

CNMARC CHENGXU SHEJI

CNMARC 程序设计

薛 红 编著



西南交通大学出版社
[Http://press.swjtu.edu.cn](http://press.swjtu.edu.cn)

CNMARC 程序设计

薛 红 编著

西南交通大学出版社

· 成 都 ·

内容简介

CNMARC (中国机读目录格式) 用于中国国家书目机构同其他国家书目机构以及中国国内图书馆与情报部门之间, 以标准的计算机可读形式交换书目信息。本书对 CNMARC 的数据结构进行详细分析, 研究 CNMARC 程序设计的基本原理和方法, 包括读取 CNMARC 记录、操作 CNMARC 记录、对 CNMARC 记录的格式和内容进行有效性验证、读取 CNMARC 记录中的书目信息、CNMARC 记录与 XML 文档之间的无损转换等内容, 最后以一个 CNMARC AJAX 编辑器为例, 介绍了 CNMARC 编程的综合应用。

图书在版编目 (C I P) 数据

CNMARC 程序设计 / 薛红编著. —成都: 西南交通大学出版社, 2012.8

ISBN 978-7-5643-1873-4

I. ①C… II. ①薛… III. ①机器可读目录—程序设计 IV. ①G254.364②TP311

中国版本图书馆 CIP 数据核字 (2012) 第 185961 号

CNMARC 程序设计

薛 红 编著

责任编辑	张 波
封面设计	原谋书装
出版发行	西南交通大学出版社 (成都二环路北一段 111 号)
发行部电话	028-87600564 87600533
邮政编码	610031
网 址	http://press.swjtu.edu.cn
印 刷	四川锦祝印务有限公司
成品尺寸	185 mm × 260 mm
印 张	14.75
字 数	370 千字
版 次	2012 年 8 月第 1 版
印 次	2012 年 8 月第 1 次
书 号	ISBN 978-7-5643-1873-4
定 价	36.00 元

图书如有印装质量问题 本社负责退换
版权所有 盗版必究 举报电话: 028-87600562

前 言

作为图书馆自动化的基础之一，CNMARC 的重要性可谓不言而喻。自从 1992 年《中国机读目录通讯格式》正式出版，CNMARC 已经走过了 20 年的历程。这么多年以来，国内出版了大量关于 CNMARC 的使用手册，但是令人奇怪的是却没有出版过任何一部关于 CNMARC 程序设计的专著。

难道 CNMARC 程序设计只是少数人的专利？或者普通的图书馆工作人员不需要掌握这些知识？显而易见，答案是否定的。笔者在从事图书馆工作的时候，就经常地遇到需要自己编程解决类似于批量修改 CNMARC 记录中的数据的情况，常常苦恼于没有一本专门介绍 CNMARC 程序设计知识的书籍。为此，作者不断尝试开发一些小的应用程序来解决实际工作中遇到的这些问题。在学习和实践的过程中，我发现 CNMARC 程序设计并不需要多么高深的计算机编程知识，只要掌握 ISO 2709 和 CNMARC 的基本结构，并具有一定的编程能力，就可以很容易地开发出自己的 CNMARC 工具箱。

本书总结了作者在 CNMARC 程序设计中的点滴经验，从最基础的知识入手，循序渐进、由浅入深，一步一步地帮助读者学习和掌握 CNMARC 程序设计的理论和技巧。希望它能起到抛砖引玉的作用，让更多的人分享到更多的 CNMARC 程序设计方面的知识。

1. 本书的读者对象

本书的读者对象是希望能够自己进行 CNMARC 程序设计并以此解决工作中的实际问题并希望更深入地揭示 CNMARC 所蕴含的信息的程序开发人员。同时本书也面向于那些从事图书馆集成管理系统设计的开发人员，帮助他们更有效地处理 CNMARC 数据。

如果读者具有 CNMARC、Visual Studio、C#语言和 LINQ 的学习背景，将有助于对本书的阅读。但是这些不是必要的前提，具有其他语言学习背景的读者也能够从本书中获得 CNMARC 程序设计的相关技巧。

2. 本书的主要内容

第一章首先简要回顾了 MARC 的发展历程，然后介绍了为什么要学习 CNMARC 程序设计，最后讨论了选择的开发工具和开发语言。

第二章全面系统地分析了 ISO 2709 格式的各个组成部分，列举了 CNMARC 对 ISO 2709 的继承和发展，最后设计了一个 Visual Studio 解决方案。该解决方案包括一个 CNMARC 类库和一个测试项目，实现了解析 CNMARC 记录结构并重新组装 CNMARC 记录。

第三章对 CNMARC 记录进行深入编程，实现了自定义格式输出、提供了错误和异常处理机制，同时，我们还实现了 CNMARC 字段和子字段的查找、增加、更新和删除等功能。

第四章讨论了如何对 CNMARC 记录进行有效性验证，包括 CNMARC 记录的格式和内容两个方面。通过这章的学习，我们可以很方便地验证 CNMARC 记录的有效性，确保数据能够被相应的工具正确解析。

第五章提供了将 CNMARC 记录中的字符和数字的实际含义转换为能被用户理解的文字

的方法。CNMARC 记录包含大量的字符和数字，这些字符和数字都有自己明确的意义，但是只凭字面却难以理解。本章提供解决这个问题方法。

第六章探讨了 CNMARC 记录和 XML 文档间进行无损转换的方法。经过多年的实践，CNMARC 的弊端已经越来越多了。为此，业界探索了众多的解决之道，其中将 CNMARC 转换为 XML 文档是其中最主要的一种。本章利用 LINQ To XML 技术实现了二者之间的无损转换。

第七章综合运用前面章节的知识，开发了一个 CNMARC AJAX 编辑器。该编辑器在服务器端和客户端采用 JSON 进行数据交换，客户端采用不唐突的 JavaScript (Unobtrusive JavaScript) 开发模式，页面的结构、样式和行为相互分离，有效地屏蔽了浏览器的差异性，在 IE、Firefox 和 Chrome 等主流浏览器中都运行良好，使用 AJAX 技术，页面局部刷新且功能齐全，具有与桌面 MARC 编辑器相似的用户体验。

3. 本书的开发环境

Visual Studio 2010 作用为开发工具，C#作为开发语言。

4. 邮件支持

本书已经尽可能地将所有源代码展示到读者的面前。因此，在学习的过程中，读者可以采用手工输入的方式查看程序运行的效果，这也是笔者希望大家采用的方式，这样可以更好地学习和理解代码的含义。当然，如果读者已经熟练地掌握了 CNMARC 程序设计所需要的背景知识，希望能够更高效地运用这些代码，可以通过电子邮件的方式向读者索取本书所有的源代码

同时，限于作者自身的水平和相关技术的不断发展，本书难免有一些错漏和疏忽之处，本人欢迎读者通过电子邮件的方式反馈对本书的看法。任何批评和建议都是对本人的鞭策和鼓励。

本人的电子邮件的地址是：xuehong@suse.edu.cn。

作 者

2012 年 5 月

目 录

第 1 章 绪 论	1
1.1 MARC 发展简要回顾	1
1.2 CNMARC 程序设计的意义	3
1.3 编程语言与开发工具	4
第 2 章 读取 CNMARC 记录	6
2.1 ISO 2709 的相关规定	6
2.2 中国机读目录格式	10
2.3 解决方案	11
第 3 章 操作 CNMARC 记录	48
3.1 格式化输出	48
3.2 错误和异常处理	54
3.3 CNMARC 数据处理	58
第 4 章 有效性验证	74
4.1 验证 CNMARC 记录结构	74
4.2 验证 CNMARC 记录内容	83
4.3 方法测试	96
第 5 章 读取 CNMARC 记录信息	100
5.1 CNMARC 记录通用处理信息	101
5.2 CNMARC 通用编目项目	116
5.3 编码信息块（除去 100 字段）	119
第 6 章 CNMARC 与 XML	143
6.1 CNMARC XML Schema	143
6.2 CNMARC 转换为 XML 文档	151
6.3 XML 文档转换为 CNMARC 记录	160
第 7 章 CNMARC AJAX 编辑器	169
7.1 AJAX 技术和 JSON	169
7.2 创建 ASP.NET MVC3 应用程序	171
7.3 页面布局	173
7.4 创建数据库	178
7.5 导入数据	180

7.6 检索数据	198
7.7 显示 CNMARC 记录详细信息	201
7.8 编辑 CNMARC 记录	207
7.9 创建新 CNMARC 记录	216
7.10 保存 CNMARC 记录	220
7.11 尾 声	228
参考文献	230

第1章 绪论

随着图书馆自动化建设的不断深入,我们的日常业务工作也越来越离不开机读目录格式(Machine Readable Cataloging, 以下简称为 MARC)了。我们对文献的加工、整理、揭示和报导、检索和利用都是在 MARC 的基础上开展起来的。虽然所有的图书馆集成管理系统都提供了解析、编辑和批处理 MARC 数据的工具。但由于这些工具本身功能的局限性和我们需求的不可预测性,在实际的应用场景中,还是会有大量的需要我们自己来处理 MARC 数据的情形出现。本书的目的就是要对 CNMARC 的格式进行详细分析,介绍 CNMARC 程序设计的基本原理和方法,帮助我们开发自己的 CNMARC 工具箱。

1.1 MARC 发展简要回顾

20 世纪 50 年代末,计算机从科学计算逐渐扩展到数据处理、事务处理及机构的信息管理,应用计算机成为一种新的发展方向,LC(美国国会图书馆)已敏锐地觉察到了这项技术必将对图书馆的发展带来前所未有的变革。与此同时,LC 开展了一项关于在其内部业务操作中使用自动化技术的可行性调查。该调查由 G. W. King 等人主持,并于 1963 年正式发表调查报告。报告建议设立一个专门小组来研究编目、检索、索引和文献查找等领域的自动化技术,报告同时认为:国会图书馆的书目系统十年内能够实现自动化。1965 年,又制定了一个将目录数据转换成标准的机读目录的报告,并从图书馆资源委员会处得到资助,发展这一项目。经历三年的努力研究,LC 于 1966 年研制成功了 MARC I 机读目录系统,并开始实施机读目录(MARC)试验计划,磁带书目记录定期被发放到参与试验的 16 个图书馆。1966 年 11 月至 1968 年 6 月,该试验工程结束。针对试验中出现的一系列问题,LC 与英国国家书目机构密切合作,在 MARC I 的基础上,研制成功了 MARC II 机读目录系统。

在 LC 研究 MARC II 机读目录的同时,美国国内一些大学图书馆及英、德、法等国的图书馆也陆续开始研究计算机在图书馆的应用工作。但在 1968 年 MARC II 推广以后,这些国家的图书馆把自己的研究成果与 MARC II 系统进行了比较,都肯定了 MARC II 机读目录的长处,所以,英、澳、加等英语国家图书馆很快就接受了 MARC II 的编目系统。虽然所有称“MARC”的格式都符合 ISO 2709 标准,但在数据元素标识和内容选取上,即便是亲缘关系最近的 LC MARC(即 USMARC)和英国的 BNB MARC(即 UKMARC)之间也存在显著的差别,从而使全球范围的书目数据交换成为一个非常严重的问题。20 世纪 70 年代初期,

国际编目界开始讨论开发一种超级机读书目格式 (SUPERMARC) 的可能性。这种书目格式作为各种书目格式交换的媒介, 可以节省财力和人力, 并进一步实现“全球书目控制” (UBC) 理念。

1972 年国际图联 (IFLA) 布达佩斯年会上, IFLA 编目委员会和自动化委员会共同成立了内容标识符工作组 (IFLA Working Group on Content Designators), 由 MARC 设计者艾弗拉姆担任主席。工作组的任务是制定一套标准的内容标识符, 适用于所有类型的资源, 使机读书目数据能够在不同机构的 MARC 格式之间进行交换。工作组意识到各国 MARC 格式存在严重的差别, 所以它制定了建立国际标准的 9 条基本原则。UNIMARC (通用机读目录格式) 就是依据这些共同原则编制的, 并能为绝大多数机构接受的国际机读格式, 《UNIMARC: 通用 MARC 格式》一书于 1977 年出版。许多图书馆以它作为交换书目记录的标准, 各国也可以 UNIMARC 为范本, 建立自己的机读目录格式, CNMARC (中国机读目录格式) 就是采取这种方式编成的。

关于 CNMARC 的确立, 我国图书馆界进行了较长时间的研究和探索。1975 年, 刘国钧先生发表了《马尔克计划简介》一文, 第一次向我国图书馆界较为全面地介绍了美国国会图书馆研制、发行的西文图书机读目录——MARC。后来, 他又发表了《用电子计算机编制图书目录的几个问题》一文, 提出了在我国建立机读目录的设想。刘国钧先生的工作, 对我国西文图书机读目录的研制起到了先驱的作用。1978 年, 朱南同志在北京图书馆第二次科学讨论会上, 发表了《利用 MARC II 机读目录系统建立书目数据库共享情报图书资源的探讨》, 首次提出引进 MARC 磁带, 用以建立我国西文图书书目数据库, 存储全国情报图书资源馆藏数据的设想, 引起图书馆界的广泛注意。1978 年初, 南京大学数学系情报检索教研室首先进行了编制西文图书机读目录的尝试, 他们仿照 MARC II 款式编辑书目数据, 试验输入了几百条西文图书的书目记录, 第一次用计算机打印成书本目录。1979 年南京大学建立了南大图书馆系统 (NDTS), 采用简化了的 MARC 格式, 建立了一万余条数据的西文书目数据库。1979 年底, 北京图书馆、中国科学院图书馆、北京大学、清华大学、中国人民大学和中国图书进出口公司六单位的有关同志, 经过多次协商, 决定成立“北京地区研究试验西文图书机读目录协作组”, 协作组对引进的 LCMARC 磁带进行了试验, 并研制了一个模拟系统, 为深入研究 MARC 格式、制定国家标准和发行中国机读目录, 在理论和实践方面积累了宝贵的经验。从此, 我国图书馆界研制机读目录的事业进入了新的阶段。

1982 年 8 月, 中国台湾地区完成了《中文机读目录格式》的定稿工作。台湾《中文机读目录格式》正式出版以后, 对大陆图书馆影响很大。1986 年 12 月, 北京图书馆自动化发展部提出了《中国机读目录通讯格式》初稿, 经过广泛征求各方面意见, 1989 年 12 月, 在全国各方面专家参加的会议上, 初步定稿为 CNMARC。1992 年 2 月, 《中国机读目录通讯格式》(CNMARC) 正式出版。《中国机读目录通讯格式》是在国际机读目录格式 (UNIMARC 1980 版) 的基础上, 根据《文献著录总则》(GB 3792.1—83)、《普通图书著录规则》(GB 3792.2—83)、《连续出版物著录规则》(GB 3792.3—85) 和《中文书刊名称汉语拼音写法》(GB 3259—82) 及其他有关国家、国际标准编制而成, 供中国国家书目机构以计算机可读形式同其他国家书目机构之间进行书目信息交换。《中国机读目录通讯格式》的生成, 是我国计算机自动化中文信息处理技术的成功, 标志着我国图书馆自动化技术又向前迈进了一大步, 为我国机读目录实现标准化及与国际接轨, 从数据结构方面提供了坚实的保障。1995 年, 《中国机读

目录格式使用手册》一书由华艺出版社出版。1996年,《中国机读目录格式》作为文化行业标准(WH/T0503/96)开始实施。与此同时,国家技术监督局通过文化部下达国家图书馆研制机读目录格式的国家标准。经过近8年时间,国家标准《中国机读目录格式》已经通过文化部组织的专家鉴定。期间,科学技术文献出版社于2001年出版了《中国机读目录格式使用手册》的修订版;2004年,根据国家标准编写的《新版中国机读目录格式使用手册》由北京图书馆出版社出版发行。

1.2 CNMARC 程序设计的意义

1. CNMARC 批量处理的需要

不仅国内所有的图书馆集成管理系统都内置了 CNMARC 编辑工具,而且网络上还存在着许多具有类似功能的 CNMARC 编辑工具,它们可以逐条创建、修改、编辑 CNMARC 记录。那么,我们为什么还需要学习和研究 CNMARC 编程呢?为什么还要开发自己的 CNMARC 处理软件呢?主要原因在于我们需要批量处理 CNMARC 记录。批量处理 CNMARC 记录是国内每个图书馆随时都可能遇到的问题。比如前几年,随书光盘的管理和利用成为了图书馆研究的热点。很多图书馆尝试开发自己的随书光盘管理系统,他们普遍采用的方式是利用 CNMARC 格式中的 856 字段(电子资源地址与检索),将随书光盘的 ISO 文件下载地址著录到 856 字段的相应子字段中,然后读者可以通过公共查询系统(Online Public Access Catalogue,以下简称 OPAC)进行检索和下载。这种方式就可能涉及批量处理 CNMARC 记录。比如批量为含有随书光盘的图书的 CNMARC 记录批量添加 856 字段,提供文件下载地址;或者当服务器地址改变时,我们需要批量修改 856 字段中相应的数据等等。如果我们不具备 CNMARC 编程能力,就只能逐条修改记录。这将极大地降低我们的工作效率,浪费大量的人力资源。CNMARC 编程可以为我们处理 CNMARC 数据提供最大的自由度和灵活性。

2. 全方位、多角度、深层次揭示文献特征的需要

文献著录的过程是对文献实体进行描述,对文献内容和形式特征进行分析、选择和记录的过程。在传统手工编目年代,文献著录结果是款目。采用计算机编目,著录的结果就是 MARC 记录。MARC 记录蕴含着大量的信息,不仅包含有传统目录款目的内容,而且还包含了大量关于记录本身的信息。机器可读而非人可读是 MARC 的设计者最初就已制定的原则。因此如果不借助工具,任何人在面对一条 CNMARC 记录的时候都会感到十分困惑。将一个个数字、字母或其他文字组成的记录与实际的文献特征对应起来简直就是一个不可能完成的任务。不依靠计算机而采用手工方式处理 CNMARC 记录无异于天方夜谭。例如,“20070326d2007 em y0chiy50 ea”是一条 CNMARC 记录的 100 字段(通用处理数据)的 \$a 子字段的数据。这区区 36 个字符蕴含了丰富的信息。这些信息包含记录的入档时间是 2007 年 3 月 26 日;文献的出版时间类型是一次或一个年度内出版的专著,出版时间是 2007 年;该著作合适的阅读人群是青年(14~20 岁)和普通成人;文献为非政府出版物;现有字

符集可以满足转录出版物文字数据的需求,记录无变更;编目语种为中文;未使用音译方案;用于记录交换中的主要图形字符集为 ISO 10646 Level 3 (Unicode 统一编码字符集);文献正题名使用的文字是广义中文。但是,在 OPAC 中,100 字段\$a 子字段中的数据只是简简单单地显示在电脑上,这么多的信息却没有任何文字作任何描述。我们要通过 CNMARC 编程将记录中的信息全方位、多角度、深层次地揭示出来,展现到读者面前。

3. CNMARC 发展的需要

经过 40 多年的发展和应用,机读目录的缺陷已经越来越明显,傅立云、刘新总结了 MARC 格式中存在的一些问题:① 数据不利于人们理解;② 字段设置过于复杂且个别字段存在重复,降低了编目效率;③ 某些字段的逻辑意义缺乏统一规定,各馆的理解不同导致数据著录的不一致;④ 三段式格式造成直接检索速度太慢,另外设置索引又增加了数据冗余,不易维护;⑤ 以字节为单位存储,既浪费存储空间,又降低了处理的速度;⑥ 无法通过因特网上的通用搜索引擎来检索。虽然机读目录存在着这样那样的问题,近 20 余年来一直受到业界的质疑,LC 等图书馆致力于用 XML 替代 MARC 的研究,编目精灵等人甚至提出了要让机读目录“安乐死”。但是,无数图书馆 40 余年来累积的 MARC 数据是日常业务工作的基石,不可能在一夜之间就消亡。RDA (资源描述和检索)对 MARC 格式的支持显示了机读目录的生命力仍然十分旺盛,还会存在相当长的一段时间。不论 MARC 的未来究竟会怎么样,都是建立在广大图书馆工作人员的理论研究和编目实践的基础上的。对 CNMARC 编程的学习和研究,有助于我们加深对 CNMARC 格式和字段设置的进一步理解,帮助我们把握 CNMARC 格式的发展方向,加快 CNMARC 的发展步伐。

1.3 编程语言与开发工具

1. 开发语言

不论是选择 C、C++、JAVA,还是 VB.NET、Delphi、Visual C#,任何高级语言都完全能够胜任 CNMARC 编程的开发语言。本书采用的开发语言是 Visual C#。Visual C#是微软公司 2000 年 6 月发布的一种基于现代面向对象设计方法的语言,可以用来建立运行在 .NET Framework 上的应用程序。经过 10 余年的发展,Visual C#已经逐渐被大家所接受,受到了越来越多的开发人员的青睐。Visual C#具有以下优点:① 语法表现力强,而且简单易学。任何熟悉 C、C++或 Java 的开发人员通常在很短的时间内就可以开始使用 C#高效地进行工作;② .NET Framework 的类库中的种类繁多、内容全面,为从文件输入和输出、字符串操作、XML 分析到 Windows 窗体控件的所有内容提供了各种有用的功能,C#可以方便地调用库中这些内容;③ 简单易学,降低了编程爱好者的学习门槛;④ 功能强大。Visual C#支持泛型、lambda 表达式、匿名函数、LINQ (Language Integrated Query,语言集成查询)、逆变和协变、动态支持等一系列高级功能。并且,C#允许有限制地使用纯指针(Native Pointer)和“不安全”代码,使程序员可以进行系统底层开发。另外,C#支持多线程编程和分布式编程;⑤ .NET

Framework 的公共运行时 (CLR) 使 C# 实现了语言互操作性, 可以很方便地调用其他语言编写的能被 .NET Framework 托管的其他类型和实例; ⑥ 强类型机制使代码更加安全、健壮; ⑦ 资源回收机制降低了开发人员管理内存的难度; ⑧ 面向对象的编程模式, 支持继承、封装和多态; ⑨ 功能强大的开发工具。Visual Studio 为快速、高效开发 Visual C# 程序提供了强有力的保障。

2. 开发工具

本书采用的开发工具是 Visual Studio 2010, 其开发代号为 “Hawaii”。Visual Studio 是微软公司推出的应用程序集成开发环境 (Integrated Development Environment, IDE)。早在 1997 年, 微软就发布了 Visual Studio 1997, 致力于使用开发人员能够通过一个完整的开发周期开共享和管理大型项目。次年 8 月, 微软发布了经典的 Visual Studio 6.0, 使开发人员可以创建基于组件的方案, 为分布式异构体系结构提供了综合性解决方案。2002 年, 随着微软的战略重心向 .NET 转移, 适时发布了与之配合的 Visual Studio .NET (2002)。它采用了全新的托管代码编程模型。程序不再被直接编译成机器语言, 而是被编译成了中间语言 (IL)。当公共语言运行时 (CLR) 加载了 IL 后, 即时编译器 (JIT Compiler) 再将它们转换为机器码并交由 CPU 执行。2003 年, 微软发布了 Visual Studio .NET 2003, 集成了 .NET Compact Framework, 增加了对移动和嵌入式设备。2005 年末, 微软发布了 .NET Framework 2.0 和 Visual Studio 2005。它加入了团队开发的功能, 能轻松实现所有团队成员之间的协作、缩短开发时间并提高开发过程的可预见性和可靠性。2008 年 2 月 1 日, Visual Studio 2008 上市。它是面向 Windows Vista、Office 2007、Web 2.0 的开发工具, 引入了 250 多个新特性, 支持 .NET Framework 2.0/3.0/3.5。2010 年 4 月 12 日, Visual Studio 2010 正式上市。它为开发人员提供了设计、开发、调试、测试、部署和管理的一体化解决方案, 是一个有助于简化从设计到部署等整个开发流程的集成环境。Visual Studio 2010 使用 WPF (Windows Presentation Foundation) 技术重新设计和组织了 IDE, 提高了可读性。多显示器支持、垂直代码块、即时搜索、调用层次结构、测试先行、应用程序生命周期管理、数据访问和建模、并行编程、云计算支持等等, 这些新功能、新特性只不过是 Visual Studio 2010 亮点中的很少一部分, 还有很多可以添加到这个长长的列表中。开发 Windows 平台应用程序, Visual Studio 2010 必定是我们的首选。

第 2 章 读取 CNMARC 记录

CNMARC 编程的首要任务是能够正确地解析出 CNMARC 记录的各个组成部分,包括记录头标、地址目次区、字段标识符、字段指示符和子字段。这既是本章的主要内容,也是本书的基础。为了完成这个任务,我们首先要了解 CNMARC 记录遵循的国际、国内标准,并在此基础上进行编程。

2.1 ISO 2709 的相关规定

ISO 2709 (信息和文献——信息交换格式, Information and documentation — Format for information exchange) 是国际标准组织 (International Organization for Standardization, 简称 ISO) 制定的所有资源的书目数据和其他记录的通用交换格式,是 CNMARC 记录的框架格式。

2.1.1 引用标准

ISO/IEC 646: 1991, 信息技术—ISO 信息交换 7 位编码字集;
ISO/IEC 6429: 1992, 信息技术—编码字符集用控制函数;
ISO/IEC 10646: 2003, 信息技术—通用多 8 位编码字符集, 亦称大字符集。

2.1.2 术 语

字符 (character): 用来组织、控制或表示数据的元素集合。

数据字段 (data field) 描述记录内容的字符串。

地址目次区 (directory): 记录内字段地址的索引。

地址目次区项目结构 (directory map): 规定地址目次区款目结构的一组参数。

字段 (field): 记录的变长部分, 包括一个特别的数据类别, 位置在地址目次区的后面, 并且与地址目次区中的一个款目相关。一个字段可以包含一个或多个子字段。

字段分隔符 (field separator): 控制字符, 用于地址目次区、记录结束符、参考字段和数据字段的末尾。

字段标识符 (tag): 用于标识字段的一组 3 字节符号。

字段指示符 (indicator): 如果使用, 则其第 1 位用来提供或关于字段内容的进一步信息, 或关于本字段与记录内其他字段关联的信息, 或关于在数据处理等方面所需的操作等。

8 位组 (octet): 连续的 8 位一组的二进制数码; 它既可能代表一个字符 (ISO/IEC 646 中的字符), 也可能是字符 (ISO/IEC 10646 中的字符) 的一部分。

记录 (record): 字段的集合, 包括 1 个记录头标、1 个地址目次区和数据。

记录标识号字段 (record identifier field): 标识记录的字符串。

记录头标 (record label): 出现在每条记录的开始部分, 为处理记录提供参数。

记录结束符 (record separator): 记录末尾的控制字符。

参考字段 (reference field): 处理记录时需要的数据。

分隔字符 (separating character): 用来分隔和标识数据逻辑单元的控制字符, 并且在某些情况下分层。

结构 (structure): 记录组成部分的排列顺序。

子字段 (subfield): 字段的一部分, 包含一个已定义的信息单元。

子字段标识符 (identifier): 在一个或多个字符串数字元素的前面, 用来标识子字段。

子记录 (subrecord): 一条记录中的字段组, 该记录可以被视为一个实体。

2.1.3 记录结构

ISO 2709 规定记录由以下四部分组成:

记录头标 Record label	地址目次区 Directory	字段 Fields	记录结束符 Record separator
----------------------	--------------------	--------------	---------------------------

其中记录头标为固定长度, 地址目次区和字段的长度都是可变的。按字段标识符的取值范围, 字段又分为记录标识号字段、参考字段和数据字段并依次排列。地址目次区和字段都有一个字段分隔符作为结束标志, 记录以记录结束符为结束标志。字段分隔符 IS2 和记录结束符 IS3 应该与 ISO/IEC 646 或 ISO/IEC 6429 相一致。

2.1.4 基本字符编码

记录头标、地址目次区、指示符、子字段标识符、字段分隔符和记录结束符中的所有数据都应使用 ISO/IEC 646 字符集或者使用 UTF-8 编码的 ISO 10646。

1. 记录头标

记录头标由固定长度为 24 位的八位组字节组成, 具体定义如下:

(1) 记录长度 (Record length, 字符位置 0~4)

记录长度是记录头标、地址目次区、字段和记录结束符长度的总和，由一个 5 位的十进制数字表示，右对齐，如果位数不够则用数字 0（零）填充。

(2) 记录状态 (Record status, 字符位置 5)

记录状态由一位字符表示，描述记录的状态，如：新记录或修改的记录。

(3) 执行代码 (Implementation codes, 字符位置 6~9)

ISO 2709 未定义执行代码，由交换书目数据的机构自行定义。

(4) 指示符长度 (Indicator length, 字符位置 10)

指示符长度由 1 位十进制数字表示，如果未使用指示符，则指示符长度设为 0（零）。

(5) 子字段标识符长度 (Identifier length, 字符位置 11)

子字段标识符长度由 1 位十进制数字表示，它的第一位或仅有一位必须使用 ISO/IEC 646 或 ISO/IEC 6429 的分隔符 IS1。如果不使用子字段分隔符，则其长度设为 0（零）。

(6) 数据基地址 (Base address of data, 字符位置 12~16)

由 5 位十进制数字表示，右对齐，不足的位由数字 0（零）填充，等于记录头标的长度与地址目次区长度（包括其末尾的字段分隔符）之和。

(7) 用户系统自定义 (Positions defined by user systems, 字符位置 17~19)

由用户系统自行定义。

(8) 地址目次区项目结构 (Directory map, 字符位置 20~23)

其结构如下：

① 字符位置 20：1 位十进制数字，定义地址目次区每个款目“字段长度”部分的字符串长度；

② 字符位置 21：1 位十进制数字，定义地址目次区每个款目“起始字符位置”部分的字符串长度；

③ 字符位置 22：1 位十进制数字，定义地址目次区每个款目“执行定义”部分的字符串长度；

④ 字符位置 23：为将来使用保留。

2. 地址目次区 (Directory)

地址目次区由不同数量的款目组成，每个款目与一个字段相对应。地址目次区以字段分隔符为结束标志。

(1) 目次项 (Directory entry)

目次项由以下部分并按该顺序组成：

① 字段标识符；

② 该字段所包含的字符数，即字段长度；

③ 起始字符位置；

④ 定义的执行代码部分。

字段标识符的长度为三位。目次项的其他三部分在地址目次区项目结构中定义，即记录头标的第 20~22 位。地址目次区所有目次项具有相同的结构。

(2) 字段标识符 (Tag)

字段标识符由三位字符组成，其含义应该由一个执行中的国际标准定义。如果没有该标准，则由交换书目数据的机构相互间约定。

(3) 字段长度 (Length of field)

字段长度应该是以下三种情形中的一种:

- ① 以单字节计算字段所包含的字符数 (含指示符和字段分隔符), 或者
- ② 0 (零), 表示字段的长度大于地址目次区目次项所能表示的最大长度, 或者
- ③ 情形 b 描述的字段的最后部分 (含字段指示符) 以单字节计算的字符数。

(4) 起始字符位置 (Starting character position)

起始字符位置, 用一个十进制数字来表示该字段第一字符在字段区中的位置, 相当于数据的基地址 (也就是说, 紧随着地址目次区的第一个字段的起始字符位置为 0 (零))。

(5) 定义的执行代码部分 (Implementation-defined part)

如果存在该部分, 则它包含了被款目参考的相关控制信息。

3. 字段 (Fields)

字段的末尾是一个字段分隔符。有以下三种类型的字段:

记录标识号字段 (record identifier field): 字段标识符为 001 (符号“0”为零);

参考字段 (reference fields): 字段标识符为 002 ~ 009 和 00A ~ 00Z;

数据字段 (data fields): 字段标识符为 010 ~ 999 和 0AA ~ ZZZ。

(1) 记录标识号字段

记录标识号字段包含识别记录的字符串并且由创建记录的机构指定。该字段无字段指示符或子字段标识符。

(2) 参考字段

参考字段为处理记录提供必备的数据。该字段无字段指示符或子字段标识符。

(3) 数据字段

每个数据字段由字段指示符 (可选)、子字段标识符 (可选)、数据和字段分隔符组成。字段指示符、子字段标识符在记录头标中定义 (字符位置分别是 10 和 11)。它们的定义不同, 也就有以下四种类型的数据字段:

- ① 当字段指示符和子字段标识符的长度都为 0 时, 如下:

数据	字段分隔符	数据	字段分隔符	数据	字段分隔符	
----	-------	----	-------	----	-------	-------

- ② 当字段指示符为 0, 子字段标识符长度>0 时, 如下:

子字段标识符	数据		子字段标识符	数据	字段分隔符	子字段标识符	数据		子字段标识符	数据	字段分隔符	
--------	----	-------	--------	----	-------	--------	----	-------	--------	----	-------	-------

- ③ 当字段指示符>0, 子字段标识符长度为 0 时, 如下:

字段指示符	数据	字段分隔符	字段指示符	数据	字段分隔符	字段指示符	数据	字段分隔符	
-------	----	-------	-------	----	-------	-------	----	-------	-------

- ④ 当字段指示符>0, 子字段标识符长度>0 时, 如下:

字段指示符	子字段标识符	数据		子字段标识符	数据	字段分隔符	字段指示符	子字段标识符	数据		子字段标识符	数据	字段分隔符	
-------	--------	----	-------	--------	----	-------	-------	--------	----	-------	--------	----	-------	-------

2.2 中国机读目录格式

1996 年, 文化部颁布了文化行业标准《中国机读目录格式》(China MARC Format, 以下简称为 CNMARC 格式), 2004 年 CNMARC 格式成为了国家标准。它是 ISO 2709 的一个特定形式, 具体化了 ISO 2709 的有关规定。我国每一条用于交换的书目记录都必须遵循它所制订的标准记录结构。根据北京图书馆出版社 2004 年 3 月出版的《新版中国机读目录格式使用手册》(以下简称使用手册), CNMARC 记录的格式主要有以下变化:

1. 引用标准

GB 1988 信息处理 信息交换用七位编码字符集(等效采用 ISO 646)

GB 2260 中华人民共和国行政区划代码

GB 2312 信息交换用汉字编码字符集 基本集

GB/T2659 世界各国和地区名称代码(等效采用 ISO 3166)

GB/T 2901 文献工作——书目信息交换用磁带格式(等同采用 ISO2709)

GB 3100 国际单位制及其应用(等效采用 ISO 1000)

GB 3101 有关量、单位和符号的一般原则(等效采用 ISO 31—0)

GB 3102 (所有部分)量和单位(等效采用 ISO 31(所有部分))

GB 3469 文献类型与文献载体代码

GB 3792 文献著录规则(系列)(参照采用 ISBD 系列)

GB/T 4880 语种名称代码(等效采用 ISO 639)

GB/T 7408 数据元和交换格式 信息交换 日期和时间表示法(等效采用 ISO 8601)

GB 12406 表示货币和资金的代码(等效采用 ISO 4217)

GB 13000.1 信息技术 通用多八位编码字符集(UCS)第一部分: 体系结构与基本多文种平面(等同采用 ISO/IEC 10646.1)

UNIMARC Manual: Bibliographic Format

2. 在 ISO 2709 的基础的所作的具体化规定

(1) 字段标识符

格式具体化了每个字段标识符的含义和取值范围。如: 001 代表记录标识号、005 代表记录处理时间标识、010 代表国际标准书号等。

(2) 指示符长度

格式规定记录指示符的长度为 2 个 8 位字符。

(3) 子字段标识符长度

格式规定子字段标识符长度为 2 个 8 位字符。

(4) 记录头标

① 记录状态(字符位置 5)。CNMARC 使用 5 种字符代表记录的处理状态。

② 执行代码(字符位置 6~9)。CNMARC 规定字符位置 6 代表记录类型、字符位置 7