xsd/imsmd_v1p2
      http://www.imsglobal.org/xsd/imsmd_v1p2.xsd "
      identifier="Manifest1-CEC3D3-3201-DF8E-8F42-3CEED12F4198"
      version="IMS CP 1.1.4">
    <metadata>
        <schema>IMS Content</schema>
        <schemaversion>1.1</schemaversion>
        <imsmd:lom>
            <imsmd:general>
                <imsmd:title>
                    <imsmd:langstring xml:lang="en_US">IMS Content Packaging Sample - A
Relative xml:base</imsmd:langstring>
                </imsmd:title>
            </imsmd:general>
        </imsmd:lom >
    </metadata>
    <organizations default="TOC1">
        <organization identifier="TOC1">
            <title>default</title>
            <item identifier="ITEM1" identifierref="RESOURCE1">
                <title>Lesson 1</title>
                <item identifier="ITEM2" identifierref="RESOURCE2">
                    <title>Introduction 1</title>
                </item>
                <item identifier="ITEM3" identifierref="RESOURCE3">
                    <title>Content 1</title>
                </item>
                <item identifier="ITEM4" identifierref="RESOURCE4">
                    <title>Summary 1</title>
                </item>
            </item>
            <item identifier="ITEM5" identifierref="RESOURCE5">
                <title>Lesson 2</title>
                <item identifier="ITEM6" identifierref="RESOURCE6">
```

```
                    <title>Introduction 2</title>
                </item>
                <item identifier="ITEM7" identifierref="RESOURCE7">
                    <title>Content 2</title>
                </item>
                <item identifier="ITEM8" identifierref="RESOURCE8">
                    <title>Summary 2</title>
                </item>
            </item>
        </organization>
    </organizations>
    <resources>
        <resource identifier="RESOURCE1" type="webcontent" href="lesson1.htm"
xml:base="lesson1/">
            <file href="lesson1.htm"/>
            <file href="picture1.gif"/>
        </resource>
        <resource identifier="RESOURCE2" type="webcontent" href="intro1.htm"
xml:base="lesson1/">
            <file href="intro1.htm"/>
            <file href="picture2.gif"/>
        </resource>
        <resource identifier="RESOURCE3" type="webcontent" href="content1.htm"
xml:base="lesson1/">
            <file href="content1.htm"/>
            <file href="picture3.gif"/>
        </resource>
        <resource identifier="RESOURCE4" type="webcontent" href="summary1.htm"
xml:base="lesson1/">
            <file href="summary1.htm"/>
            <file href="picture4.gif"/>
        </resource>
        <resource identifier="RESOURCE5" type="webcontent" href="lesson2.htm"
xml:base="lesson2/">
            <file href="lesson2.htm"/>
            <file href="picture1.gif"/>
        </resource>
        <resource identifier="RESOURCE6" type="webcontent" href="intro2.htm"
xml:base="lesson2/">
            <file href="intro2.htm"/>
            <file href="picture2.gif"/>
        </resource>
        <resource identifier="RESOURCE7" type="webcontent" href="content2.htm"
xml:base="lesson2/">
            <file href="content2.htm"/>
            <file href="picture3.gif"/>
        </resource>
        <resource identifier="RESOURCE8" type="webcontent" href="summary2.htm"
xml:base="lesson2/">
            <file href="summary2.htm"/>
            <file href="picture4.gif"/>
        </resource>
    </resources>
</manifest>
```

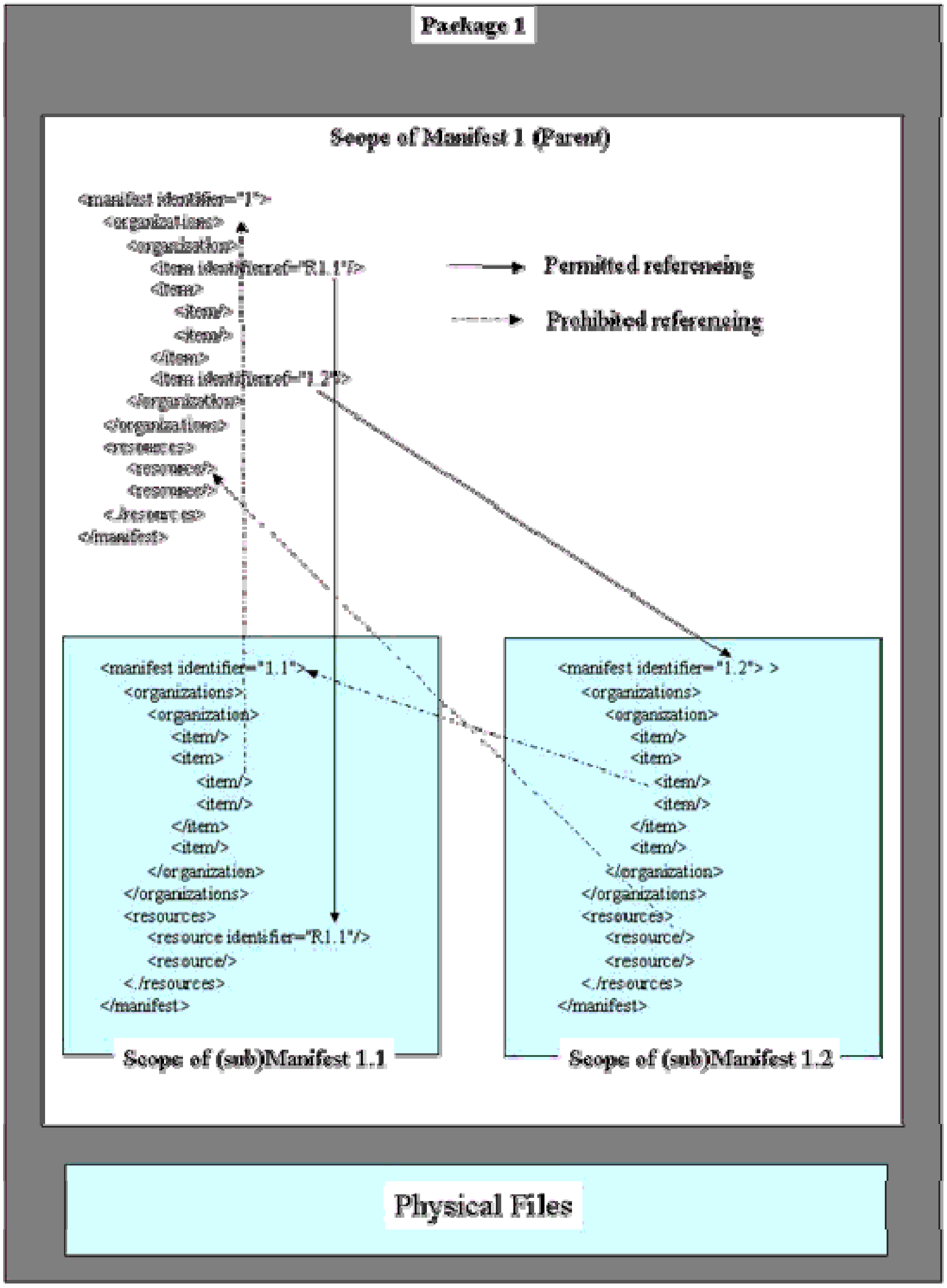
The following is an example of using xml:base to specify the path to resources that are external and absolute.

```
<?xml version="1.0"?>
<manifest identifier="MANIFEST1"
xmlns="http://www.imsglobal.org/xsd/ims_cp_rootv1p1">
    <metadata>
        <schema>IMS Content</schema>
        <schemaversion>1.1</schemaversion>
        <imsmd:lom>
            <imsmd:general>
                <imsmd:title>
                    <imsmd:langstring xml:lang="en_US">IMS Content Packaging Sample - A
Remote xml:base</imsmd:langstring>
                </imsmd:title>
            </imsmd:general>
        </imsmd:lom>
    </metadata>
    <organizations default="TOC1">
        <organization identifier="TOC1">
            <title>Big Title</title>
            <item identifier="ITEM1" identifierref="RESOURCE1">
                <title>Lesson 1</title>
                <item identifier="ITEM2" identifierref="RESOURCE2">
                    <title>Introduction 1</title>
                </item>
                <item identifier="ITEM3" identifierref="RESOURCE3">
                    <title>Content 1</title>
                </item>
                <item identifier="ITEM4" identifierref="RESOURCE4">
                    <title>Summary 1</title>
                </item>
            </item>
        </organization>
    </organizations>
    <resources xml:base="http://repository.imsglobal.org/foo/bar/">
        <resource identifier="RESOURCE1" type="webcontent" href="lesson1.htm"/>
        <resource identifier="RESOURCE2" type="webcontent" href="intro1.htm"/>
        <resource identifier="RESOURCE3" type="webcontent" href="content1.htm"/>
        <resource identifier="RESOURCE4" type="webcontent" href="summary1.htm"/>
    </resources>
</manifest>
```

6.8.4Package Scope

The scoping rules for manifests and sub-manifests is shown in Figure6.2.

Figure 6.2 Scoping rules for sub-manifests.



The scope of Package 1's manifest is considered to be itself and any sub-Manifests defined within Package 1. This includes the manifest of Package 1 (Manifest 1) and any sub-manifests found in the sub-packages - Manifest 1.1 and Manifest 1.2.

The scope of Package 1.1’s manifest is itself and any sub-Manifests defined within Package 1.1. This includes the manifest of Package 1.1 (Manifest 1.1) and any sub-Manifests defined within Package 1.1. In Figure 6.2 there are no sub-Manifests defined in Package 1.1, so the scope is itself.

The scope of Package 1.2’s manifest is itself and any sub-Manifests defined within Package 1.2. This includes the manifest of Package 1.2 (Manifest 1.2) and any sub-Manifests defined within Package 1.2. In Figure 6.2 there are no sub-Manifests defined in Package 1.2, so the scope is itself.

Packages, specifically organizational items, may not reference manifests elements (<resource> elements) that are outside the scope of manifest. Referenced elements must be contained in the same manifest from which they were referenced, including elements that are in sub-manifests within the Package. In the case above, Package 1 manifest’s elements can reference elements found in sub-Manifest 1.1 and sub-Manifest 1.2, since these are in scope of Package 1’s manifest. sub-Manifest 1.1 and sub-Manifest 1.2 can only reference manifest elements within itself. Sub-Manifests elements are prohibited from referencing (dashed lines above) manifest elements in any manifests in which they are contained (child manifests elements are not permitted to reference elements in any parent manifests).

While it is currently permitted for an ‘item’ in the parent manifest to reference a specific ‘resource’ in a sub-Manifest (the reference to ‘resource’ R1.1 in Figure 6.2), it is debatable whether or not this is good practice. This is because alternative XML schemes are available to support such a referencing mechanism and so it is possible that this form of direct reference will be disallowed in later versions of this specification.

6.8.5<identifierref> Referenced Elements

The following elements can be referenced using the identifierref attribute of an item:

(a) Identifier attribute of a manifest (references the entire manifest that is in scope of the referencing manifest);

(b) Identifier attribute of a resource (references the resource found in a sub-Manifest that is in scope of the referencing manifest);

(c) Identifier attribute of an item (references the item found in a sub-Manifest that is in scope of the referencing manifest);

(d) Identifier attribute of an organization (references the organization found in a sub-Manifest that is in scope of the referencing manifest).

6.9 Min/Max Binding Constraints

In the Content Packaging Information Model the concept of min/max was adopted. This concept assumes that the size constraints are defined such that an implementation must support the given value as the smallest possible maximum size. In the case of an identifier with a min/max size of 1000 characters then the smallest maximum size of the identifier supported in each and every implementation is 1000 characters. Some implementations may support larger sizes but interoperability is defined such that only the first 1000 characters is guaranteed to be exchanged consistently.

This constraint is not enforced in the XML schema (XSD). Therefore a validating parser cannot enforce the constraint as defined by the Information Model. Therefore, system implementations must be such that all min/max constraints are explicitly supported.

6.10 Using the ‘isvisible’ Attribute

The ‘isvisible’ attribute is used to denote if the ‘item’ is to be visible when the organization tree is rendered for the ‘user’ by the system. The default value for ‘isvisible’ is ‘true’ and this must be assumed if the attribute is not used on the ‘item’. This property is not inherited by the children of an ‘item’. Some examples of the rendering is shown in the following Table 6.1.

Table 6.1 - Examples of using the ‘isvisible’ attribute.

Example XML Code	Rendered Items

`<item identifier="1">`
   `<title>A</title>`
   `<item identifier="2">`
       `<title>B</title>`
       `<item identifier="3">`
           `<title>C</title>`
       `</item>`
   `</item>`
   `<item identifier="4">`
      `<title>D</title>`
   `</item>`
`</item>`
`<item identifier="5">`
   `<title>E</title>`
`</item>`
	A
	   B
	       C
	    D
	E

`<item identifier="1" isvisible="false">`
   `<title>A</title>`
   `<item identifier="2">`
      `<title>B</title>`
      `<item identifier="3">`
          `<title>C</title>`
      `</item>`
   `</item>`
   `<item identifier="4">`
      `<title>D</title>`
   `</item>`
`</item>`
`<item identifier="5">`
   `<title>E</title>`
`</item>`
	B
	   C
	D
	E

<item identifier="1" isvisible="false">
  <title>A</title>
  <item identifier="2" isvisible="true">
    <title>B</title>
    <item identifier="3" isvisible="true">
      <title>C</title>
    </item>
  </item>
  <item identifier="4" isvisible="true"> <title>D</title>
  </item>
</item><item identifier="5" isvisible="true">
  <title>E</title>
</item>
	B
	  C
	D
	E

<item identifier="1" isvisible="true">
  <title>A</title>
  <item identifier="2" isvisible="false">
    <title>B</title>
    <item identifier="3" isvisible="false">
      <title>C</title>
    </item>
  </item>
  <item identifier="4" isvisible="false">
    <title>D</title>
  </item>
</item>
<item identifier="5" isvisible="false">
  <title>E</title>
</item>
	A

<item identifier="1" isvisible="true">
  <title>A</title>
  <item identifier="2" isvisible="false">
    <title>B</title>
    <item identifier="3" isvisible="true">
      <title>C</title>
    </item>
  </item>
  <item identifier="4" isvisible="false">
    <title>D</title>
  </item>
</item>
<item identifier="5" isvisible="false">
  <title>E</title>
</item>
	A
	  C

<item identifier="1" isvisible="true">
  <title>A</title>
  <item identifier="2" isvisible="true">
    <title>B</title>
    <item identifier="3" isvisible="false">
      <title>C</title>
    </item>
  </item>
  <item identifier="4" isvisible="false">
    <title>D</title>
  </item>
</item>
<item identifier="5" isvisible="false">
  <title>E</title>
</item>
	A
	  B

7. Validation

The XML 1.0 Specification from the W3C allows for two types of parsers: validating and non-validating. Non-validating parsers are only concerned with the well-formedness of a document—that is, ensuring that the syntactic rules of XML have been followed. Validating parsers, on the other hand, are required to implement the full XML 1.0 Specification. This means that validating parsers must follow all of the rules concerning structure, data types, and external references that are specified by a schema.

Schemas describe which elements may exist in a document and how those elements may be structured. The IMS Content Packaging specification provides limited guidance on the use of XML Schema Definition (XSD). While each of these schemas has different capabilities, any of these schemas can provide basic document validation. It is expected that any Manifest document in a Package that is written according to the IMS Content Packaging specification can be validated using the XSD schema available with this specification.

The IMS Content Packaging specification is accompanied by one XSD (imscp_v1p1.xsd). While it is technically feasible to validate documents that use DTDs, it is not possible to use a DTD to differentiate between two elements that use an element name in incompatible ways (for example IMS Meta-Data and IMS Content Packaging both use <resource> in meaningful, but incompatible ways, and IMS Content Packaging and IMS Question and Test both use <item> in meaningful, but incompatible ways). Rather than alter the IMS Content Packaging Information Model to adjust to the requirements of DTD validation, the Content Working Group made a decision to be forward-looking, towards XML Schemas, with respect to validation.

7.1 W3C SCHEMA Validation

IMS has updated the Content Packaging Schema to support the Final Recommendation of the W3C XML Schema specification (dated) 2 May 2001. Currently, several commercial tools support Schema validation including: Xerces, XML Authority, XML Spy, and Oracle parsers.

The 'xml:' namespaced attributes are defined in file 'http://www.w3.org/2001/xml.xsd'. This is the reference to be used for on-line validation or a copy of this file must be placed in the root of the content package for local validation.

Note that the file 'http://www.w3.org/2001/xml.xsd' is always the most up to date version of the file 'http://www.w3.org/2001/03/xml.xsd' as maintained by W3C.

8. Conformance

Conformance to a packaging specification is an important issue for stakeholders involved with the IMS Content Packaging specification. Conformance clarifies content interoperability. It sets an expectation for content vendors and their customers about how that content will be repackaged, and possibly used by compliant LMSs, computing platforms supporting instructional content, and learning service providers as content moves about within systems, between systems, and across the Web. It also helps LMS vendors, computing platforms, and learning services to control the scope of their data stores and tools or sub-systems required to operate on content Packages.

This specification addresses two levels of conformance to guide content developers in how LMS vendors, computing platforms, or learning services may deal with the elements and extensions content developers place within an IMS Manifest file. These same levels of conformance should guide those who repackage content for redistribution within their systems, across systems, or across the Web.

8.1 Package Conformance

For the purposes of conformance, an IMS Content Package is the relevant imsmanifest.xml file and all resources directly or indirectly referenced by this document (also known as the Package Interchange File).

8.1.1 Package Conformance Level 0 (no extensions)

(a) The Package must contain a file called imsmanifest.xml in the root of the distribution medium (archive file, CD-ROM, etc.);

(b) The Package must contain any directly referenced controlling files used DTD, XSD) in the root of the distribution medium (archive file, CD-ROM, etc.);

(c) The imsmanifest.xml file must contain well-formed XML that adheres to the XML format described in section 3 of the IMS Content Packaging XML Binding specification;

(d) If the imsmanifest.xml file contains IMS Meta-Data, it must contain a namespace extension to include meta-data according to the IMS Meta-Data Specification v1.2.1;

(e) The imsmanifest.xml file must not reference any elements using XInclude. (This requirement may be relaxed when it is generally supported in XML parsers);

(f) All files that a local resource (i.e., a resource that is contained entirely within the Package Interchange File) is dependent on must be identified by <file> elements in the <resources> section of the imsmanifest.xml file and must be contained within the directory or sub-directories that contain imsmanifest.xml.

8.1.2 Package Conformance Level 1 (urilizes extensions)

(a) All level 0 conformance requirements (except ‘e’) apply;

(b) The imsmanifest.xml file may contain additional namespace extensions. If additional namespace extensions are described and controlled using a schema or modified DTD, then any directly referenced control files must be included in the Package.

8.2 System and Tool Conformance

For the purposes of conformance, system and tool conformance refers to the systems and tools that import, export, create, and manipulate IMS Content Packages.

8.2.1 System and Tool Conformance Level 0 (may not preserces extensions)

(a) A conforming system or tool must recognize and process any conforming IMS Content Package that conforms to level 0 or level 1. The features and functionality of systems and tools that process IMS Content Packages are purposely not specified;

(b) All elements of the IMS Content Packaging XML Binding Specification v1.1.4 and IMS Meta-Data Specification v1.2.1 or the IEEE P1484.12.3 Draft Standard for Extensible Markup Language (XML) Schema Definition Language Binding for Learning Object Metadata that are present in imsmanifest.xml must be preserved upon re-transmittal;

(c) Name-spaced extensions, other than the IMS Meta-Data Specification v1.2.1 namespace or the IEEE P1484.12.3 Draft Standard for Extensible Markup Language (XML) Schema Definition Language Binding for Learning Object Metadata namespace, may be ignored and may not be re-transmitted.

8.2.2 System and Tool Conformance Level 1 (preserces extensions)

(a) Level 0 conformance requirements (a) and (b) apply;

(b) All name-spaced extensions must be preserved upon re-transmittal.

8.3 Best Practice Recommendations for IMS Package Conformance Levels

This section contains additional recommendations to support the functionality and interoperability of IMS Content Packages.

(1) A general recommendation to all who create, deliver, or repackage content is that they publish at their public Websites which level of the IMS Content Package Conformance Level or System and Tool Conformance Levels they support. An organization or enterprise that originates a namespace extension is encouraged to make public the DTD or XSD files that define it;

(2) It is expected that content producers will organize their content for expected aggregations or disaggregation. That is, if content producers do not expect, or desire their content to be disaggregated, it should be encoded in a monolithic manifest. Conversely, sub-Manifests should be used to organize content according to expected levels of aggregation and disaggregation;

(3) The IMS Content Working Group expects that vendors of training systems, platforms, and learning spaces will actively use namespaced elements that are relevant to their product(s) or the training communities they serve. Additionally, content creators may want to use proprietary namespaces to support a richer set of features in their content than would otherwise be available, and negotiate support for those features with vendors of training systems, platforms, and learning spaces. Hence, the IMS Content Working Group strongly encourages systems and tools to recreate an originating IMS Manifest file’s use of third party namespaces and namespaced elements when such content is repackaged for transmission from their system or tool to elsewhere on the Web;

(4) Content re-packagers should be guided by an original Package’s use of sub-Manifests or references to external manifests when aggregating or disaggregating content. That is, a portion of a

course or curriculum that is a candidate for aggregation or disaggregation will be held in a sub-Manifest. So, a system or tool should preserve the original sub-Manifest(s) or externally referenced manifests or, be able to replicate them when repackaging content to export out of their environment. It is expected that there will be no additions or deletions to elements and attributes within a sub-Manifest or externally referenced manifest.

9. Extensibility

To allow developers the most flexibility possible, the XML binding of a manifest may be freely extended. All elements that serve as containers for other elements may be extended to include new elements. Elements that contain data types (e.g., string, integer) and elements with a ‘closed’ data model may not be extended. Examples of elements with a closed data model include <schema> and <schemaversion>. Extensions must provide references (e.g., via namespacing) to the source of the extensions.

There are at least two cases where extensions can cause problems for developers. The first case is when interoperability with other content packaging tools and vendors is required. Custom extensions must then be agreed upon between individual parties making global interoperability very difficult. The second case is when a developer wishes to add extensions and also provide or alter a schema that will allow document validation. Each schema DTD or XSD) requires a different approach to handle extensions that can be validated. The following sections provide some brief explanations of approaches that may be used for handling extensions.

Note: The following examples consist of XML fragments to illustrate basic concepts of extensibility. These samples are not well formed and are missing some information such as any references to a control document (DTD or XSD). Complete sample files with their associated schemas can be found at http://www.imsglobal.org/content/packaging/.

9.1Extending <metadata>

A content publisher or LMS vendor may need to transport or store meta-data that is not defined by the IMS Meta-Data Specification v1.2.1 [MD, 901].

For example, assume the fictitious LMS ‘LitWare Inc.,’ needs to maintain meta-data about the Instructional Design methodology used to create a course. The following steps illustrate how easily this can be done when using a schema based upon XML Schema Definition Language:

(1) Create an XML schema that defines the new element(s). For the given example, the XML schema could consist of the following:

```
<xsd:schema targetNamespace="http://www.litwareinc.net/xsd/litware"
     xmlns:xml="http://www.w3c.org/XML/1998/namespace"
     xmlns:xsd="http://www.w3c.org/2001/XMLSchema"
     xmlns:xsi="http://www.w3c.org/2001/XMLSchema-instance"
     xmlns="http://www.litwareinc.org/xsd/litware"
     elementFormDefault="qualified">
<xsd:element name="instructionaldesignmethodology" type="xsd:string"/>
```

```
</xsd:schema>
```

(2) When exporting to the learning management system, the element would appear as follows in imsmanifest.xml.

The following sample shows a way to extend the IMS Meta-Data v1.2 [MD, 501]:

```
<manifest identifier=MANIFEST1>
     <metadata>
          <schema>IMS Content</schema>
          <schemaversion>1.1</schemaversion>
          <imsmd:lom>
             <imsmd:general>
               <imsmd:title>
                         <imsmd:langstring xml:lang=en_US>Sample
               Manifest</imsmd:langstring>
               </imsmd:title>
               <imsmd:description>
                         <imsmd:langstring xml:lang=en_US>Metadata
               tensions</imsmd:langstring>
               </imsmd:description>
               <litware:instructionaldesignmethodology>LWI Mindmapping Methodology
               </litware:instructionaldesignmethodology>
             </imsmd:general>
          </imsmd:lom>
     </metadata>
     <organizations> . . .</organizations>
     <resources>. . .</resources>
</manifest>
```

9.2 Extending <organizations>

It is expected that over time, many different approaches to content organization will emerge. The ADL has been developing one such approach in connection with its releases of SCORM versions. At the time of this specification release, the sample manifest, included with the Bindings and Examples from the Content Packaging website (http://www.imsglobal.org/content/packaging/) was a work in progress and may not be the final direction the ADL takes in future versions of SCORM. While some of the ideas expressed in this sample may not be complete and the file can not be properly validated, it still provides a good conceptual model of how the IMS Content Packaging specification allows different content organization schemes to essentially ‘plug-in’ to a Package Manifest file.

9.3Extending <resources>

Extending <resources> using both external and in-line references is an important feature of Content Packaging. However, IMS is currently doing more testing and work in this area before providing samples of extending <resources> in this Best Practice Guide.

9.4 Extending with DTDs

In the examples above, the content models of the schemas must be ‘open’ to enable extensibility. To accomplish the same goal using the IMS Content Packaging DTD, a new DTD must be created to include the extensions. Such a DTD would differ from the IMS Content Packaging DTD. This approach would allow a document to be validated with extensions in it, but it limits the interoperability of the content Package.

Annex A
Supporting Files

A number of supporting files accompany the IMS Content Packaging specification documents and are available in the download .zip file (imscp_v1p1p4.zip). The files in the zip file are as follows:

imscp_infov1p1p4.pdf	IMS Content Packaging Information Model
\imscp_bindv1p1p4.pdf	IMS Content Packaging XML Binding
\imscp_bestv1p1p4.pdf	IMS Content Packaging Best Practice Guide (this document)
\imscp_sumcv1p1p4.pdf	IMS Content Packaging Summary of Changes
\schema\ imscp_v1p1.xsd	IMS Content XML Schema, version 1.1.4
\samples\All_Elements	Illustrates a simple manifest using Content Packaging elements.
\samples\QTI_Example	Illustrates a simple manifest packaging QTI elements.
\samples\Full_Metadata	Illustrates a manifest that uses all elements and attributes defined in the IMS Content Packaging specification.
\samples\Multiple_Organizations	Illustrates the use of multiple <organizations>, to provide different paths through a course.
\samples\Simple_Manifest	Illustrates a simple manifest.
\samples\Sub_Manifests	Illustrates the use of sub-Manifests to promote reuse. This example takes the Simple Manifest example, and implements it using sub-Manifests.

Annex B
Additional Resources

B1-Various Documents

IMS Content Documents

IMS Content Packaging Information Model:
http://www.imsglobal.org/content/packaging/

IMS Content Packaging XML Binding:
http://www.imsglobal.org/content/packaging/

IMS Content Packaging Best Practice Guide:
http://www.imsglobal.org/content/packaging/

IMS Meta-Data Documents

The IMS Meta-Data Best Practice and Implementation Guide:
http://www.imsglobal.org/metadata/

The IMS Learning Resource Meta-Data Information Model:
http://www.imsglobal.org/metadata/

IMS General Reference

IMS Persistent, Location-Independent Resource Identifier Handbook:
http://www.imsglobal.org/implementationhandbook/imsrid_handv1p0.html

Using IMS Content Packaging to Package Instances of LIP and Other IMS Specifications:
http://www.imsglobal.org/implementationhandbook/

ADL /AICC Documents

Sharable Content Object Reference Model: http://www.adlnet.org/

Aviation Industry CBT Committee (AICC) API for Web Implementation:
http://www.aicc.org/

Internet Engineering Task Force (IETF)

RFC 2396: Uniform Resource Identifiers (URI): http://www.ietf.org/rfc/rfc2396.txt

IEEE

IEEE LTSC 1484.12 Learning Object Metadata: http://www.ltsc.ieee.org/wg12/

XML

XML Version 1.0 specification of the W3C: http://www.w3.org/TR/1998/REC-xml-19980210

XML Namespace Recommendation of W3C: http://www.w3.org/TR/1999/REC-xml-names-19990114

XML Inclusion Technical Report: http://www.w3.org/TR/xinclude

XML Schema Recommendation of W3C: http://www.w3.org/TR/2001/REC-xmlschema-0-20010502/

B2- Namespacing and Schema Reference

The namespaces, filenames, and namespace prefixes for XML instances using the XML Schema files are as follows:

	Namespace	Filename	Prefix
Content Packaging	http://www.imsglobal.org/xsd/imscp_v1p1	imscp_v1p1.xsd	imscp:
Meta-Data	http://www.imsglobal.org/xsd/imsmd_v1p2	imsmd_v1p2.xsd	imsmd:
LIP	http://www.imsglobal.org/xsd/imslip_v1p0	imslip_v1p0.xsd	imslip:
QTI	http://www.imsglobal.org/xsd/imsqti_v1p1	imsqti_v1p1.xsd	imsqti:
Simple Sequencing	http://www.imsglobal.org/xsd/imsss_v1p0	imsss_v1p0.xsd	imsss:
Learning Design	http://www.imsglobal.org/xsd/imsld_v1p0	imsld_v1p0.xsd	imsld:

All of the samples provided with this specification, as listed in Appendix A, make use of the schema XSD) files located on the IMS website. The specification editors used XML Spy v5.1 and Turbo XML v2.3.1 to validate each of the samples listed in Appendix A, against the XSD files on the IMS website. It is expected that other XML Schema-capable parsers will also validate sample files as long as the parser is able to locate the online XML Schema files. It is best practice to use the online Schema file references (see Online XSD Files example below) as the XSD files on the IMS website will be the most up-to-date. Using the online XSD files requires the parser to have an open, functional connection to the Internet. If, however, an Internet connection is not available or users wish to validate files locally, they will need to change the namespace declarations in their samples to match the Local XSD Files example below.

Online XSD Files

For those XML instances using the XSD files as located on the IMS website, the declaration in the root <manifest> element is of the form:

```
<manifest
   xmlns=http://www.imsglobal.org/xsd/imscp_v1p1xmlns:ims
   md=http://www.imsglobal.org/xsd/imsmd_v1p2xmlns:xsi=htt
   p://www.w3.org/2001/XMLSchema-instancexsi:schemaLocat
   ion="http://www.imsglobal.org/xsd/imscp_v1p1
                    http://www.imsglobal.org/xsd/imscp_v1p1.x
                    sd
                    http://www.imsglobal.org/xsd/imsmd_v1p2
                    http://www.imsglobal.org/xsd/imsmd_v1p2.
                    xsd"
   identifier="Manifest01" version="IMS CP 1.1.4">
```

Local XSD Files

For XML instances in which the XSD files are locally available, in the same directory as the instance, the declaration in the root <manifest> element is of the form:

```
<manifest
    xmlns="http://www.imsglobal.org/xsd/imscp_v1p1"xmlns:imsmd
    ="http://www.imsglobal.org/xsd/imsmd_v1p2"xmlns:xsi="http://
    www.w3.org/2001/XMLSchema-instance"xsi:schemaLocation="h
    ttp://www.imsglobal.org/xsd/imscp_v1p1.xsd
                                imscp_v1p1.xsdhttp://www.imsglobal.
                                org/xsd/imsmd_v1p2imsmd_v1p2.xsd
                                "
    identifier="Manifest01" version="IMS CP 1.1.4">
```

The ‘version’ attribute is optional.

Annex C
Harmonization

Harmonization between IMS specifications is important and IMS is committed to ensuring its specifications use similar strategies for vocabularies, GUIDs, element names, and others across all standards. For information about harmonization or sample implementations of the IMS Content Packaging specification and other IMS specifications,see the following:

- Using IMS Content Packaging to Package Instances of LIP and other IMS specifications. A general implementation handbook illustrating how to package instances of LIP that could also be applied to packaging instances of Meta-Data, QTI, or Enterprise. To download this document, visit the Implementation Handbook portion of the IMS website: http://www.imsglobal.org/implementationhandbook/.

Annex D
Possible Future Directions

This section describes open issues that the Content Packaging Working Group consider as recommendations for future version releases.

Table D1 List of issues to be resolved in the future.

Issue Identifier	Comment
CP113-49	Using URNs as identifiers or use PLIDs instead of xs:ID.
CP113-51	Clarification on the usage of the ‘isvisible’ attribute.
CP113-55	[identifierref] needs to be made more solid and clearly defined.
CP113-56	Common semantic interpretation of sub-Manifests.
CP113-62	Externalizing sub-Mmanifests.
CP113-63	Provide optional presentation hints.
CP113-64	Creating an RDF version of the spec.
CP113-65	Language for Title.
CP113-66	Ambiguity in boolean values in binding.
CP113-68	ADL ‘location’ extension for the ‘meta-data’ element.
CP113-80	Lack of clarity in package scope in BPG.
CP113-110	Adding “variation” element.
CP113-161	Extra files in manifest.

內容包裝

— 參考資料、爭議事項、英中名詞對照

1.參考資料

IMS Content Packaging Information Model v1.1.4, C.Smythe, A.Jackl, IMS Global Learning Consortium, Inc., October 2004.

IMS Content Packaging XML Binding v1.1.4, C.Smythe, A.Jackl, IMS Global Learning Consortium, Inc., October 2004.

IMS Content Packaging Best Practice and Implementation Guide v1.1.4, C.Smythe, A.Jackl, IMS Global Learning Consortium, Inc., October 2004.

IMS Content Packaging Summary of Changes v1.1.4, C.Smythe, A.Jackl, W.Kraan, IMS Global Learning Consortium, Inc., October 2004.

Using IMS Content Packaging to Package Instances of LIP and Other IMS Specifications v1.0, B.Olivier, M.McKell, IMS Global Learning Consortium, Inc., August 2001.

IMS Persistent, Location-Independent, Resource Identifier Implementation Handbook v1.0, M.McKell, IMS Global Learning Consortium, Inc., April 2001.

ISO (International Organization for Standardization). ISO/IEC 10646-1993 (E). Information technology - Universal Multiple-Octet Coded Character Set (UCS) - Part 1: Architecture and Basic Multilingual Plane. [Geneva]: International Organization for Standardization, 1993 (plus amendments AM 1 through AM 7).

IMS Meta-data Best Practice Guide for IEEE 1484.12.1-2002 Standard for Learning Object Metadata v1.3, P.Barker, L.Campbell, A.Roberts, IMS Global Learning Consortium, Inc., May 2004.

IMS Learning Resource Meta-data v1.2.1, S.Thropp, M.McKell, IMS Global Learning Consortium, Inc., September 2001.

IMS Learning Resource Meta-data v1.2, T.Anderson, M.McKell, IMS Global Learning Consortium, Inc., May 2001.

The Unicode Consortium. The Unicode Standard, Version 2.0. Reading, Mass.: Addison-Wesley Developers Press, 1996.

XML 1.0 Specification of the W3C:
http://www.w3.org/TR/1998/REC-xml-19980210.

XML Namespace Recommendation of W3C:
http://www.w3.org/TR/1999/REC-xml-names-19990114.
XML Schema Recommendation of W3C:
http://www.w3.org/TR/2001/REC-xmlschema-0-20010502/.

2.爭議事項

無

3.英中名詞對照表

	-A-
Amendment	修正
	-B-
Binding	繫結
	-C-
Character set	字元集
Conformance	符合性
Conformance statement	符合性聲明

Container	容器
Content	內容
Content Packaging, CP	*內容包裝*
	-D-
Dependency	*從屬物*
Document Type Definition, DTD	文件型式定義
Dynamic sequencing	*動態排序*
	-E-
Element	元件
	-F-
File	檔案
	-G-
	-H-
	-I-
Identifier	識別符
Information Model	資訊模型
Item	項目
	-J-
	-K-
	-L-
Learning Management System, LMS	學習管理系統
	-M-
Manifest	*內容清單*
Metadata	詮釋資料
	-N-
	-O-
Organization	組織
	-P-
Package	*套裝*
Parameter	參數
Persistent Location Independent Resource Identifier, PLIRI	*永久性定位獨立資源識別符*
Resource	資源
	-Q-
	-R-
	-S-
Schema	*架構*
Sub-manifest	*子內容清單*
	-T-
Title	標題
Traverse	遍歷
Type	型式
	-U-
	-V-

Version	版本

-W-

World Wide Web Consortium, W3C	全球資訊網聯盟

-X-

Extensible Mark-up Language, XML	可延伸標誌語言

-Y-

-Z-

4.中英名詞對照表

子內容清單	Sub-manifest
元件	Element
內容	Content
內容包裝	Content Packaging, CP
內容清單	Manifest
文件型式定義	Document Type Definition, DTD
套裝	Package
可延伸標示語言	Extensible Mark-up Language, XML
永久性定位獨立資源識別符	Persistent Location Independent Resource Identifier, PLIRI
全球資訊網聯盟	World Wide Web Consortium, W3C
字元集	Character set
版本	Version
型式	Type
架構	Schema
修正	Amendment
容器	Container
動態排序	Dynamic sequencing
參數	Parameter
從屬物	Dependency
符合性	Conformance
符合性聲明	Conformance statement
統一資源定址	Universal Resource Locator, URL
統一資源識別符	Universal Resource Identifier, URI
組織	Organization
項目	Item
遍歷	Traverse
詮釋資料	Metadata
資訊模型	Information Model
資源	Resource
標題	Title

學習管理系統	Learning Management System, LMS
檔案	File
繫結	Binding
識別符	Identifier