The abc of ABCD Central : the Reference Manual

Version 1.3t

Egbert de Smet

The abc of ABCD Central : the Reference Manual: Version 1.3t

Egbert de Smet

Some words of gratitude : this document would not have been possible without the invaluable contributions to both the software and the manual by mrs. Guilda Ascencio (Venezuela) and Mr. Ernesto Spinak (Uruguay).

Publication date December 3, 2013

Abstract

This document aims at providing all relevant background information and instructions on how to use the integrated library and documentation management system 'ABCD'. Both the basic operations and the advanced management of the 'Central module' of the software are discussed, as are useful and necessary topics concerning the ISIS-software on which ABCD is based.

Chapter 1. Introduction

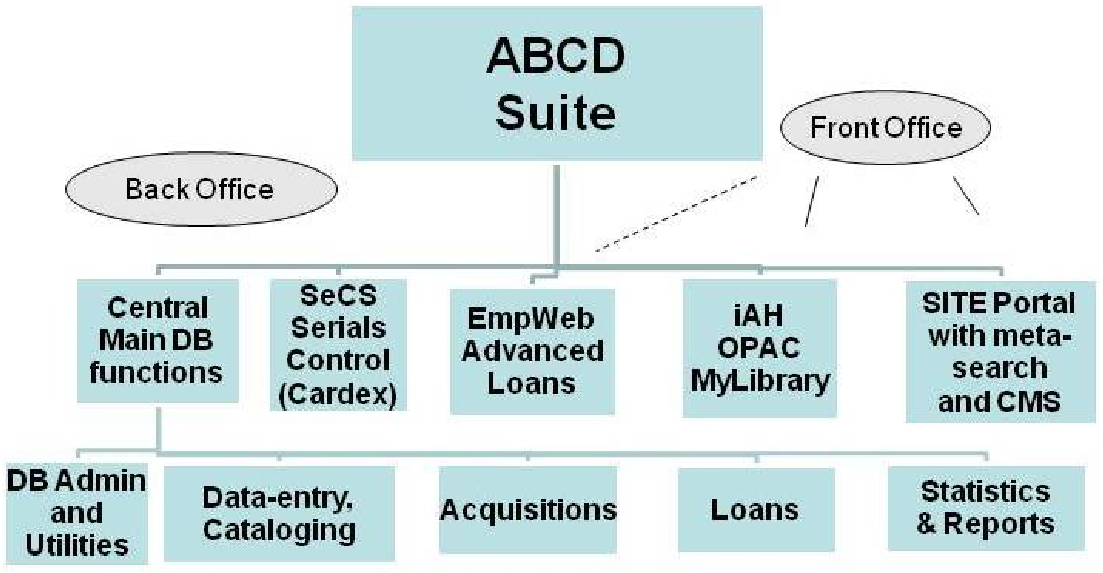
1. Background information

1.1. General introduction to ABCD as a software suite

ABCD is the acronym for a software suite for the automation of libraries and documentation centres. In Spanish this is, in full : '**A**utomatisación de **B**ibliotécas y **C**entros de **D**ocumentación', which keeps the same acronym valid also for French (**A**utomation des **B**ibliothèques et **C**entres de ***D***ocumentacion) or Portugese (**A**utomatização das **B**ibliotecas e dos ***C***entros de **D**ocumentação). Even in other non-latin languages, with some slight but quite acceptable variations, - e.g. Dutch : '**A**utomatisering van **B**ibliotheken en **C**entra voor **D**ocumentatie' - the acronym can still be maintained.

The name itself already expresses the ambition of the software suite : not only providing automation functions for the 'classic' libraries but also other information providers such as documentation centres. Flexibility and versatility are at the forefront of the criteria on which the software is developed. This flexibility e.g. is illustrated by the fact that in principle, but also practically, any bibliographic structure can be managed by the software, or even created by itself. Even non-bibliographic structures can be created, as long as the information is mainly 'textual' information, as this is the limitation put by the underlying database technology, which is the (CDS/)ISIS textual database. Good understanding of some basic ISIS-related concepts and techniques, e.g. the Formatting Language, is crucial for full mastering of the ABCD-software. For this reason some sections of this Manual will also deal with the underlying ISIS-technology.

ABCD is called a 'suite' of softwares for library and documentation centres automation because it exists of some relatively independent modules, which can fully co-operate but also can exist without each other. In fact some existing advanced softwares, mostly having already shown their potential in demanding environments in BIREME-applications (within the Virtual Health Library context), were adopted and adapted into ABCD - that is why the original names such as iAH, SeCS (both developed by BIREME) and EmpWeb (Empréstimos en Web) developed originally by KALIO ltda. of Uruguay and amply tested in Chili) are maintained. These main parts are shown, with their hierarchical relationships, at the second level in the following picture and subsequently discussed briefly :



1. **the ABCD Central**

   The 'Central' module of ABCD comprises modules for Database Administration (creation of databases, editing of database-structures, database utilities), Cataloging, Acquisitions, Circulation/Loans and Statistics. A thesaurus management module is also being prepared as part of the cataloging module for a specific thesaurus-structure database with consistency control of the hierarchical levels. As part of this 'central module' we can also like to mention import- and export services, printing and database-tools like blocking/unblocking and 'global changes' to fields in records. If the (new) modules of 'Document Supply' and 'Digital Library' are installed, the librarian or system manager can also create supply requests and digital documents here. In fact any database can be fully 'managed' (edited, searched, formatted...) from this central 'hub'. This 'Central' part in fact represents the 'back-office' part of ABCD, end-users will not be confronted with this but what they will see and be offered is fully defined in this central management part of the software !

Any ISIS-database structure can be defined and managed, with currently records of 1Mb maximum size and 4Gb max, databases (but these restrictions will be made obsolete by the NBP-based next generation of ISIS and ABCD). As compared to 'normal' ISIS-technology ,60-character (as compared to 30-character) indexing keys are used, there are much stronger authority control features available (picklists based on tables or authority databases such as thesauri) at the data-entry stage with flexible validation formats) and all interaction is based on WWW-technology of course, allowing e.g. HTML-coded text-strings for full-text indexing, hyperlinks to help-pages etc.

It is perfectly possible to fully automate a smaller library with mostly internal users with all necessary functions, only using this Central part, as e.g. an advanced searching option is built-in, so that all functions are covered with a minimum of technological complexity (i.e. only ISIS and PHP).

2. t**he ABCD OPAC** (iAH)

   The public search interface (OPAC) is an adapted version of BIREME's general 'advanced interface for Health information' (iAH). It has all the 'classic' elements of online database searching through either a very simple 'Google'-like interface or more advanced ways of searching with field-selectors and Boolean operators.

   The iAH interface developed by BIREME is currently being upgraded to iAHx, ensuring it will align perfectly with modern Information Retrieval concepts and techniques (e.g. clustering, relevance ranking based on Lucene indexing).

3. **the ABCD Site**

   The search function is offered as part of an 'end-users' portal page, presenting the own catalog(s) in a much wider information context by providing access to other information resource.s (e.g. Google, Medline...) and communication (announcements, alerts), also paving the way for 'Web 2.0'-like functions. It allows meta-searches on not only the local catalogs but also many other information resources. A link to the dedicated OPAC's for each resource is also available.

   The Site Administrator actually is a specific Content Management System which allows designing the structure and components of the portal page of ABCD.

4. **the ABCD Serials Control System** (SeCS)

   This module offers an advanced management tool for serials/journals (classical and/or electronic) of any publication type (referring to periodicity). Serials as such but also issues of a serial and all types of publication patterns can be managed by this module. BIREME uses this technology e.g. for its products 'Portal of Scientific Journals' (see : http://portal.revistas.bvs.br/main.php?home=true&lang=en) and SCAD (see : http://scad.bvs.br/php/index.php?lang=en) which is the Brazilian union catalog of over 12.000 journals (with millions of issues) of more than 50 libraries.

5. **the ABCD Advanced Loans module** (EmpWeb)

   This module offers advanced loans management with some more extra features for larger and more complex organisations. It offers a 'MyLibrary' function to end-users through the OPAC and is based on 'web-services' technology. It can be used to replace the integrated loans module of ABCD in case of a need to cope with multi-branch/policy and very high transactions volume situations.

The 'suite' idea reflects the fact that ABCD has relatively independent parts - as is the case with office automation suites (e.g. Open Office, Microsoft Office) - but with obvious links to co-operate. The Statistics module e.g., as part of the Circulation/Loans module, can work on any ISIS-database, while the iAH-OPAC also can offer advanced web-based retrieval in any (set of) ISIS-databases, not only ABCD-maintained ones. The Serials Control System (SeCS) manages ISIS-databases for serials within or outside the ABCD-context. But together, we believe, these parts constitute a very powerful suite of tools, and as an integrated system we hope 'the sum is more than just the parts added up' !

1.2. The ISIS software family (history and overview)

In this paragraph we want to briefly introduce the wider 'ISIS software family' to which ABCD belongs. As with all 'families', members share a lot of characteristics but not all.

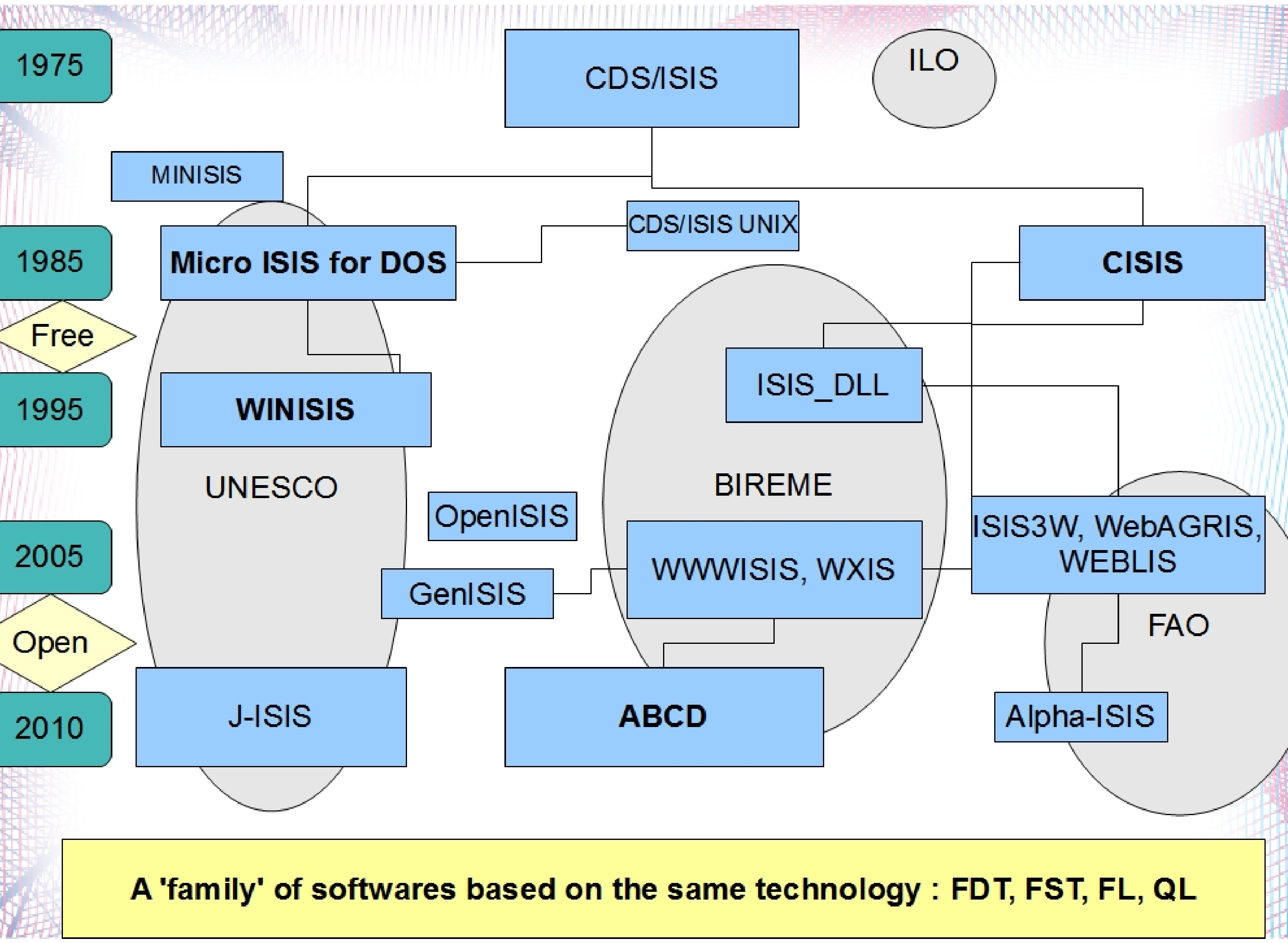
The common characteristics of the ISIS family relate to the way how information (of textual nature) is stored and managed by putting it in repeatable fields of variable length with the possibility of subdividing fields into subfields. Fields are in f act couples of a field-ID (a 'tag') combined with a field-value (a text, or in newer ISIS generation, any object, like e.g. 'binary large objects' or blobs). This 'No-SQL' database approach has become quite popular nowadays with the WWW (e.g. Google's BigTable database) and many new similar implementations are being produced, see http://www.nosql.org . But chances are good that ISIS did it first... and in addition added the built-in 'ISO-2709' format as the format of the 'value' part of the key-value pairs. For bibliographic records this has the advantage that by only parsing the ISO-2709 'header' the whole record and its structure is fully 'known' to the software.

In addition to technological common characteristics, most if not all ISIS family members share also 'social' characteristics, e.g.

- being mainly used in Developing Countries or 'the South', with e.g. a very strong presence in Latin-America, but also - more than can be 'measured' in all kinds of small, often deprived non-connected (no Internet) information centres in Africa and Asia.

- being promoted by many United Nation members and projects, of course first of all in UNESCO-environments, but - as shown by the BIREME example - also WHO and FAO (the AGRIS and ASFISIS systems of FAO can be given as examples here, but also the origin of the WEBLIS library system). The United Nations IFAP and 'Knowledge Society' programmes should not underestimate how much *real* impact comes from the UNESCO-promoted information tools like ISIS, IDAMS, Greenstone etc. - sometimes even indicating that the impact can be the reverse of given financial input or publicity.

The following illustration summarizes the full family up to now.



One could summarize the history by claiming the 'family' now has 4 generations while the 5th generation is being prepared :

- The first generation : CDS/ISIS and Micro-ISIS
- The second generation : enriched ISIS/Pascal interfaces, CISIS-tools
- The third generation : graphical, multimedia and multi-database : WinISIS, ISISDLL
- The fourth generation : WWW-enabled versions (wwwisis, isis3w, openisis…) .

In view of some major technological changes introduced in the newest generation as of 2008 one should perhaps consider the newest ISIS-members (J-ISIS and ISIS/NBP) as representing yet another new 5th generation.

Some highlights of each generation is given below.

1. 1975 - The first generation

   a. 1975 :

CDS/ISIS at the International Labour Organisation (ILO) Centralised Documentation System merged with Integrated Set of Information Services Running on VAX OS on mainframes

b. 1985

Micro-ISIS G. Del Bigio joins UNESCO and creates PC-DOS based version and integrates separated functions into one general customizable, multilingual menu-based interface with full documentation as version 2.3 Version 3.0 – 3.8 : networked multi-user, ISIS/Pascal UNIX-version for Intel-based UNIX OS World-wide distribution and huge success in Developing Countries

2. 1985 - The second generation

    - ISIS/Pascal programmed add-ons (e.g. Heurisko, ADEM, IRIS and ODIN, LAMP) create rich tools; e.g. IRBIS (Russia) for libraries, FAO uses ISIS for its AGRIS-system and ODIN/IRIS extensions for its ASFISIS-system
    - Bireme/OPS (WHO Brazil) creates CISIS-tools suite for command-line database management, uses it for its huge health information databases on the Internet; these are multi-platform (run on Unix/Linux and DOS)

3. 1995 - The third generation

    - UNESCO produces Windows version : WinISIS, with many graphical, multi-media and multi-database features
    - Full library automation systems can be and are developed, e.g. PURNA (India)
    - other libraries start using ISIS for full library automation, e.g. SNAL (Tanzania) uses networked ODIN/IRIS based library system for its university library
    - Bireme distributes a web-server version of ISIS as ‘wwwisis’ running on both DOS/Windows and UNIX/Linux; many applications are developed JavaISIS (Italy) and isis3w (Poland) added to the family

4. 2005 – The fourth generation

    - Advanced web-based tools spearhead further developments : GenISIS (France) allows easy creation of web-based search interfaces
    - WEBLIS (Poland/FAO) is a full-fledged advanced web-based library automation system
    - Bireme develops WXIS and adds XML to ISIS
    - WXIS-based library systems are developed in Latin-America (e.g. OpenMarcoPolo)
    - OpenISIS (Germany) creates first fully Open Source version (webserver, PHP-library) but goes its own way (Malete, Selene)

5. 2008 - The fifth generation

    - UNESCO developes a completely new Java-based graphical interface 'J-ISIS" using not only JAVA-technology but also the embedded Berkeley DB for the storage layer. This project is a fully FOSS-oriented project.
    - BIREME developes ABCD and - at the same time - a fully new technology for its future ISIS-products : ISIS/NBP. ABCD is meant to be the first applcation to be migrated into NBP.

      NBP or 'Network Based Platform' is the new ISIS technology with as the main characteristics :

        - flexible archtecture in which 'ISIS-cells' will communicate through known protocols with several platforms and interfaces; ISIS-cells will also allow to use different storage models as these will be contained within the cells but they behave in the same standardized way towards the external technology used;
        - ISIS databases will no longer have out-dated limitations re database-, record- and field-sizes;

- ISIS databases will be UNICODE compatible
- Indexing will be done by using other FOSS full-text indexers such as Lucene (from Apache Software Foundation).

ISIS is being used by ten-thousands of users, mostly in the Developing Countries where it is promoted by UNESCO and BIREME (for mostly Latin America). In Latin America ISIS is very strongly represented in libraries and documentation centres (it has a 'dominant' position even here), in Africa and Sout-East Asia there are an unknown but high number of users, many of them often non-connected to the internet and therefore still using older technology and with relatively poor ICT-skills. This creates a special challenge to the support of the users-community.

At the 3rd World Congress on ISIS (Rio de Janeiro, Brazil, September 2008) the Users Community decided to make ISIS fully 'FOSS' and co-ordinated by an 'International Co-ordination Committee on ISIS' (ICCI), see : http://portal.unesco.org/ci/en/ev.php-URL_ID=27760&URL_DO=DO_TOPIC&URL_SECTION=201.html

Summarizing the long history of ISIS, one could say that ISIS combines very sound basic 'textual database' principles, a strong tradition and a world-wide but insufficiently co-ordinated users' community with still modern state-of-the-art technological development.

1.3. From 'free' to 'FOSS'

Tip

You might be interested to read the full article on this topic, published at : '*Innovation*', no. 36 June 2008, p. 39-47.

CDS/ISIS as a software has been 'free' and 'open' since its early days, long before 'FOSS' (Free and Open Source Software) became a known software model (or should it be put in the reverse way : long before 'commercial closed software' became widely practised').

1.3.1. ISIS as 'open' software

Whereas ISIS, from the DOS-version produced and distributed by UNESCO as from 1985, always has been ‘free’ – i.e. without cost but with a restriction to the not-for-profit sectors only – the software was not ‘open’ in the strict meaning of the concept as nowadays known as ‘Open Source Software’ with its different definitions (see http://www.opensource.org/docs/osd) and licenses (e.g. (L)GPL, BSD, Creative Commons..).

But in 3 meanings there were, already from this beginning - and therefore long before the FOSS movement began to become really visible –, elements of being ‘open’ in addition to being free(ware) :

1. the standards were open and published. In the ‘CDS/ISIS Reference Manual’, written by its founding father Gianpaolo Del Bigio (working for ILO then UNESCO), the technical details were published in the annexes, allowing others to program their own versions of ISIS using the same compatible standards. E.g. in Slovakia Marek Smihla had programmed executables (e.g. ADEM for data-entry) which ran independently from the ISIS-executables from UNESCO and could write and read ISIS-records. Bireme in Sao Paulo, Brazil, did something similar : they programmed writing, reading and indexing tools with lots of advanced features (e.g. joining databases, linking them as relations etc…) in the C-language (therefore CISIS) which are still the basis for their other ISIS-related software : the DLL and the webservers (WWWISIS, WXIS) and which now have expanded capacity, e.g. 4 Gb max. database size, 1 Mb record size, 60 character-index keys. Co-operation was then set up with UNESCO, e.g. allowing the ‘CDS/ISIS for Windows’ to become a mix of UNESCO-programmed and Bireme-programmed modules.

2. an open, adjustable interface : the software itself was presented as a very flexible environment, with three main features which were used heavily all over the world not only to change its ‘interface’ but also the functions and features.

a. An open menu-structure : Micro-CDS/ISIS was fully based on menus which could be produced and changed by using the software itself, including the definition of ‘actions’ to be invoked by each menu option and allowing hierarchical sub-menus as well as dropping/adding options.

b. An open message system : all messages were/are based on small ISIS-databases which can be edited (each language having its own message-database) and expanded. This not only allowed (often together with the previous feature of open menu's) creation of rather different conformations of the software – taking into account also colors and screen-features which could be changed – but also expansion and introduction of parameters (which could then be 'read' as messages) for additional software running inside ISIS (see further : ISIS/Pascal add-ons), as amply used e.g. by the cataloguing interface 'ODIN' and OPAC 'IRIS' (by the author of this article).

c. A programming tool 'ISIS/Pascal' which acted as an 'API' (with published calls for functions and their parameters) inside CDS/ISIS. ISIS/Pascal programmes, varying from a few lines to thousands of lines for sophisticated applications, could be included into the program either as 'format exits' (to expand the functions of the already very rich Formatting Language) or as 'menu exits' to expand the functions of the menus, allowing almost independent interfaces to 'take over' the CDS/ISIS environment in the creation and manipulation of its databases. One feature illustrating the 'openness' was the possibility of adding a parameter in the 'SYSPAR.PAR' initialization file to automatically invoke a menu and its option, therefore allowing the menu-interface to be skipped and immediately presenting the new ISIS/Pascal interface. In this way full OPAC (e.g. IRIS using a welcome-screen which could be invoked by a time-out mechanism after a previous session was left) and CD-ROM search modules (HEURISKO is an example) were written, loan-systems for libraries and thesaurus-management tools were produced.

d. Last but not least : the 'open character' of the Formatting Language. The Formatting Language is a grammar used to define in a detailed way how elements of the database-data, taken from repeatable fields and subfields, also from other records in the same or other databases (therefore resembling relational approaches) and with navigation links, will be 'processed' in some output (for display, sorting, printing, exporting). It was largely expanded with graphical features in the Windows version (RichText but also images and extra text- and image-boxes). Together these strong 'data-processing' and 'presentation' features of the Formatting Language have allowed the production of rather new 'identities' of the software, e.g. as a Library Management software with OPAC and Loans System (e.g. PURNA from India). In current applications, based on web-technology, the Formatting Language is still gracefully used to produce HTML-elements (e.g. links but also tables), even if more dedicated tools for that, e.g. PHP, are now added to the power of the own ISIS Formatting Language.

1.3.2. ISIS as full open source software

Already in 2001 UNESCO decided to embark on this relatively new approach of not only providing the software for free but also making the source codes in principle 'open', i.e. publicly available (see : http://portal.unesco.org/ci/en/ev.php-URL_ID=13803&URL_DO=DO_TOPIC&URL_SECTION=201.html). This has finally lead to a framework of its wider 'Free and Open Source Portal' approach promoting the idea and adding other softwares, e.g. Greenstone, into their 'basket' of supported and promoted softwares for better professional development also in the Southern and transitional countries. UNESCO's FOSS Portal can be found at : http://www.unesco.org/cgi-bin/webworld/portal_freesoftware/cgi/page.cgi?d=1, with interesting links to discussions of the FOSS history, licenses and case studies. In reality however the source codes for existing ISIS-software are to be requested from UNESCO, but the new softwares will be fully available on public websites.

At Bireme/OPS/WHO a similar decision was taken in 2006/7. No longer would the institute charge a small fee for their software (as was the case before, e.g. 150 USD for official registration as a user with support rights) and therefore make it 'free', but also the sources have been and are still being prepared for publication of all their software, including the basic CISIS-modules. Their new ISIS-generation software, called 'ISIS-NBP' (Network Based Platform) will follow FOSS-methods (including a 'community' with possibilities to contribute, discuss and download sources at the URL http://reddes.bireme.br) to show their firm commitment to FOSS. As the newest full-fledged application, ABCD will be fully published as open source, even if the original development is still centrally managed by Bireme and its own programmers, as the project is now also supported by the Flemish Interuniversity Council (VLIR) with specific requirements to present it as a full competitor to other library systems (including other FOSS–softwares like KOHA and NewGenLib) and to this end needs some more central control for specific purposes.

The advantage of becoming fully open source – for all software - lies in the fact that users, certainly (programming) skilled ones, can fully check on the internal mechanisms and propose/make changes if so desired. One example: WinISIS has a slightly different way of sorting values taken by the 'VAL'-function (i.e. removing padding 0's first) which is not a bug as such and therefore does not 'need' to be corrected by the software provider; with access to the source codes one could change this however.

As is always the case with open source software, it would be best not to make such changes without consulting/informing the ‘developers’ community’.

1.4. Aims of ABCD

ABCD aims at providing an integrated library management tool covering all main functions in a library, i.e. acquisitions, bibliographic databases management, users management, loans management, serials control, end-user searching on local and external bibliographic databases and library portal.

it is not the first time in the ISIS-history and -environment that such effort has been undertaken. Open MarcoPolo, Clabel and - as a more advanced effort - WEBLIS are predecessors to ABCD in this sense.

- ABCD as a generic, flexible bibliographic tool

As the name itself suggests, ABCD however aims not only at providing a solution for libraries, but for documentation centres as well. These typically have slightly different needs, e.g. have more specialized collections, higher needs re contents disclosure (e.g. by providing abstracts, using thesauri etc.) and requiring more flexibility in the bibliographic structures. For this reason ABCD not only has tried to include full-text features but was *principally* conceived to offer a very open solution, allowing any fields structure to be created and maintained within the same software. By the database technology of ISIS itself, which is quite flexible and non-restrictive, bibliographic structures can be created without a need to 'normalise' all elements into a series of tables or relations (as is the case with relational database technology) and in most cases all bibliographic elements can be contained into one single database - only for optimization purposes ISIS would expect some semi-relational approaches to be implemented.

As a library system, however, ABCD comes pre-configured for some major bibliographic standards, i.e. MARC21, CEPAL and AGRIS. But we repeat : the same mechanisms, interface and forms can be used to create and maintain *any* structure, whether bibliographic or not. In version 2 ABCD also adds tools for 'digital library' (databases with full-text contents) and document delivery functions.

So, to put the aims a bit more precise : ABCD aims at providing a very generic/generalizable tool for managing libraries and documentation centres.

- ABCD as a librarian-oriented tool

Another specific aim of ABCD is to offer a tool for librarians, rather than ICT technicians. This is achieved by taking library and information science principles (rather than computer or programming principles) as the starting point, even in the design of the databases themselves. Typically a bibliographic record is one real entity in an ISIS database, not a complicated series of elements 'queried' or 'joined' together from many tables (as in relational systems), however preserving criteria like efficiency (in space usage, speed of operation..). Each entity subsequently can be thoroughly 'moulded' by the librarians themselves with the use of the ISIS Formatting Language (FL), which allows dealing with all elements of an entity (e.g. a substring from a subfield of an occurrence of one specific field at micro-detail level) without real programming - even if the FL allows some degree of programming logics like loops and nested conditions - for the creation of any output format. This output can be anything like a sort key, an indexing key, a screen format or - as is the case in e.g. ABCD - ISIS-data embedded in web-pages or any other grammar such as XML. Lots of teaching experiences with ISIS show that librarians are perfectly capable of understanding and using all this, reaching advanced results without any real programming.

- ABCD as a tool for developing countries

  ABCD aims at providing librarians and information workers in developing countries a very powerful tool, which however takes into account some specific realities, such as :

  - low availability of ICT skills : as with previous ISIS-based solutions, librarians are - in principle - enabled to solve their problems by avoiding unnecessary software architectures while still allowing flexibility within the software (e.g. through the Formatting Language);
  - low availability of bandwidth and connectivity : by using modern web-techniques such as AJAX and JavaScript, data-traffic in between client and server is kept minimal, allowing the local computer (at the 'client-side') to process the data as much as possible without always referring to the server; also the graphical design is kept rather sober for the same reason.

1.5. Actors and partners of ABCD

ABCD, as with all major software projects, is a conglomerate effort of several actors and partners.

At the following URL a list of the main actors and partners is maintained :

http://reddes.bvsaude.org/projects/abcd/wiki/HallFame?version=20. [to be updated]

The main input, obviously, comes from the Brazilian BIREME institute (see http://www.bireme.br), which has availed all of its ISIS-based technology to be combined into one 'culmination' product which is indeed ABCD. In fact the original idea stems from its actual Director, Mr. Abel Packer, who has generously availed also worktime of his programmers and software managers.

A special mentioning certainly is appropriate for Mrs. Guilda Ascencio, Venezuela, who was the main programmer of the ABCD Central part with its modules, based on her own 'Orbital Documental' software, in which she had proved that very advanced applications, combining library and other documentation management issues, could be built using ISIS and web-technology.

Both authors of this book have acted as coordinators of the ABCD development project, trying to assemble the many pieces of the puzzle - and to make sure the final picture of the puzzle not only is more or less correct but also somehow attractive.

Two more institutional partners have to be mentioned as well :

- UNESCO : as explained above in the section about ISIS history, it is clear that UNESCO has an enormous merit in developing and promoting ISIS. ABCD will become part of the set of UNESCO-promoted ISIS products.

- VLIR/UOS : the 'Development Co-operation' section of the Flemish Interuniversity Council (VLIR, Belgium, see http://www.vliruos.be), through a project '**D**evelopment **O**f and **C**apacity **B**uilding in **I**SIS-**B**ased **L**ibrary **A**utomation **S**ystems' (DOCBIBLAS) which is promoted by the Belgian co-author of this manual, has selected ABCD as the library automation solution it wants to promote with its partner university libraries in the South (Latin-America, Africa and South-East Asia). Most of the additional developments of ABCD (e.g. Unicode, Digital Library...) was funded by VLIR/UOS. A big thank you is clearly deserved here.

2. ABCD technology

2.1. ISIS databases

ISIS databases are files in which information is contained in sequentially numbered records (MFN's or Master File Numbers) with values (mostly textual) stored in fields with a 'tag' (or numerical identifier) and subfields (with a one-character identifier). Subfields, fields and records all are variable-length and 'variable occurrence' varying from 0 (not present) to any higher number of occurrences, with a maximum depending on the ISIS-technology used but in the newest generation (in J-ISIS and #-ISIS) without limit.

Records are structurally described in a 'header' for each record itself, instead of the usual table-header in relational databases. By doing so ISIS reflects more the concept of each record being a 'document' in its own right with its own document structure, like e.g. books, articles or web-pages indeed. Therefore we prefer to call ISIS a 'documentary database' in which documents are stored as a record with variable structure and length. This avoids complicated structures of 'normalized' relational structures, which are very efficient in storing highly structured data but less so for semi-structured textual data.

This means that the records themselves can be quite polymorph, meaning structurally different with any combinations of fields. In principle ISIS can handle bibliographic records along with user-data and transactional data (e.g. loans) in one single database, but because of 'semi-relational' capabilities (fast retrieval of any part of a record in any ISIS-database at run-time, i.e. by the Formatting Language creating the output without the need for these 'relations' to be pre-defined) typically ISIS-applications will use some few databases, e.g. in ABCD only 3 or 4 databases (one for bibliographic entities, one for users, one for transactions and possibly one for items) can allow to run a full library.

In the 'classic' ISIS technology [1]all variable-length records (with (sub-)fields containing the values) are stored in a 'Master' file (.MST) and record-positions are kept track of in the 'Cross-Reference' (.XRF) file, which can be seen as a 'first-order' normal index on the records in the database. New or even just edited records are always appended at the end of the Master file and references in the XRF are updated accordingly, necessitating some 'compacting' at times to get rid of deleted and/or inactive (versions of) records.[2]

All values specified by a 'Field Selection Table' (which uses the Formatting language, therefore allowing very flexible and powerful definition of selected elements), are included into a B-tree 'Inverted File', which can be seen as a 'dictionary' of terms with the exact 'address' (record, field-tag, occurrence, position within occurrence) attached to them. This allows very efficient retrieval, including full-text based, of any element defined as being 'retrievable'. ISIS was one of the first databases to offer full-text, which became only popular decades later. This 'Inverted File' (or IF) has several components (with nodes .N01/.N02 and leaves .L01/L02 files) for efficient organization - because in certain applications with intensive indexing the IF can be even larger than the database file itself !

So typically ISIS-databases exist of some 10 files : a MST with XRF, the B-Tree Inverted File files and some definition tables for the fields, the data-entry form and the indexation.

All this is changing with the new database technologies introduced in 2009 with e.g. J-ISIS : Berkeley DB uses a different storage in separate files with the definitions incorporated into the main data-files. But basically the concept of 'tag-value' pairs (an identifier and a content), on which a powerful Formatting Language and field-based plus full-text indexing is applied, remain the core of ISIS-databases.

2.2. CISIS

CISIS is the software developed by BIREME to handle ISIS databases from the Command Line in UNIX/Linux or DOS/Windows. This sofware has been written in the C-programming language and hence the name of this ISIS family member. CISIS mainly exists of a series of 'utilities', i.e. command-driven executables which perform all types of functions in ISIS-databases, like creating records, updating and searching them, updating the Inverted File, import and export and many other functions, sometimes unique in the 'ISIS Family', e.g. joining records from different databases according to common keys, indexing and searching from different Inverted Files for one database.

Actually CISIS as a set of utilities contains more than 25 different tools or executables. As this is not a manual on CISIS, we will not deal with all of them, but some are worth being mentioned, certainly also because we will use them for some off-line functions of ABCD.

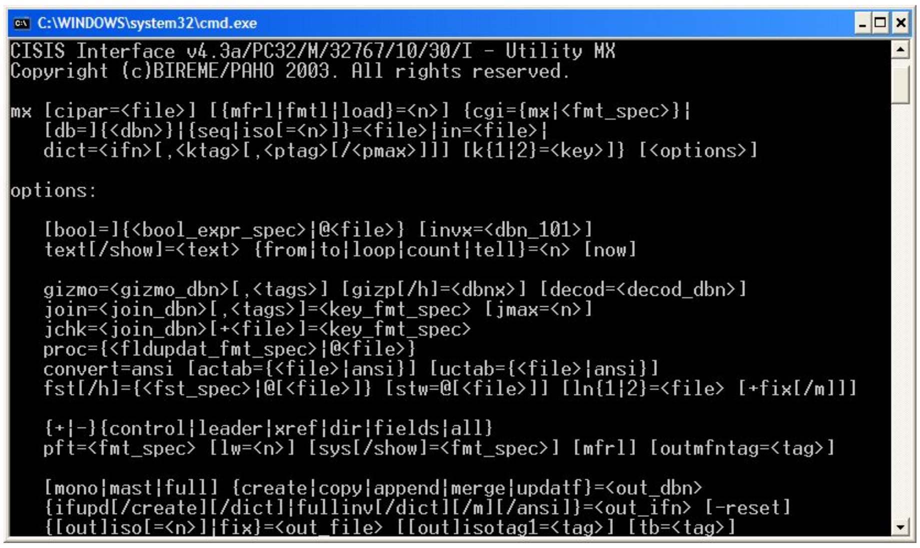
In recent years BIREME started creating 'variations' of their CISIS-utilities, mostly based on different lengths for index-keys (from 30 up to 60 and 256), different max. record sizes (up to 1Mb) and overall database sizes (from 512Mb up to 4 Gb). E.g. the 'ffi'-version of CISIS works with large records up to 1 Mb and large databases up to 4Gb. The 'Lind'-version on the other hand drops proximity-searching for faster and compacter (but non-incremental) indexes, mostly for full-text purposes. The 'BigISIS' version finally (for Linux only) goes up to 512Gb, paving the way for very large databases.

Also the 'Unicode'-version is a special version of CISIS, which in ABCD has to be put in a dedicated sub-folder of the cgi-bin folder and linked to any database using it from the 'ABCD.def' file (see later).

2.2.1. The Master / Xross-reference tool : mx

The mx tool is the main CISIS utility, it could easily be baptised as 'CDS/ISIS for the command line', meaning most things which can be done with (M)asterfiles and (X)rf-files - therefore 'mx' indeed - with ISIS can also be done with MX. Just to give an idea we give the list of parameters mx accepts (as this list is given when invoking the command in a command-line environment such as the CMD-window in Windows or a terminal-window in UNIX/Linux. As one will see, too many parameters are available, meaning mx is an enormously powerful tool for ISIS-database management, but it deserves a manual and training in its own right !

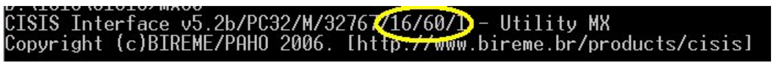
[1]'classic' refers the technology of ISIS since its introduction in the 1970's up to the introduction of J-ISIS and #-ISIS) in 2009.
[2]This behaviour, necessary because of the variable length of records, makes ISIS less suited for very dynamic databases, such as transactional applications (loans e.g.).



```
CISIS Interface v4.3a/PC32/M/32767/10/30/I - Utility MX
Copyright (c)BIREME/PAHO 2003. All rights reserved.

mx [cipar=<file>] [{mfrl|fmtl|load}=<n>] {cgi={mx|<fmt_spec>}|
   [db=]{<dbn>}|{seq|iso[=<n>]}=<file>|in=<file>|
   dict=<ifn>[,<ktag>[,<ptag>[/<pmax>]]] [k{1|2}=<key>]} [<options>]

options:

   [bool=]{<bool_expr_spec>|@<file>} [invx=<dbn_101>]
   text[/show]=<text> {from|to|loop|count|tell}=<n> [now]

   gizmo=<gizmo_dbn>[,<tags>] [gizp[/h]=<dbnx>] [decod=<decod_dbn>]
   join=<join_dbn>[,<tags>]=<key_fmt_spec> [jmax=<n>]
   jchk=<join_dbn>[+<file>]=<key_fmt_spec>
   proc={<fldupdat_fmt_spec>|@<file>}
   convert=ansi [actab={<file>|ansi}] [uctab={<file>|ansi}]
   fst[/h]={<fst_spec>|@[<file>]} [stw=@[<file>]] [ln{1|2}=<file> [+fix[/m]]]

   {+|-}{control|leader|xref|dir|fields|all}
   pft=<fmt_spec> [lw=<n>] [sys[/show]=<fmt_spec>] [mfrl] [outmfntag=<tag>]

   [mono|mast|full] {create|copy|append|merge|updatf}=<out_dbn>
   {ifupd[/create][/dict]|fullinv[/dict][/m][/ansi]}=<out_ifn> [-reset]
   {[out]iso[=<n>]|fix}=<out_file> [[out]isotag1=<tag>] [tb=<tag>]
```

A glance at the many parameters show that mx can not only search ISIS-databases (bool=) but apply on-the-fly GIZMO (string-substitutions) and ANSI-conversion (ansi=), join fields of records from different databases but identified by their IF-entry (join= and jchk=), apply data-entry processes (proc=) and inverted file operations.

As CISIS comes in several varieties, according to the capacity of the databases and Inverted File keys intended, we need to specify that for ABCD we will only use the '16/60' variety of mx and other CISIS-tools. This can be verified from the information mx gives when invoked without any parameter as illustrated :



```
CISIS Interface v5.2b/PC32/M/32767/16/60/I - Utility MX
Copyright (c)BIREME/PAHO 2006. [http://www.bireme.br/products/cisis]
```

The most relevant uses of mx in this context of ABCD are :

1. import of ISO-records into an ISIS-database, e.g. the command :

   ```
   mx iso=myISOrecords.iso create=mydb now -all tell=100
   ```

   will read the file myISOrecords.iso and create an ISIS database 'mydb' without waiting for any user-input ('now'ait) and without showing any information on the screen (-all) but showing progress after every 100 records imported.

   ### Note

   In ABCD we use this to import a larger quantity of ISO-records into a database, as a high number and therefore long processing time would invoke the time-out of the web-server to stop the process.

2. index an ISIS-database, e.g. the command :

   ```
   mx mydb ifupd/create=mydb fst=@mydb.fst stw=@mydb.stw now -all
   ```

   will create an 'Inverted File' named 'mydb' using the mydb database with the indexing specifications given in the FST 'mydb.fst' and omitting the stopwords listed in mydb.stw, again without interactive mode or output (now -all).

   ### Note

   In ABCD we use this to create an index off-line in case - as is often the case - the database is r

2.2.2. Inverted File tools : mz, ifupd, ifkeys, ifload, ifmerge

These are more specialized tools to generate/update the ISIS Inverted File with its B-Tree technology and parts (leaves and nodes) from the command line with some more optimized speed and more options. E.g. MFN-ranges can be defined, keys can be taken from the previously created LK (link) files (ifload) or nodes-files (ifmerge) of the B-Tree, which can be balanced etc.

We don't normally need to use this with ABCD, but knowing the possibilities exist, especially in the case of very large databases, is certainly useful.

2.2.3. other CISIS-tools

Other tools to be briefly mentioned only are e.g. :

1. retag : this tool will change the tags of the fields according to a given specification - which can have instructions on many fields in one run
2. mfcrunch and ifcrunch : to convert the ISIS-files (resp. MST and the IF-files) from DOS/Windows to Unix and v.v.
3. mkxrf : to re-create the XRF-file for a given database, in case this is lost or got corrupted - the tool will analyze the MST-file and assign XRF-records into the XRF.
4. ctlmfn : to edit the values of the 'control-record' of the database, in which the maxMFN and other very technical values for the database are stored - for experts only !

2.3. ISIS Formatting Language

The ISIS Formatting Language (FL) is one of the most important parts of the software because it gives ISIS-managers the possibility to exactly define what ISIS will produce out of the databases at many stages of the software, e.g.

- what ISIS will show on the screen, i.e. 'present' (defined in the Print Format Table or PFT)
- what ISIS will use for the creation of indexing keys (defined in the 3rd column of the Field Select Table or FST)
- what ISIS will use for sorting the records
- what ISIS will use as exported values (defined in the reformatting FST)
- what ISIS will use as values to validate input in fields (given in the validation tables).

2.3.1. The FL for presenting values

This is by far the most important function of the Formatting Language : specifying which data exactly need to be taken and how they will be 'displayed' or 'printed' (to the screen, to a printer, to a file, to a webpage...).

Separate documents exist to deal with this extensive language, e.g. the dedicated chapter in the ISIS Reference Manual, published by UNESCO (June 2004, chapter 8, p. 94-122). BIREME has published a dedicated manual on the CISIS Formatting Language, available online.

Basically there are three types of statements in the ISIS FL :

1. values from fields, given as : Vx, where 'V' denotes the value (or 'contents') of a field with tag 'x', Vx^a is the value of the subfield a (^a) of field x and (Vx/) is the series of all occurrences of field X separated by a 'new-line' (/) since the parenthese embrace a 'repeatable group' of statements to be applied to all occurrences (repeatable fields are a strong special feature of ISIS).
2. literals or quotes strings, which can be 'unconditional' (single quotes), |conditional| (pipes indicate the string will only be produced if the related field is present) and "repeatable" (double quotes will only produce the string at the first occurrence of a repeatable field).
   ISIS-applications on the web, such as ABCD, create web-pages with HTML-tags using this method of adding literals to field-values, e.g.

   ```
   '<table><tr><td>' Vx '</td><td>' Vy '</td></tr></Table>'
   ```

will display resp. the fields x and y in two columns of a table in HTML. Note that all HTML-codes are quoted (as unconditionals) and the values taken from the fields in the database are inserted by referring to them with the V-statement.

3. commands, which can be of different types, e.g. :

- mode commands : mhl/u (mode heading lowercase/uppercase), mdl (mode data upper/lowercase) or mpl/u (mode proofreading upper/lowercase)
- (in Windows-environments) : commands defining screen attributes (colors, fonts, boxes) or links (requesting the operating system to open other data, e.g. multimedia data referred to in a record), e.g.

  LINK('click here for tull-text', OPENFILE Vx) will request - when the user clicks on the hyperlinked text 'click here for full-text', Windows to open the file of which the name is in Vx, with the Windows-application associated to the extension of that file.

- the REF-command, which can retrieve data from other records (in the same or another database when expressly referenced to), allowing semi-relations setups in ISIS-applications (but with the advantage that the relation is followed only at run-time when requested). e.g.

  REF(['users']) L(['users']V2),V1) will retrieve the value from field 1 in the database 'users' if the L(ookup) function has found the value of field 2 (in the actual database) in the index of the users-database, so that the MFN of the record can be identified.

- conditional routing statements : e.g. 'IF...THEN... (ELSE....)FI' or even the 'SELECT [case1 case2...] ELSE-CASE... ENDSEL construct can be used to apply formatting statements only to database values which comply with given conditions.
- in the CISIS-environment extra FL statements are available, the most important one being a command which will actually PROCess a record to alter the contents of the fields. The general syntax is :

  proc(x|y...) where x or y can be any of the following : 'Dxxx' (to delete field with tag xxx) - |Axx#|value|#| (to Add value into field xx)

- functions, mostly for string-operations (e.g. substr, size, val) or numerical (e.g. rmin, rmax, rsum...)

  Full documentation on the Formatting Language is available, e.g. the 'CISIS Formatting Language' published by BIREME.

2.3.2. The FL for definition of indexing keys

The same formatting language, but of course without any appearance-related effects, can be used to exactly define which values should go into the Inverted File of ISIS. This will be defined in the third column of the 'Field Select Table' where the extraction format using the FL is to be used. See also the discussion of the FST definition in the chapter on database definition and management of this manual.

Since the full formatting language - except graphical elements - is available, the REF-function e.g. can be used to take into the Inverted File values different from the actual field contents, even from another database. This can e.g. be used to substitute codes for their full explanation or v.v.

2.3.3. The FL for definition of sorting keys

The same reasoning can be applied for the definition of keys which ISIS will use to sort records : again the actual sorting values can be processed values derived from the actual field values, by using the FL.

2.3.4. The FL for conversion during import/export

During import/export of records, most ISIS-applications will allow the use of a 'reformatting' FST, which has in its third column the exact definition of what to export/import, and in the first column (the 'IDentifier') the tag to be assigned to this value.

2.3.5. The FL for validation statements

The Formatting Language can also be used to create error messages in case defined conditions are (not) met. These conditions will be checked when passing data entered into a data-entry form into the record for storage. ABCD provides this technique by default as explained in the section on record validation. An example again can clarify this easily :

```
if a(Vx) then 'This field is mandatory, please check again !' fi
```

This statement will produce, on the screen, the message 'This field is mandatory, please check again' if the value of the field with tag x does not exist or is A(bsent).

More sophisticated statements can be used for more advanced quality/consistency checking, e.g. using a 'SELECT' construct, or even checking the value in another database (with the earlier discussed 'REF'-function) to see whether it is a valid entry.

2.4. ISIS Script

ISIS Script is a scripting language developed by BIREME in order to make stronger functions available to the ISIS webserver 'WWWISIS'f or creation of pages with elements from ISIS-databases. ISIS Script in fact was one of the main elements in the stepping-up from WWWISIS to 'WXIS' which is the underlying web-ISIS-server for ABCD.

ISIS Script scripts are stored as files with an extention .XIS. ABCD uses more than 100 such scripts, most of them in the php/dataentry/wxis folder but also iAH (the OPAC) makes extensive use of such scripts.

Obviously we cannot discuss the whole power of the ISIS Script language here. As a longuage it uses XML-like statements, e.g. in between the tags <pft> and </pft> a print format can be given and this format can be displayed by putting it in between <display> and </display> tags. All WXIS parameters can be defined within the <parm> and </parm> tags and fields can be defined with values, e.g.

```
<field action="replace" tag="6000">ValueOfField6000</field>
```

will put the string 'ValueOfField6000' into the field with tag 6000 (such high-value tags, in fact all tags above 999, are mostly used within ISIS-applications for temporary internal values which are not really stored in ISIS-records but rather 'virtual records'.

ISIS Script allows more flexible manipulation of data-elements, taken from ISIS-databases, in web-pages. In combination with PHP (see the dedicated section on PHP), which is a language for creation of web-pages powerful results are possible and this certainly adds to the general advanced functionality of ABCD.

Of course more details on the ISIS Script language can be found in the dedicated documentation, available online from BIREME.

2.5. J-ISIS

J-ISIS or 'Java-ISIS 'is the new version of ISIS fully programmed in Java. Not to be mixed up with an older 'Java-ISIS' which was only limited in its use and distribution, developed in Italy but obsolete nowadays.

J-ISIS is developed by J-C Dauphin, formerly of UNESCO, since 2010 and really representing a 'new generation' of ISIS, because it drops some 'classic ISIS' technology (such as the MST and XRF) but keeps the 'core' of ISIS : the Formatting Language, and the main characteristics of variable length records/fields and ISO-2709 compatibility. Some other technological characteristics are :

- programmed in Java, using the rich Java-libraries and development environments, as a desktop client/server application accompanied with WWW-client, combining the best of both worlds into one.
- using Lucene indexing (of Apache Software Foundation), so full-text indexing and retrieval with ranked output, is possible.

-
- maintaining the ISIS Formatting Language fully (including external REFs etc.), even extending it with a powerful grammar parser to help creating formats
- freely available and published as FOSS at the URL : https://www.kenai.com/projects/j-isis .
- allowing fast digital library applications (based on Apache Tika document-converting-to-text)

In view of its many promises and capabilities, J-ISIS provides possibly a future developing-road for ABCD.

2.6. PHP

PHP is a 'Hypertext Preprocessing' language, which means it is a programming language for web-pages. As one of the successful 'FOSS' products it is nowadays very popular and wide used, often in combination with Apache and MySQL databases. This has even lead to packages such as 'EasyPHP' and 'WAMP' (Windows, Apache, MySQL and PHP) which allow to install these often-combined softwares into one package.

As usual there are some criticisms on PHP as a language, but a fact is that it is very popular and getting more powerful with each release. ABCD uses e.g. also 'controls' or ready-available modules for specific functions, which are freely available.

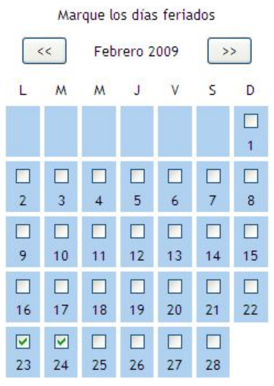
2.7. JavaScript

The official name of JavaScript is 'ECMA Script' but JavaScript is the popular name of a technology which is nowadays used in many web-pages : relatively small programmes embedded into the HTML-codes of the pages. Contrary to the name the language is not really linked to the JAVA Programming Language. JavaScript nowadays is supported by all up-to-date web-browsers and does not need any extra software or configuration. However it remains to be an option which can also be switched off (in Firefox e.g. : Tools|Options|Content, where both JavaScript and Java can be disabled), so make sure that the JavaScript option is enabled for the use of ABCD.

ABCD uses JavaScript 'scripts' inside its pages in very many instances, one reason being that by doing so the local computer can process data without a need for high traffic in between the server and the client (which is important in slow connectivity conditions).

As an example of a simple JavaScript we can refer to the script 'lrtrim.js' (in the ABCD-folder \ABCD\www \htdocs\php\dataentry\js\) which is called upon from several ABCD-PHP pages. The script trims white-spaces at the right or at the left side from strings. This can be easily done locally, no need for sending the string to the server together with the request to trim it and then having it returned from the server. Therefore the script is loaded into an ABCD page and executed locally.

Also generally available JavaScript existing modules are being used, e.g. for the calendar function in the Loans module or for the 'HTML Editor' (FCKEditor.js). Under here the calendar example is shown, based on the JavaScript 'popcalendar.js' which can be found e.g. in the folder php/loans/js of the ABCD home-folder (/ABCD/www/htdocs). This little tool displays any month of the calendar and allows marking the holidays to take them into account when calculating the loans period !



L	M	M	J	V	S	D
						☐ 1
☐ 2	☐ 3	☐ 4	☐ 5	☐ 6	☐ 7	☐ 8
☐ 9	☐ 10	☐ 11	☐ 12	☐ 13	☐ 14	☐ 15
☐ 16	☐ 17	☐ 18	☐ 19	☐ 20	☐ 21	☐ 22
✓ 23	✓ 24	☐ 25	☐ 26	☐ 27	☐ 28	

Most JavaScript functions however are not visible on the screen, but perform useful functions within the web-pages of ABCD. So even if tools like the above mentioned (the HTML editor or the calendar) are considered unnecessary, still it is important to keep the option to run JavaScript within your browser 'on'. As with Java, this option e.g in Firefox can be checked in the Tools|Options|Content tab (in Internet Explorer one has to activate 'Enable for Active Scripting' in the Scripting section of the security zone 'Internet' under Tools|Options|security).

2.8. JAVA, Groovy and Jetty

JAVA is at the same time a programming language (like e.g. 'C++') and a 'runtime compiler', which means that programs written in JAVA need a 'RunTime Environment' (RTE) version of JAVA which will compile the program for the given Operating System and CPU combination at run-time (i.e. when the user executes the program). By doing so the JAVA programs are completely 'multi-platform' (Windows, UNIX, Linux, OS/X...) because such RTE's exist for all platforms and are freely available for installation. So make sure your computer has its own JAVA RTE installed ! Both Sun (the real Java promoter) and Microsoft offer free versions of Java (e.g. at http://java.com/en/download). JAVA is not only free but also 'Open Source' and therefore can be reported as being fully 'FOSS', as is ABCD.

ABCD uses JAVA only for the 'advanced' loans module, which comes as an extra option (see the chapter on the Circulation module). This advanced circulation management module is intended only for larger institutions with more complex circulation rules and multiple branches with their own loans policies or with user-databases in other formats (e.g. SQL). Also more interactive 'MyLibrary' -style functions can be offered. In order to allow such more complicated software-combinations, ABCD calls upon JAVA to provide web-services and links with other database-models.

Groovy is an object-oriented programming language for the Java Platform which can be used as a scripting language for the Java Platform.

The advanced ABCD Loans module (EmpWeb) also uses Jetty-technology, which is aHTTP Server and Servlet container written in Java.

Jetty can be used as:

- a stand-alone traditional web server for static and dynamic content
- a dynamic content server behind a dedicated HTTP server such as Apache using mod_proxy
- an embedded component within a java application

2.9. MySQL

MySQL is a relational database developed as FOSS but with a 'dual license' scheme, allowing both commercial and free applications. MySQL has been taken over by Sun Microsystems, a strong defender of FOSS software, e.g. Java, but then went into the hands of Oracle along with Java. This has lead to a 'fork' with many developers moving to MariaDB as they believe FOSS has to be kept out of big commercial companies.

As a database MySQL has become incredibly popular because of its ease of use and combined packing with e.g. Apache and PHP for easy deployment of database-driven websites.
Examples of such pre-packaged combinations of Apache/PHP with MySQL are : EasyPHP (http://www.easyphp.org) and WAMP for Windows or XAMP for Linux (http://www.wampserver.com). Both are Open Source and free to use (GPL license).

Critics claim its 'relational' qualities are still lagging behind - even if improved a lot since the earlier days - as compared to e.g. PosGreSQL or of course the major relational databases like Oracle or IBM DBII. Some other library automation packages are completely using MySQL for the databases, the best known being KOHA, albeit that KOHA currently adds another type of database, i.e. 'Zebra', exactly to avoid limitations of MySQL for library purposes, esp. regarding full-text indexing.

The 'SQL' part of the name means 'Standard Query Language', denoting a standard grammar for retrieving data out of relations (related tables), heavily relying however on its relational structure. For this reason e.g. ISIS is not using SQL as it does not store its data into tables with fixed cells and structures.

MySQL will only be used in ABCD within the 'Advanced Loans' module, which is a non-standard extra (see the chapter on the loans module in this manual). There it will be used to store the transactions of the loans system, as these are administrative data which can be more efficiently handled by this type of database as compared to ISIS with all its - in this case unnecessary - flexibility and text-oriented features. Since EmpWeb uses the 'Java DataBase Connector' (JDBC), in fact any other compabitle RDBMS can be used as well.

2.10. YAZ

2.10.1. Introduction

YAZ is a freely available software library for embedding the Z39.50 protocol in applications.

Z39.50 is used as a protocol to retrieve data from other catalogues, mostly in MARC-format.

A PHP-implementation is available, which is what ABCD uses for its 'Z39.50' function in the cataloging module.

2.10.2. Compilation in Windows

2.10.2.1. The problem

Working in a 'Free and Open Source Software' (FOSS) environment is a blessing indeed : one has 'free access' to the software including its underlying code. Free both in financial terms as in 'freedom', all sorts of use, change etc. being allowed within some limits posed by the specific FOSS-license used. However FOSS also has its flaws and disadvantage(s) : many different communities (developer groups) exist, sometimes disagreeing or sometimes using their 'freedom' to divert but always with the aim of improving the software. In the case of ABCD ILS, the upgrade from PHP (the main scripting language used to create the WWW-interface) from version 5.2 to 5.3, caused some headaches and problems to solve. One of these is related to the specific relationship with the Apache web-server, which is also FOSS and therefore also follows not-always-obvious paths of development. Concretely the changing way how Apache and PHP for Windows (in both cases not their prime focus or preference !) are compiled with versions of Microsoft Visual C++ caused problems which needed to be addressed. This manual explains how to go about this problem

2.10.2.2. Rationale

This document is for those who want to install Apache and PHP5.3 in any Windows version (Windows 7, XP.. etc.) and whether 32-bits or 64-bits. With previous versions like PHP5.2 or earlier, there was no problem, it only started with PHP5.3 regardless of the Apache 2.x version. Why ? The standard Apache software is provided by the 'Apache.org' group, and is compiled with MSVC++ v.6, a non-free version which is also old. The binaries (executables) of Apache for Windows still are compiled with this version. PHP for Windows also was compiled with MSVC++ v.6 and v.9 until PHP 5.2, but since PHP 5.3, compilation is only done with MSVC++ v.9, causing the modules not being necessarily compatible with the software of Apache.org.

PHP 5.2 php 5.3 apache.org Yaz.dll is compiled by the PHP team; this is the version used by ABCD v.1 PHP 5.3 and its modules need to be compiled with MSVC++ v.6. It is not advised due to performance problems and memory management with multi-core CPU's. apache lounge Idem Needs to be compiled with MSVC++ v.9 or later, which is the aim of this manual.

Table 1.1. Apache and PHP versions

	PHP5.2	PHP5.3/4
apache.org	Yaz.dll is compiled by the PHP team; this is the version used by ABCD v.1	PHP 5.3 and its modules need to be compiled with MSVC++ v.6. It is not advised due to performance problems and memory management with multi-core CPU's.
apache.lounge	idem	Needs to be compiled with MSVC++ v.9 or later, which is the aim of this installation instructions.

A complete explanation can be found at : http://windows.php.net/download/#php-5.3 Left sidebar: Which version do I choose?

Notes:

o MSVC++ v.9 = MSVC++ Express 2008 and later

o ABCD v.1.2 will be distributed with Apache Lounge and PHP 5.3 with php_Yaz.dll already compiled

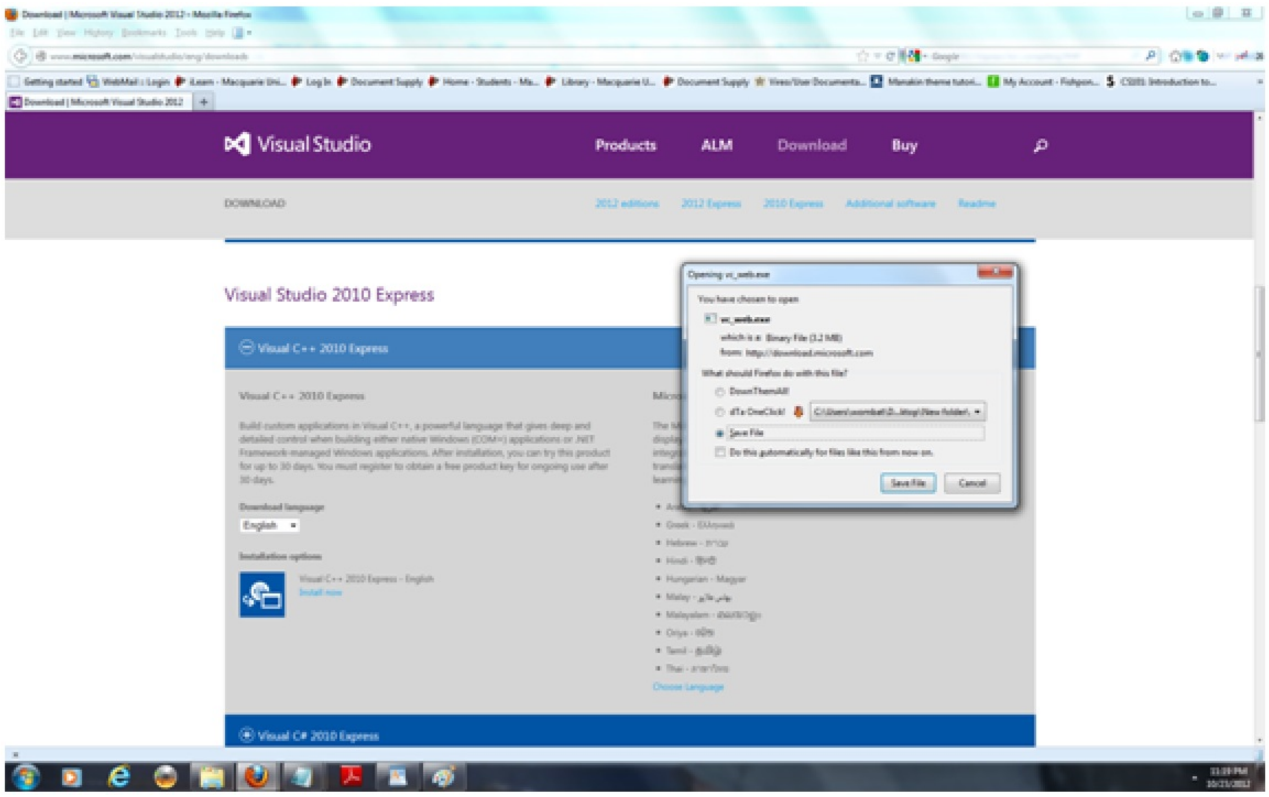
o What happens if the PC is virgin and the distribution package is copied? There could be a problem with the dependencies executables. Therefore one has to install the libraries of Install Windows C++ runtime: Microsoft Visual C++ 2010 SP, explained later. The instructions of this paragraph is not required for the compilation, only for the execution or running of the software

2.10.2.3. Compile YAZ for PHP 5.3 on Windows 7

The following instructions describe how to compile the YAZ-library for Z39.50 shared cataloging on PHP 5.3 running in Windows 7, using the binary from Apache Lounge that is compatible with VC9 version of PHP. Note that one has to re-compile with other binaries in case another PHP-version (non-VC9) is used.

2.10.2.3.1. Install a C++ compiler: Visual Studio C++ Express 2010

Download Visual Studio C++ Express 2010 from http://www.microsoft.com/visualstudio/eng/downloads. Click on the downloaded application file to install. The installation has a graphical wizard that will guide you through the process. Use the following screen captures as guide in your installation



Opening vc_web.exe

You have chosen to open

vc_web.exe

which is a: Binary File (3.2 MB)

from: http://download.microsoft.com

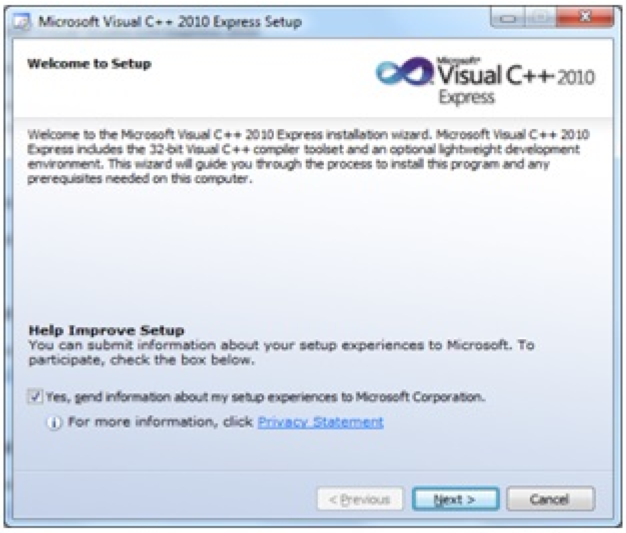
What should Firefox do with this file?

DownThemAll!

dTa OneClick!

[✓ Save File]

Do this automatically for files like this from now on.


Microsoft Visual C++ 2010 Express Setup

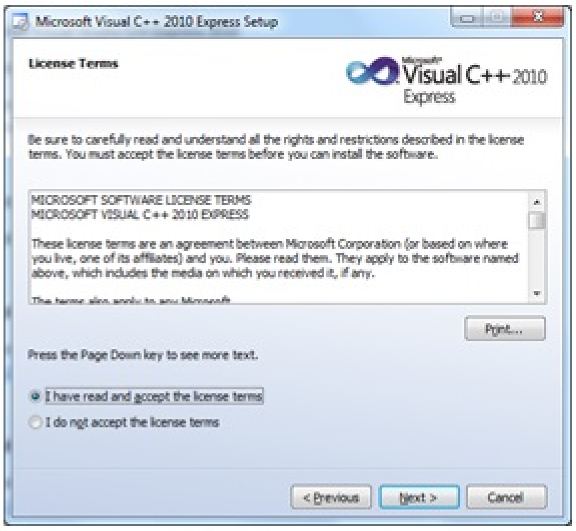
Welcome to Setup

Welcome to the Microsoft Visual C++ 2010 Express installation wizard. Microsoft Visual C++ 2010 Express includes the 32-bit Visual C++ compiler toolset and an optional lightweight development environment. This wizard will guide you through the process to install this program and any prerequisites needed on this computer.

Help Improve Setup
You can submit information about your setup experiences to Microsoft. To participate, check the box below.

[✓] Yes, send information about my setup experiences to Microsoft Corporation.

For more information, click Privacy Statement


Microsoft Visual C++ 2010 Express Setup

License Terms

Be sure to carefully read and understand all the rights and restrictions described in the license terms. You must accept the license terms before you can install the software.

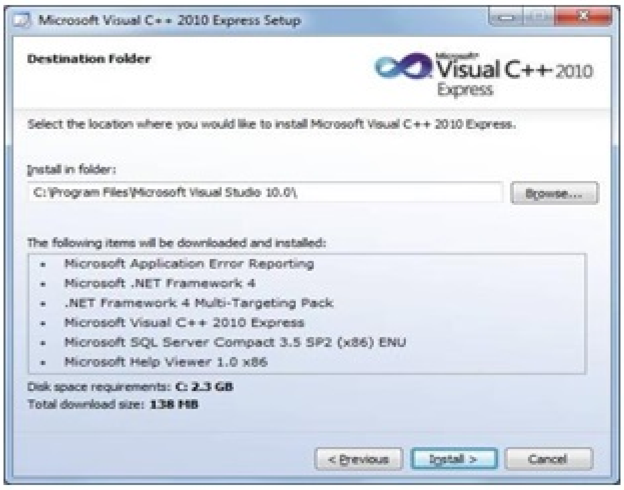
MICROSOFT SOFTWARE LICENSE TERMS
MICROSOFT VISUAL C++ 2010 EXPRESS

These license terms are an agreement between Microsoft Corporation (or based on where you live, one of its affiliates) and you. Please read them. They apply to the software named above, which includes the media on which you received it, if any.

The terms also apply to any Microsoft

Press the Page Down key to see more text.

[✓ I have read and accept the license terms]
I do not accept the license terms


Microsoft Visual C++ 2010 Express Setup

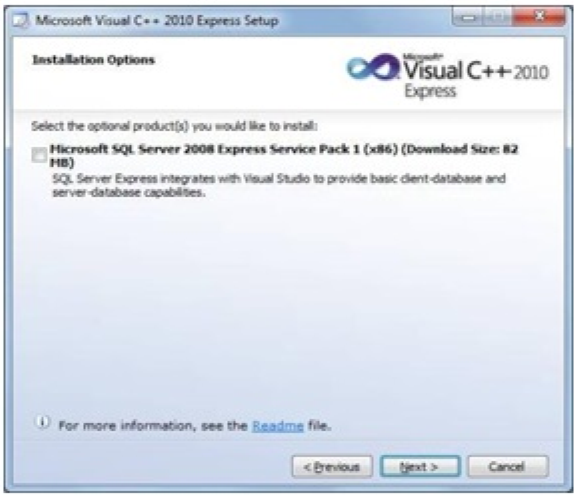
Destination Folder

Select the location where you would like to install Microsoft Visual C++ 2010 Express.

Install in folder:
C:\Program Files\Microsoft Visual Studio 10.0\

The following items will be downloaded and installed:
- Microsoft Application Error Reporting
- Microsoft .NET Framework 4
- .NET Framework 4 Multi-Targeting Pack
- Microsoft Visual C++ 2010 Express
- Microsoft SQL Server Compact 3.5 SP2 (x86) ENU
- Microsoft Help Viewer 1.0 x86

Disk space requirements: **C: 2.3 GB**
Total download size: **138 MB**

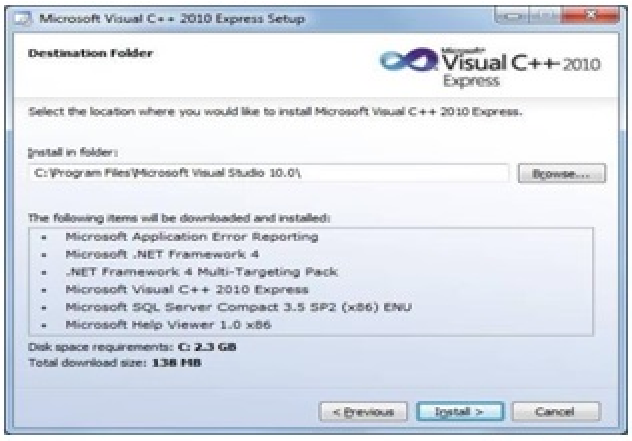

Microsoft Visual C++ 2010 Express Setup

Installation Options

Select the optional product(s) you would like to install:

[ ] **Microsoft SQL Server 2008 Express Service Pack 1 (x86) (Download Size: 82 MB)**
SQL Server Express integrates with Visual Studio to provide basic client-database and server-database capabilities.

For more information, see the Readme file.



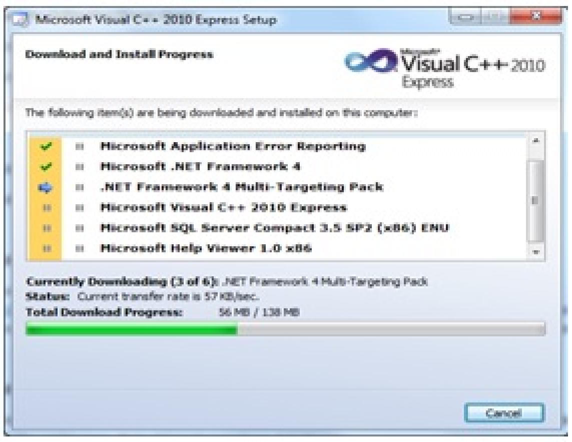
Destination Folder

Select the location where you would like to install Microsoft Visual C++ 2010 Express.

Install in folder:
C:\Program Files\Microsoft Visual Studio 10.0\

The following items will be downloaded and installed:
- Microsoft Application Error Reporting
- Microsoft .NET Framework 4
- .NET Framework 4 Multi-Targeting Pack
- Microsoft Visual C++ 2010 Express
- Microsoft SQL Server Compact 3.5 SP2 (x86) ENU
- Microsoft Help Viewer 1.0 x86

Disk space requirements: **C: 2.3 GB**
Total download size: **138 MB**

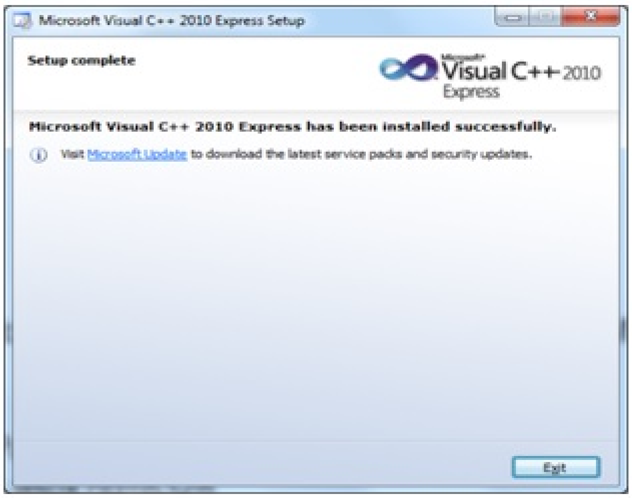


Download and Install Progress

The following item(s) are being downloaded and installed on this computer:

- ✓ Microsoft Application Error Reporting
- ✓ Microsoft .NET Framework 4
- .NET Framework 4 Multi-Targeting Pack
- Microsoft Visual C++ 2010 Express
- Microsoft SQL Server Compact 3.5 SP2 (x86) ENU
- Microsoft Help Viewer 1.0 x86

Currently Downloading (3 of 6): .NET Framework 4 Multi-Targeting Pack
Status: Current transfer rate is 57 KB/sec.
Total Download Progress: 56 MB / 138 MB



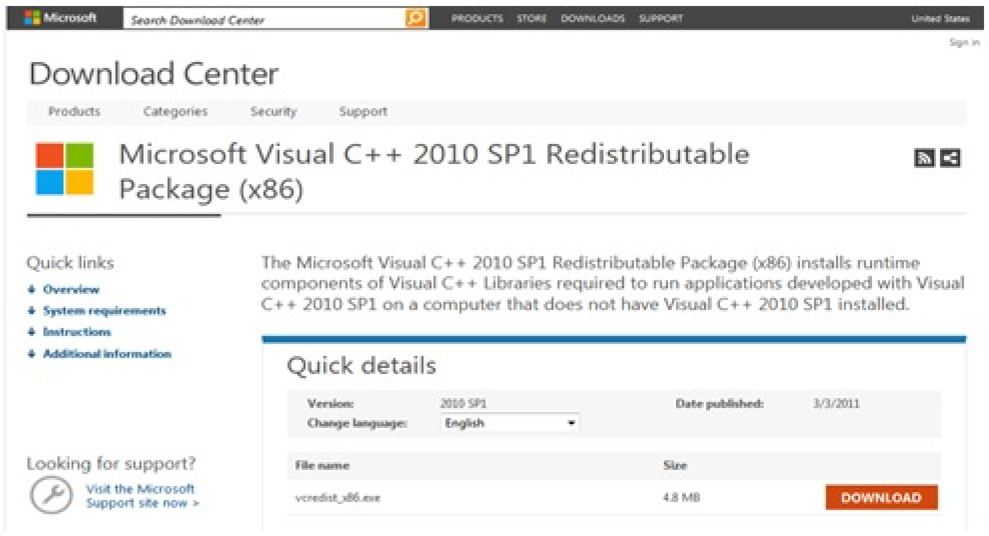
Setup complete

Microsoft Visual C++ 2010 Express has been installed successfully.

Visit Microsoft Update to download the latest service packs and security updates.

2.10.2.3.2. Install Windows C++ runtime: Microsoft Visual C++ 2010 SP1

Download Microsoft Visual C++ 2010 SP1 Redistributable Package (x86) from http://www.microsoft.com/en-us/download/details.aspx?id=8328 You need to install these to avoid some C library linkage errors during compilation of PHP.


Microsoft | Search Download Center | PRODUCTS STORE DOWNLOADS SUPPORT | United States | Sign in

Download Center

Products Categories Security Support

Microsoft Visual C++ 2010 SP1 Redistributable Package (x86)

Quick links
- Overview
- System requirements
- Instructions
- Additional information

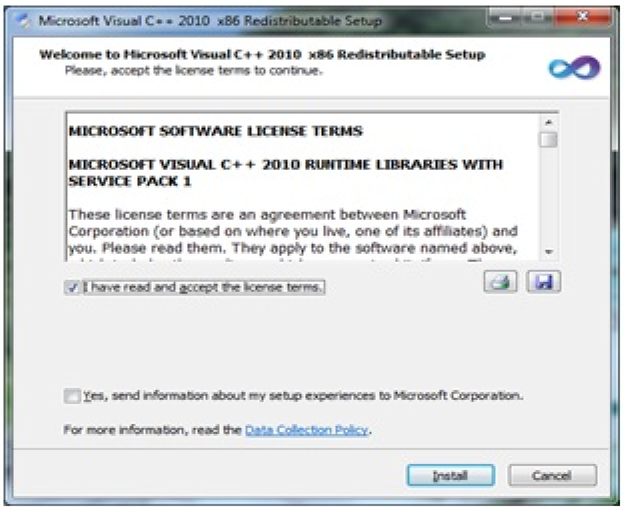
The Microsoft Visual C++ 2010 SP1 Redistributable Package (x86) installs runtime components of Visual C++ Libraries required to run applications developed with Visual C++ 2010 SP1 on a computer that does not have Visual C++ 2010 SP1 installed.

Quick details

Version: 2010 SP1 | Date published: 3/3/2011
Change language: English

File name	Size	
vcredist_x86.exe	4.8 MB	DOWNLOAD

Looking for support?
Visit the Microsoft Support site now >


Microsoft Visual C++ 2010 x86 Redistributable Setup

Welcome to Microsoft Visual C++ 2010 x86 Redistributable Setup
Please, accept the license terms to continue.

MICROSOFT SOFTWARE LICENSE TERMS

MICROSOFT VISUAL C++ 2010 RUNTIME LIBRARIES WITH SERVICE PACK 1

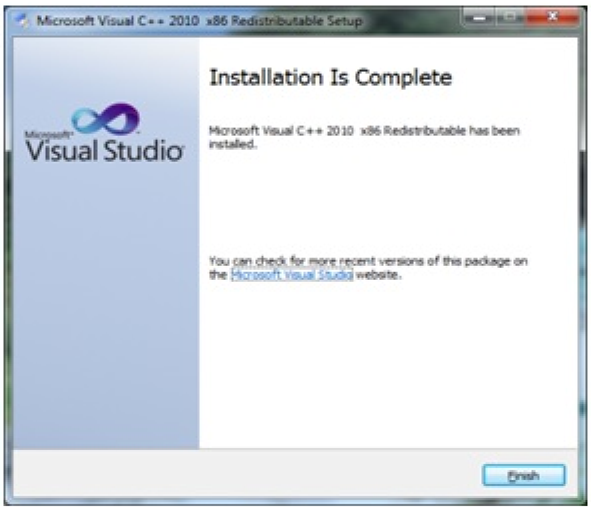
These license terms are an agreement between Microsoft Corporation (or based on where you live, one of its affiliates) and you. Please read them. They apply to the software named above,

[✓] I have read and accept the license terms.

[ ] Yes, send information about my setup experiences to Microsoft Corporation.

For more information, read the Data Collection Policy.

Install Cancel


Microsoft Visual C++ 2010 x86 Redistributable Setup

Installation Is Complete

Microsoft Visual C++ 2010 x86 Redistributable has been installed.

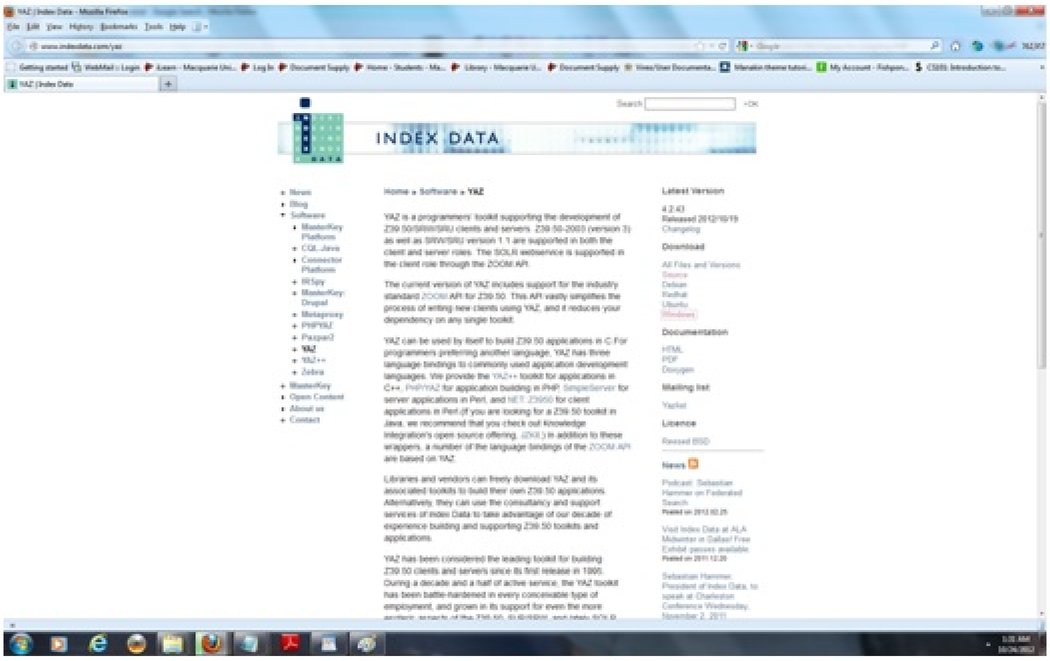
You can check for more recent versions of this package on the Microsoft Visual Studio website.

Finish

2.10.2.3.3. Install YAZ - full with sources

Download the YAZ Windows binary from http://www.indexdata.com/yaz . This is the direct link: http://ftp.indexdata.dk/pub/yaz/win32/yaz_4.2.42.exe

Make sure you install fully with sources to c:\Program Files\YAZ



Home » Software » YAZ

YAZ is a programmers' toolkit supporting the development of Z39.50/SRW/SRU clients and servers. Z39.50-2003 (version 3) as well as SRW/SRU version 1.1 are supported in both the client and server roles. The SOLR webservice is supported in the client role through the ZOOM API.

The current version of YAZ includes support for the industry standard ZOOM API for Z39.50. This API vastly simplifies the process of writing new clients using YAZ, and it reduces your dependency on any single toolkit.

YAZ can be used by itself to build Z39.50 applications in C. For programmers preferring another language, YAZ has three language bindings to commonly used application development languages. We provide the YAZ++ toolkit for applications in C++, PHP/YAZ for application building in PHP, SimpleServer for server applications in Perl, and NET::Z3950 for client applications in Perl (if you are looking for a Z39.50 toolkit in Java, we recommend that you check out Knowledge Integration's open source offering, JZKit.) In addition to these wrappers, a number of the language bindings of the ZOOM API are based on YAZ.

Libraries and vendors can freely download YAZ and its associated toolkits to build their own Z39.50 applications. Alternatively, they can use the consultancy and support services of Index Data to take advantage of our decade of experience building and supporting Z39.50 toolkits and applications.

YAZ has been considered the leading toolkit for building Z39.50 clients and servers since its first release in 1995. During a decade and a half of active service, the YAZ toolkit has been battle-hardened in every conceivable type of employment, and grown in its support for even the more

Latest Version

4.2.43
Released 2012/10/19
Changelog

Download

All Files and Versions
Source
Debian
Redhat
Ubuntu
Windows

Documentation

HTML
PDF
Doxygen

Mailing list

Yazlist

Licence

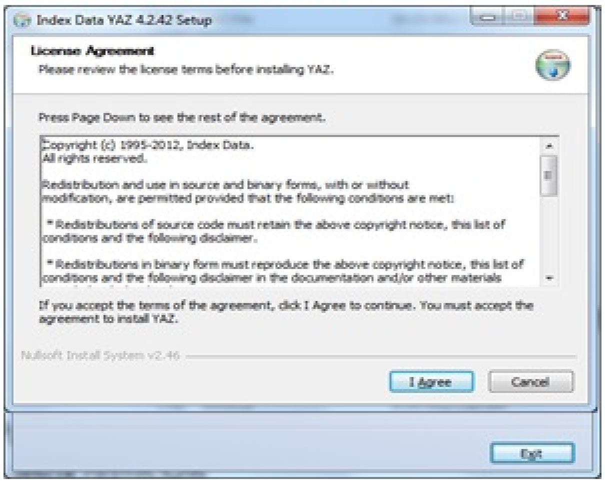
Revised BSD

News

Podcast: Sebastian Hammer on Federated Search
Posted on 2012.02.25

Visit Index Data at ALA Midwinter in Dallas! Free Exhibit passes available
Posted on 2011.12.20

Sebastian Hammer, President of Index Data, to speak at Charleston Conference Wednesday, November 2, 2011



Index Data YAZ 4.2.42 Setup

License Agreement

Please review the license terms before installing YAZ.

Press Page Down to see the rest of the agreement.

Copyright (c) 1995-2012, Index Data.
All rights reserved.

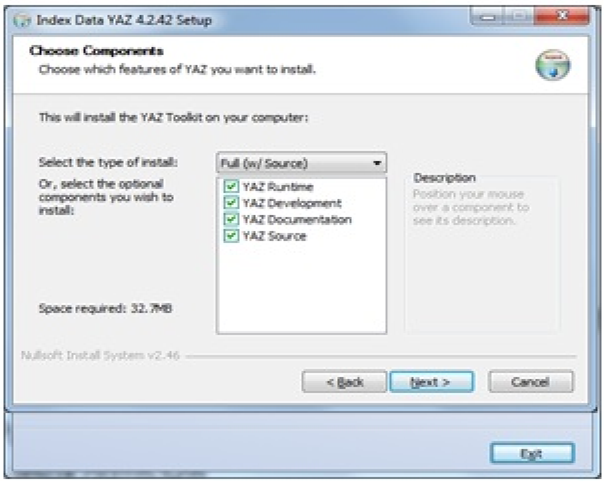
Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

* Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

* Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials

If you accept the terms of the agreement, click I Agree to continue. You must accept the agreement to install YAZ.

Nullsoft Install System v2.46



Index Data YAZ 4.2.42 Setup

Choose Components

Choose which features of YAZ you want to install.

This will install the YAZ Toolkit on your computer:

Select the type of install: Full (w/ Source)

Or, select the optional components you wish to install:

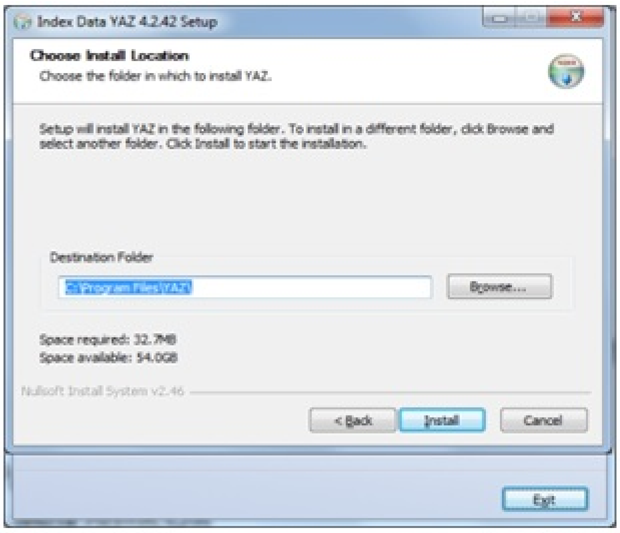
- [✓] YAZ Runtime
- [✓] YAZ Development
- [✓] YAZ Documentation
- [✓] YAZ Source

Description

Position your mouse over a component to see its description.

Space required: 32.7MB

Nullsoft Install System v2.46



Index Data YAZ 4.2.42 Setup

Choose Install Location

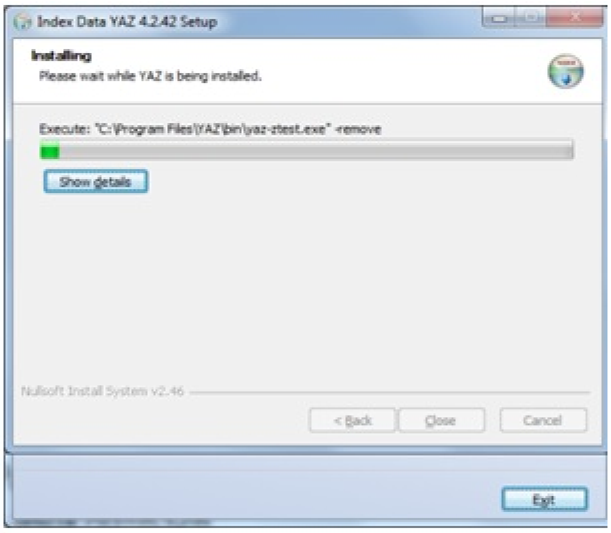
Choose the folder in which to install YAZ.

Setup will install YAZ in the following folder. To install in a different folder, click Browse and select another folder. Click Install to start the installation.

Destination Folder: C:\Program Files\YAZ\

Space required: 32.7MB
Space available: 54.0GB

Nullsoft Install System v2.46



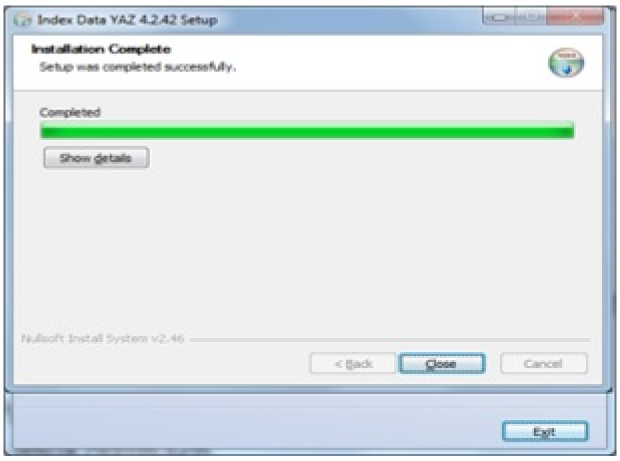
Index Data YAZ 4.2.42 Setup

Installing

Please wait while YAZ is being installed.

Execute: "C:\Program Files\YAZ\bin\yaz-ztest.exe" -remove

Nullsoft Install System v2.46



Index Data YAZ 4.2.42 Setup

Installation Complete

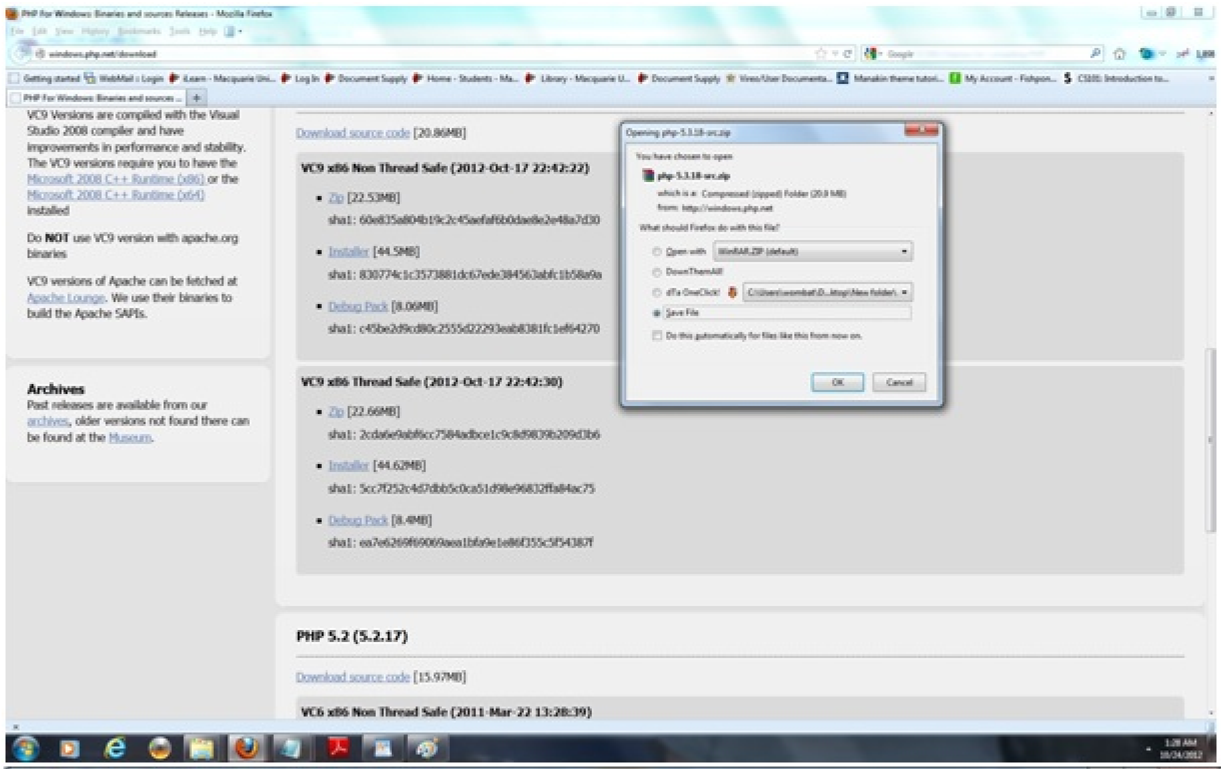
Setup was completed successfully.

Completed

Nullsoft Install System v2.46

2.10.2.3.4. Setup the build environment

Create a directory C:/PHP/dev in which we will run the compilation. Download the PHP sources from http://windows.php.net/download . PHP 5.3.18 version is used in this case but of course this can change with new versions. http://windows.php.net/downloads/releases/php-5.3.18-src.zip Extract the PHP sources (content of php-5.3.18-src.zip) to the new folder C:/PHP/dev.



PHP for Windows: Binaries and sources Releases - Mozilla Firefox

windows.php.net/download

VC9 Versions are compiled with the Visual Studio 2008 compiler and have improvements in performance and stability. The VC9 versions require you to have the Microsoft 2008 C++ Runtime (x86) or the Microsoft 2008 C++ Runtime (x64) installed

Do **NOT** use VC9 version with apache.org binaries

VC9 versions of Apache can be fetched at Apache Lounge. We use their binaries to build the Apache SAPIs.

Archives
Past releases are available from our archives, older versions not found there can be found at the Museum.

Download source code [20.86MB]

VC9 x86 Non Thread Safe (2012-Oct-17 22:42:22)

- Zip [22.53MB]
  sha1: 60e835a804b19c2c45aefaf6b0dae8e2e48a7d30
- Installer [44.5MB]
  sha1: 830774c1c3573881dc67ede384563abfc1b58a9a
- Debug Pack [8.06MB]
  sha1: c45be2d9cd80c2555d22293eab8381fc1ef64270

VC9 x86 Thread Safe (2012-Oct-17 22:42:30)

- Zip [22.66MB]
  sha1: 2cda6e9abf6cc7584adbce1c9c8d9839b209d3b6
- Installer [44.62MB]
  sha1: 5cc7f252c4d7dbb5c0ca51d98e96832ffa84ac75
- Debug Pack [8.4MB]
  sha1: ea7e6269f69069aea1bfa9e1e86f355c5f54387f

PHP 5.2 (5.2.17)

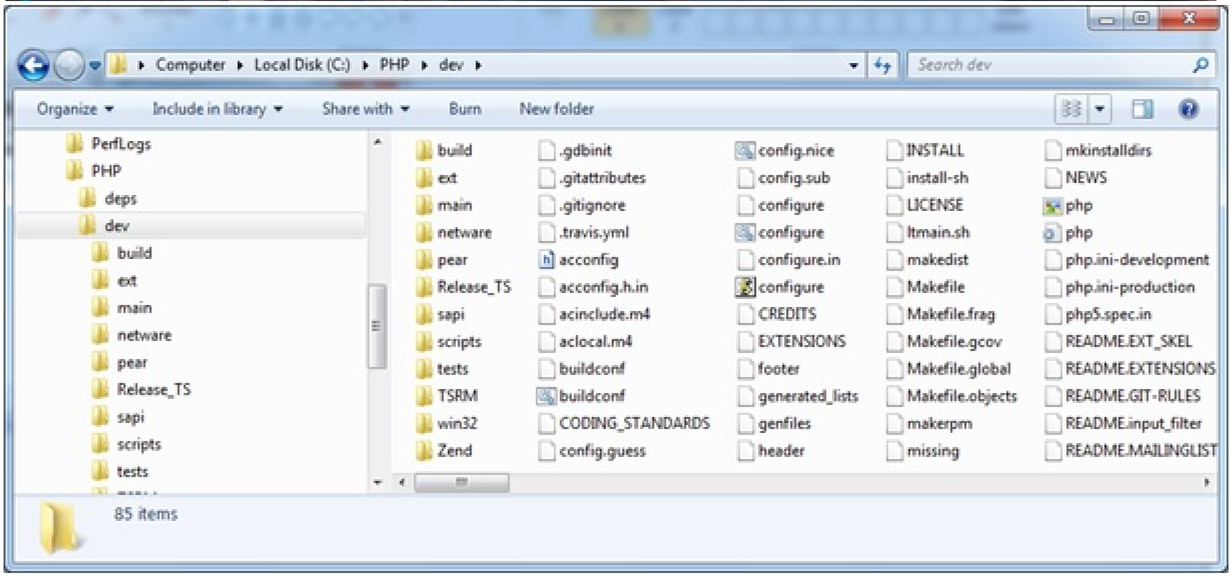
Download source code [15.97MB]

VC6 x86 Non Thread Safe (2011-Mar-22 13:28:39)

Opening php-5.3.18-src.zip

You have chosen to open
php-5.3.18-src.zip
which is a: Compressed (zipped) Folder (20.9 MB)
from: http://windows.php.net

What should Firefox do with this file?
Open with WinRAR.ZIP (default)
DownThemAll!
dTa OneClick! C:\Users\wombat\D...ktop\New folder\
[✓ Save File]
Do this automatically for files like this from now on.



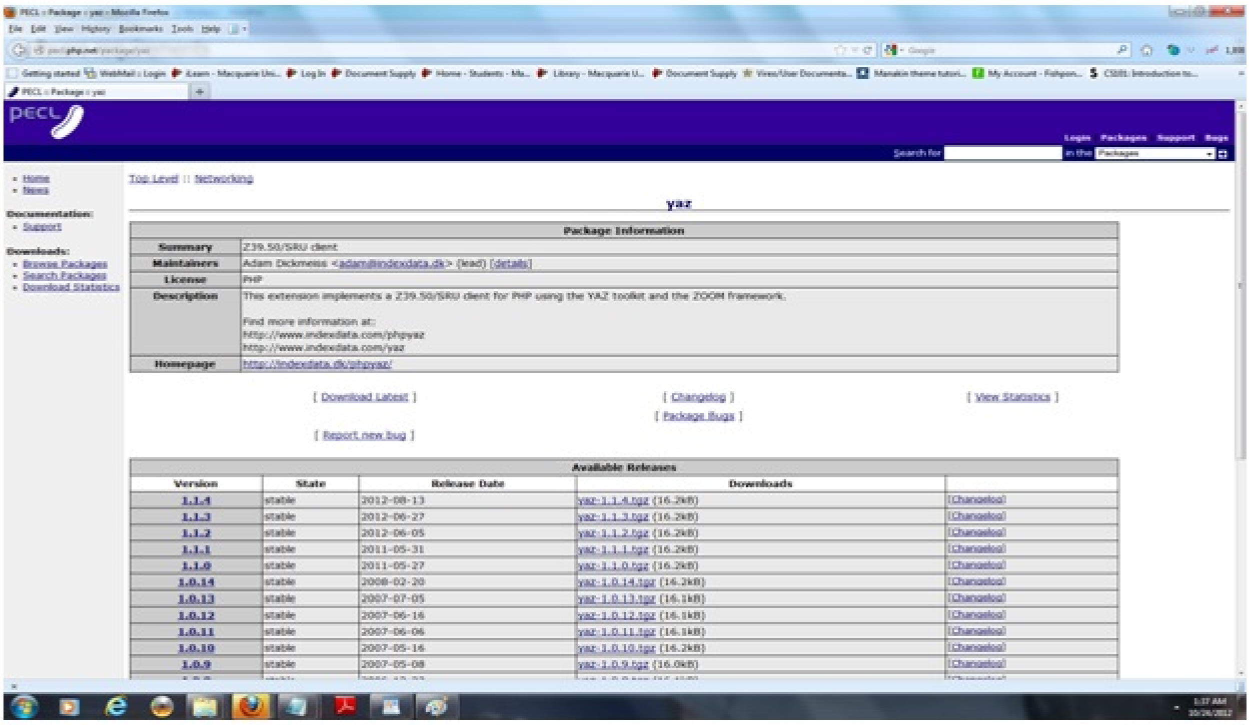
Computer ▸ Local Disk (C:) ▸ PHP ▸ dev ▸

PerfLogs
PHP
deps
dev
build
ext
main
netware
pear
Release_TS
sapi
scripts
tests

build	.gdbinit	config.nice	INSTALL	mkinstalldirs
ext	.gitattributes	config.sub	install-sh	NEWS
main	.gitignore	configure	LICENSE	php
netware	.travis.yml	configure	ltmain.sh	php
pear	acconfig	configure.in	makedist	php.ini-development
Release_TS	acconfig.h.in	configure	Makefile	php.ini-production
sapi	acinclude.m4	CREDITS	Makefile.frag	php5.spec.in
scripts	aclocal.m4	EXTENSIONS	Makefile.gcov	README.EXT_SKEL
tests	buildconf	footer	Makefile.global	README.EXTENSIONS
TSRM	buildconf	generated_lists	Makefile.objects	README.GIT-RULES
win32	CODING_STANDARDS	genfiles	makerpm	README.input_filter
Zend	config.guess	header	missing	README.MAILINGLIST

85 items

Download the PHP_YAZ sources from http://pecl.php.net/package/yaz. This extension implements a Z39.50/SRU client for PHP using the YAZ toolkit and the ZOOM framework. Extract yaz-1.1.4.tgz (latest version) to C:/PHP/dev/ext. Most Windows-tools for unzipping (but not Windows' own tool...), e.g. the free 7-Zip software, also know the '.tgz' archive format, which is typically for Linux, and can extract it like any .zip archive.



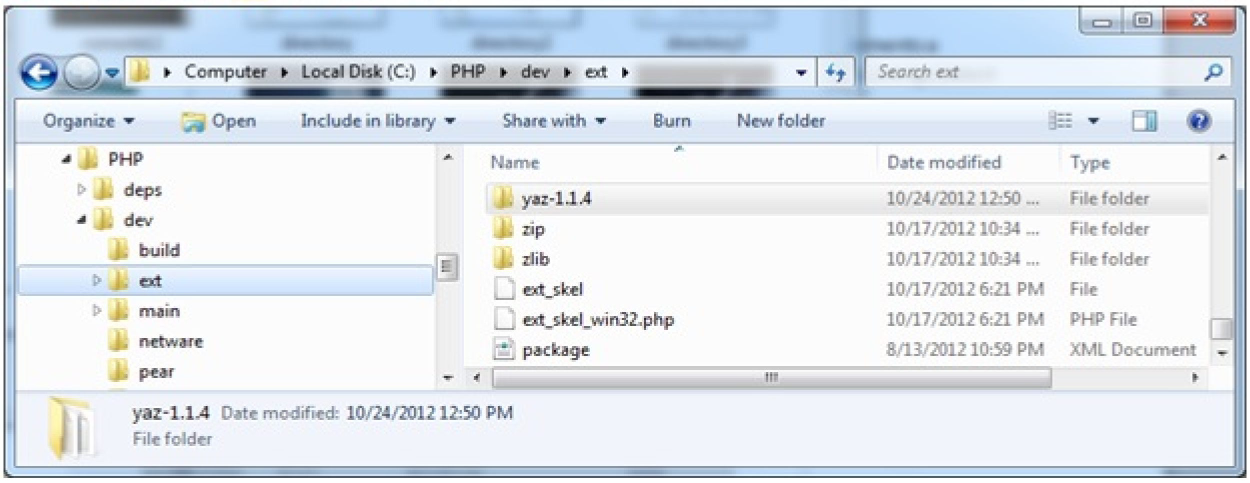
yaz

Package Information	
Summary	Z39.50/SRU client
Maintainers	Adam Dickmeiss <adam@indexdata.dk> (lead) [details]
License	PHP
Description	This extension implements a Z39.50/SRU client for PHP using the YAZ toolkit and the ZOOM framework. Find more information at: http://www.indexdata.com/phpyaz http://www.indexdata.com/yaz
Homepage	http://indexdata.dk/phpyaz/

[Download Latest] [Changelog] [View Statistics]
[Package Bugs]
[Report new bug]

Available Releases

Version	State	Release Date	Downloads	
1.1.4	stable	2012-08-13	yaz-1.1.4.tgz (16.2kB)	[Changelog]
1.1.3	stable	2012-06-27	yaz-1.1.3.tgz (16.2kB)	[Changelog]
1.1.2	stable	2012-06-05	yaz-1.1.2.tgz (16.2kB)	[Changelog]
1.1.1	stable	2011-05-31	yaz-1.1.1.tgz (16.2kB)	[Changelog]
1.1.0	stable	2011-05-27	yaz-1.1.0.tgz (16.2kB)	[Changelog]
1.0.14	stable	2008-02-20	yaz-1.0.14.tgz (16.2kB)	[Changelog]
1.0.13	stable	2007-07-05	yaz-1.0.13.tgz (16.1kB)	[Changelog]
1.0.12	stable	2007-06-16	yaz-1.0.12.tgz (16.1kB)	[Changelog]
1.0.11	stable	2007-06-06	yaz-1.0.11.tgz (16.1kB)	[Changelog]
1.0.10	stable	2007-05-16	yaz-1.0.10.tgz (16.2kB)	[Changelog]
1.0.9	stable	2007-05-08	yaz-1.0.9.tgz (16.0kB)	[Changelog]

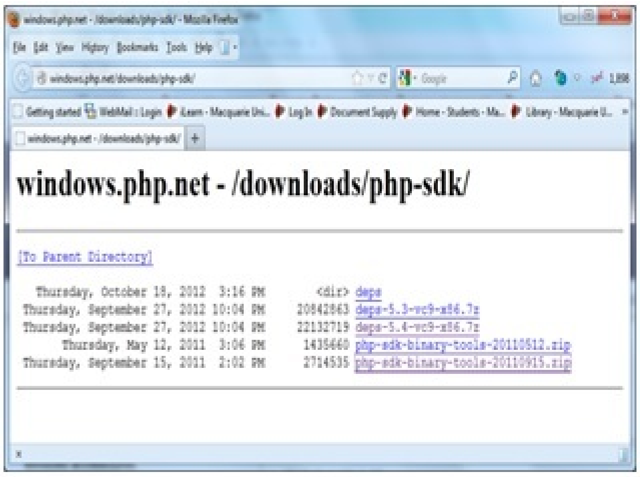


Computer ▸ Local Disk (C:) ▸ PHP ▸ dev ▸ ext ▸

Name	Date modified	Type
yaz-1.1.4	10/24/2012 12:50 ...	File folder
zip	10/17/2012 10:34 ...	File folder
zlib	10/17/2012 10:34 ...	File folder
ext_skel	10/17/2012 6:21 PM	File
ext_skel_win32.php	10/17/2012 6:21 PM	PHP File
package	8/13/2012 10:59 PM	XML Document

yaz-1.1.4 Date modified: 10/24/2012 12:50 PM
File folder

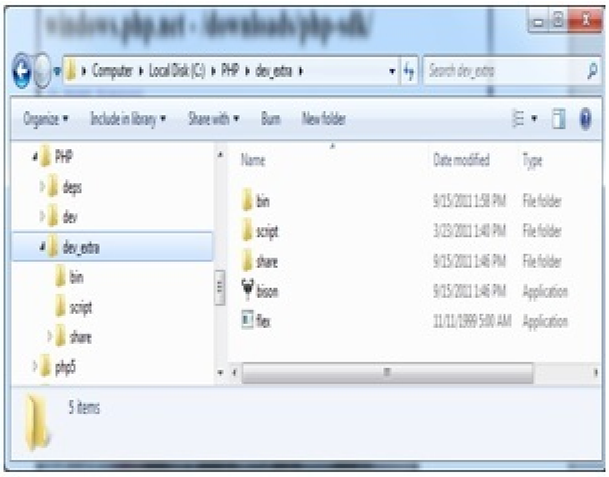
Download the binary tools from http://windows.php.net/downloads/php-sdk and unpack the archive in the C:/PHP/dev_extra directory. The file used in this installation is php-sdk-binary-tools-20110915.zip (http://windows.php.net/downloads/php-sdk/php-sdk-binary-tools-20110915.zip). Change to the subdirectory bin and copy the applications bison.exe and flex.exe to C:/PHP/dev_extra.



windows.php.net - /downloads/php-sdk/

[To Parent Directory]

```
  Thursday, October 18, 2012  3:16 PM        <dir> deps
Thursday, September 27, 2012 10:04 PM     20842863 deps-5.3-vc9-x86.7z
Thursday, September 27, 2012 10:04 PM     22132719 deps-5.4-vc9-x86.7z
       Thursday, May 12, 2011  3:06 PM      1435660 php-sdk-binary-tools-20110512.zip
Thursday, September 15, 2011  2:02 PM      2714535 php-sdk-binary-tools-20110915.zip
```

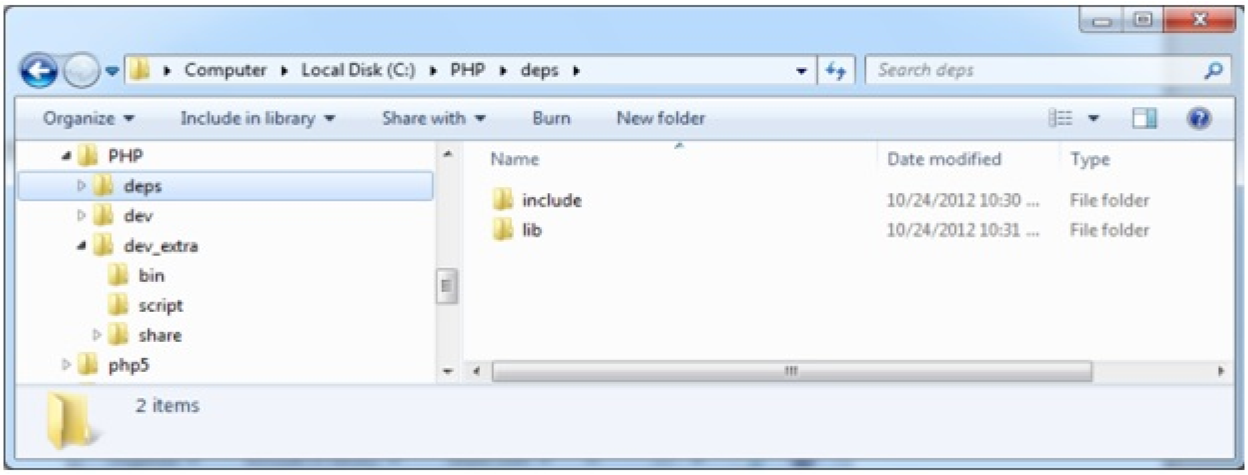


Computer ▸ Local Disk (C:) ▸ PHP ▸ dev_extra ▸

Name	Date modified	Type
bin	9/15/2011 1:58 PM	File folder
script	3/23/2011 1:48 PM	File folder
share	9/15/2011 1:46 PM	File folder
bison	9/15/2011 1:46 PM	Application
flex	11/11/1999 5:00 AM	Application

5 items

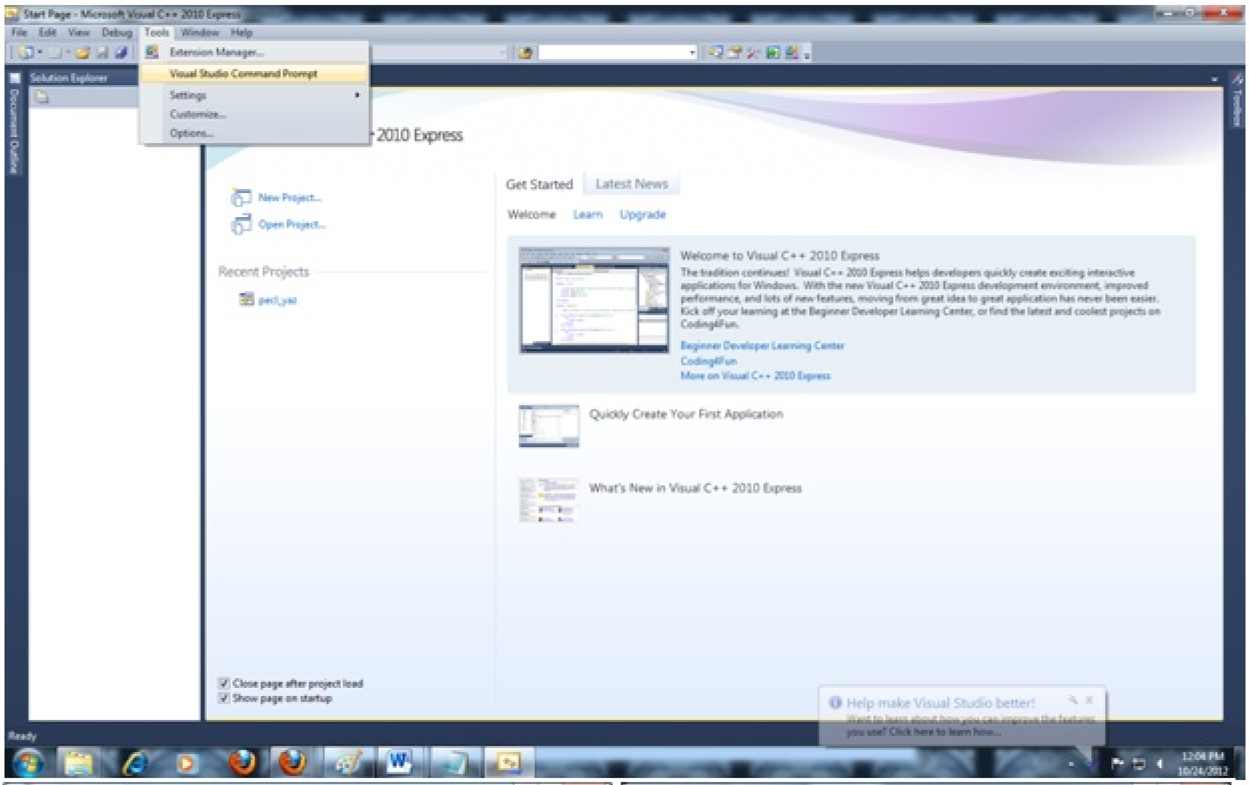
Copy the YAZ libraries folders C:\Program Files\YAZ\lib and C:\Program Files\YAZ \include to C:\PHP\deps



Name	Date modified	Type
include	10/24/2012 10:30 ...	File folder
lib	10/24/2012 10:31 ...	File folder

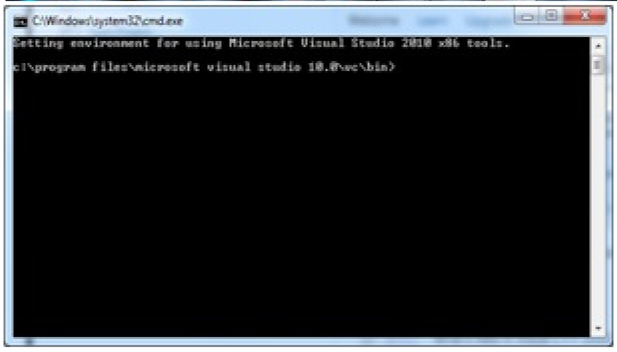
2.10.2.3.5. Build and compile YAZ with PHP using Visual C++ 2010 Express console

Click Start -> All Program -> Microsoft Visual C++ Express 2010 and then click Visual Studio Command Prompt from the 'Tools'-menu. Type the following commands in sequence in the DOS command prompt window that will pop up.

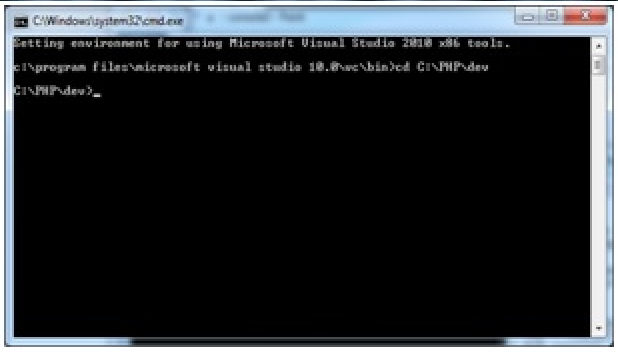

Start Page - Microsoft Visual C++ 2010 Express
File Edit View Debug Tools Window Help
Extension Manager...
Visual Studio Command Prompt
Settings
Customize...
Options...
Solution Explorer
2010 Express
Get Started
Latest News
Welcome
Learn
Upgrade
New Project...
Open Project...
Recent Projects
pecl_yaz
Beginner Developer Learning Center
Coding4Fun
More on Visual C++ 2010 Express
Quickly Create Your First Application
What's New in Visual C++ 2010 Express
Close page after project load
Show page on startup
Help make Visual Studio better!
Ready

Welcome to Visual C++ 2010 Express

The tradition continues! Visual C++ 2010 Express helps developers quickly create exciting interactive applications for Windows. With the new Visual C++ 2010 Express development environment, improved performance, and lots of new features, moving from great idea to great application has never been easier. Kick off your learning at the Beginner Developer Learning Center, or find the latest and coolest projects on Coding4Fun.



```
Setting environment for using Microsoft Visual Studio 2010 x86 tools.

c:\program files\microsoft visual studio 10.0\vc\bin>
```



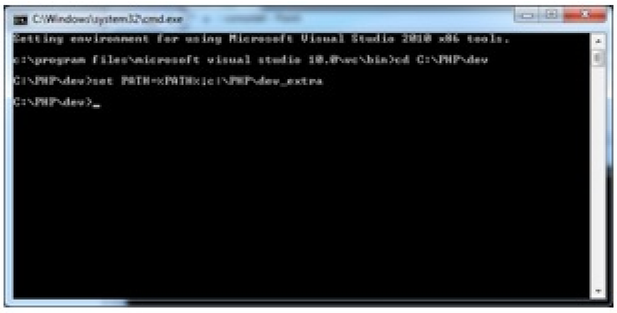
```
Setting environment for using Microsoft Visual Studio 2010 x86 tools.

c:\program files\microsoft visual studio 10.0\vc\bin>cd C:\PHP\dev

C:\PHP\dev>_
```

Change to the PHP source directory :

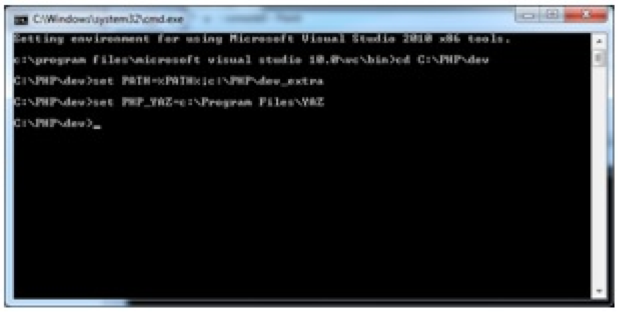
```
cd c:\PHP\dev
```

Set the environment variables :

```
set PATH=%PATH%;c:\PHP\dev_extra

set PHP_YAZ=c:\Program Files\YAZ\
```

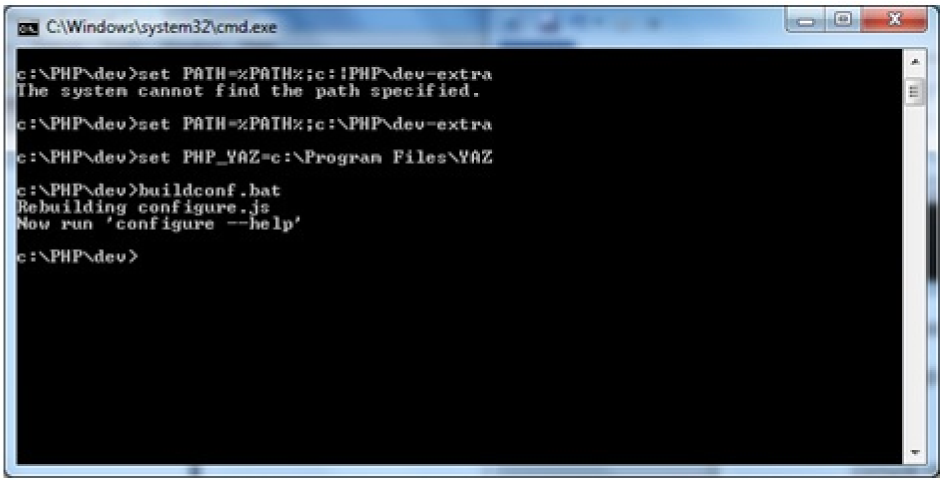


```
Setting environment for using Microsoft Visual Studio 2010 x86 tools.

c:\program files\microsoft visual studio 10.0\vc\bin>cd C:\PHP\dev

C:\PHP\dev>set PATH=%PATH%;c:\PHP\dev_extra

C:\PHP\dev>_
```



```
Setting environment for using Microsoft Visual Studio 2010 x86 tools.

c:\program files\microsoft visual studio 10.0\vc\bin>cd C:\PHP\dev

C:\PHP\dev>set PATH=%PATH%;c:\PHP\dev_extra

C:\PHP\dev>set PHP_YAZ=c:\Program Files\YAZ

C:\PHP\dev>_
```

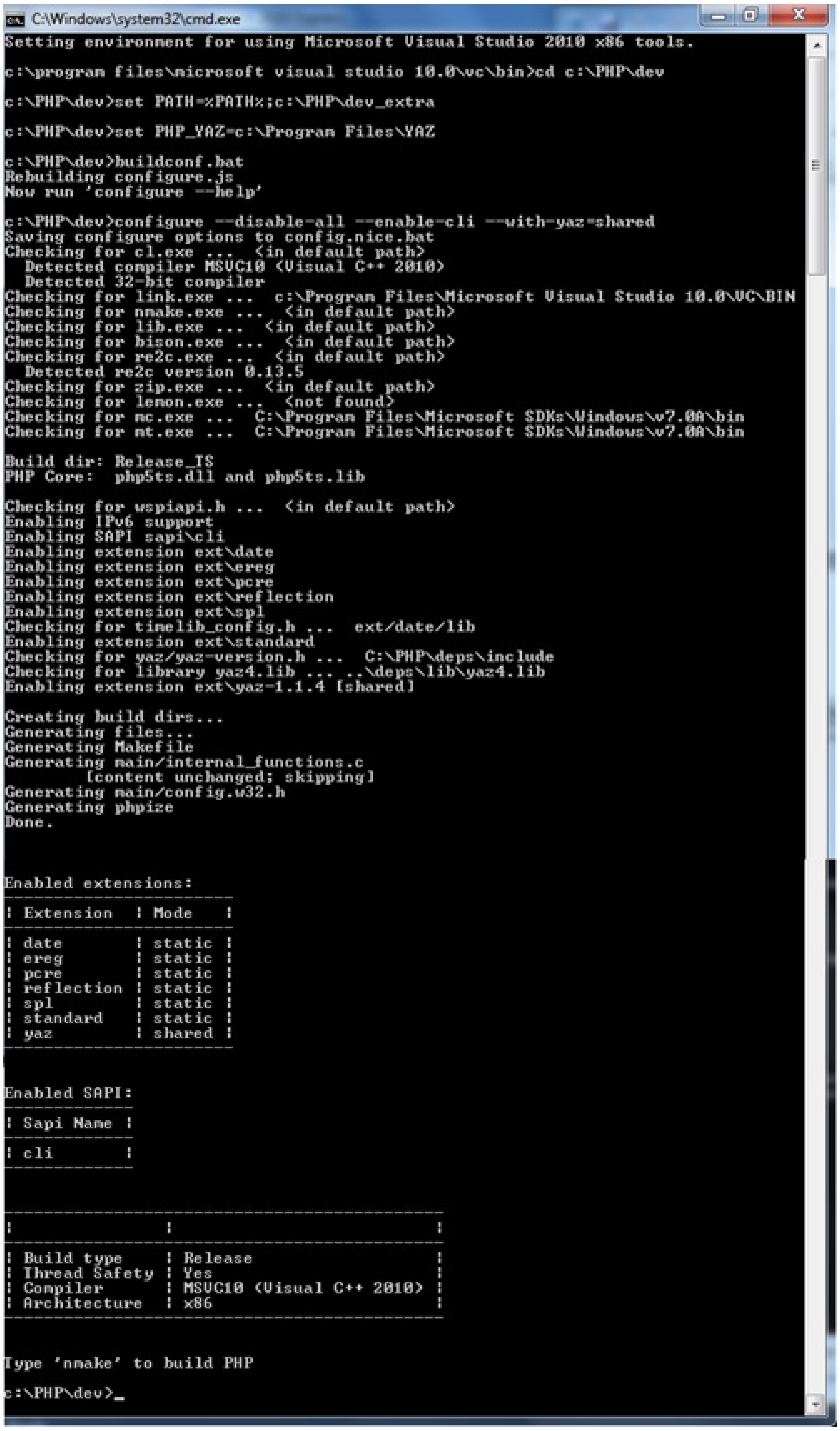
Start building (lookout for possible errors) buildconf.bat



```
c:\PHP\dev>set PATH=%PATH%;c:|PHP\dev-extra
The system cannot find the path specified.

c:\PHP\dev>set PATH=%PATH%;c:\PHP\dev-extra

c:\PHP\dev>set PHP_YAZ=c:\Program Files\YAZ

c:\PHP\dev>buildconf.bat
Rebuilding configure.js
Now run 'configure --help'

c:\PHP\dev>
```

Compile YAZ :

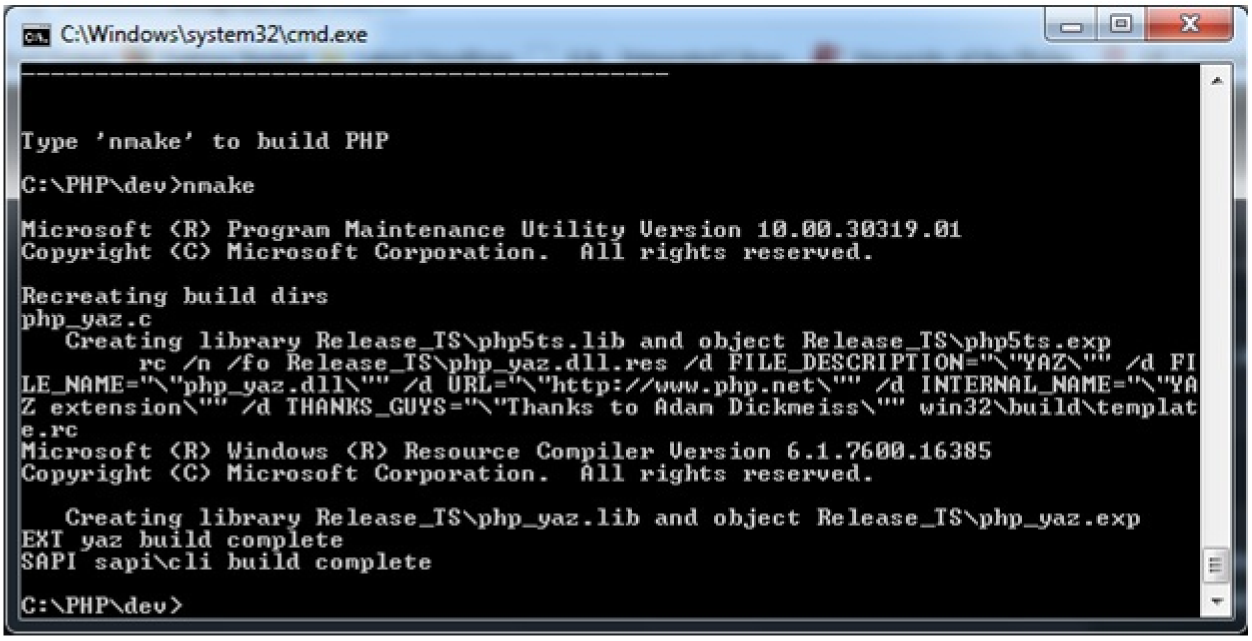
```
configure --disable-all --enable-cli --with-yaz=shared
```



```
Setting environment for using Microsoft Visual Studio 2010 x86 tools.

c:\program files\microsoft visual studio 10.0\vc\bin>cd c:\PHP\dev

c:\PHP\dev>set PATH=%PATH%;c:\PHP\dev_extra

c:\PHP\dev>set PHP_YAZ=c:\Program Files\YAZ

c:\PHP\dev>buildconf.bat
Rebuilding configure.js
Now run 'configure --help'

c:\PHP\dev>configure --disable-all --enable-cli --with-yaz=shared
Saving configure options to config.nice.bat
Checking for cl.exe ...  <in default path>
  Detected compiler MSVC10 (Visual C++ 2010)
  Detected 32-bit compiler
Checking for link.exe ...  c:\Program Files\Microsoft Visual Studio 10.0\VC\BIN
Checking for nmake.exe ...  <in default path>
Checking for lib.exe ...  <in default path>
Checking for bison.exe ...  <in default path>
Checking for re2c.exe ...  <in default path>
  Detected re2c version 0.13.5
Checking for zip.exe ...  <in default path>
Checking for lemon.exe ...  <not found>
Checking for mc.exe ...  C:\Program Files\Microsoft SDKs\Windows\v7.0A\bin
Checking for mt.exe ...  C:\Program Files\Microsoft SDKs\Windows\v7.0A\bin

Build dir: Release_TS
PHP Core:  php5ts.dll and php5ts.lib

Checking for wspiapi.h ...  <in default path>
Enabling IPv6 support
Enabling SAPI sapi\cli
Enabling extension ext\date
Enabling extension ext\ereg
Enabling extension ext\pcre
Enabling extension ext\reflection
Enabling extension ext\spl
Checking for timelib_config.h ...  ext/date/lib
Enabling extension ext\standard
Checking for yaz/yaz-version.h ...  C:\PHP\deps\include
Checking for library yaz4.lib ... ..\deps\lib\yaz4.lib
Enabling extension ext\yaz-1.1.4 [shared]

Creating build dirs...
Generating files...
Generating Makefile
Generating main/internal_functions.c
        [content unchanged; skipping]
Generating main/config.w32.h
Generating phpize
Done.



Enabled extensions:
-----------------------
| Extension  | Mode   |
-----------------------
| date       | static |
| ereg       | static |
| pcre       | static |
| reflection | static |
| spl        | static |
| standard   | static |
| yaz        | shared |
-----------------------


Enabled SAPI:
-------------
| Sapi Name |
-------------
| cli       |
-------------


--------------------------------------------
|               |                          |
--------------------------------------------
| Build type    | Release                  |
| Thread Safety | Yes                      |
| Compiler      | MSVC10 (Visual C++ 2010) |
| Architecture  | x86                      |
--------------------------------------------


Type 'nmake' to build PHP

c:\PHP\dev>_
```

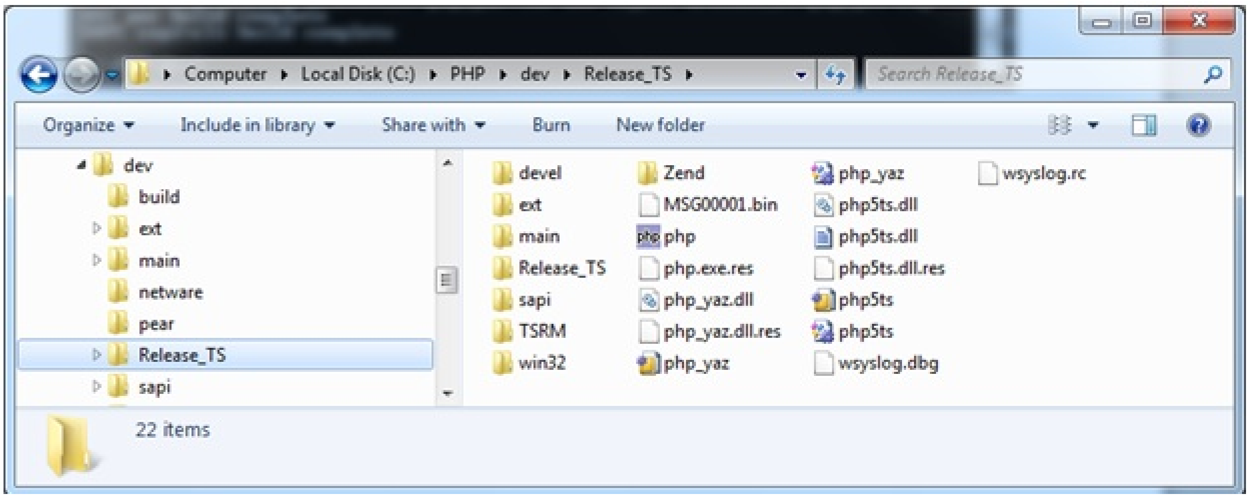
Finally build the binary with the command:

```
nmake
```



```
C:\Windows\system32\cmd.exe
--------------------------------------------


Type 'nmake' to build PHP

C:\PHP\dev>nmake

Microsoft (R) Program Maintenance Utility Version 10.00.30319.01
Copyright (C) Microsoft Corporation.  All rights reserved.

Recreating build dirs
php_yaz.c
   Creating library Release_TS\php5ts.lib and object Release_TS\php5ts.exp
        rc /n /fo Release_TS\php_yaz.dll.res /d FILE_DESCRIPTION="\"YAZ\"" /d FI
LE_NAME="\"php_yaz.dll\"" /d URL="\"http://www.php.net\"" /d INTERNAL_NAME="\"YA
Z extension\"" /d THANKS_GUYS="\"Thanks to Adam Dickmeiss\"" win32\build\templat
e.rc
Microsoft (R) Windows (R) Resource Compiler Version 6.1.7600.16385
Copyright (C) Microsoft Corporation.  All rights reserved.

   Creating library Release_TS\php_yaz.lib and object Release_TS\php_yaz.exp
EXT yaz build complete
SAPI sapi\cli build complete

C:\PHP\dev>
```

After compilation without any errors you will find the compiled DLL in C:\PHP\dev\Release_TS



Computer ▸ Local Disk (C:) ▸ PHP ▸ dev ▸ Release_TS ▸

dev: build, ext, main, netware, pear, Release_TS, sapi

devel, ext, main, Release_TS, sapi, TSRM, win32, Zend, MSG00001.bin, php, php.exe.res, php_yaz.dll, php_yaz.dll.res, php_yaz, php_yaz, php5ts.dll, php5ts.dll, php5ts.dll.res, php5ts, php5ts, wsyslog.dbg, wsyslog.rc

22 items

You can now test it by putting the DLL php_yaz.dll into the correct 'extensions'-folder of your PHP-installation (see php.ini).

2.10.3. Installation in Linux

We briefly explain the steps for compilation of YAZ under Linux Debian here.

2.10.3.1. Installing the Yaz PHP plugin and dependencies.

```
apt-get install yaz libyaz4-dev php5-dev php-pear php5-xsl make

pecl install yaz
```

2.10.3.2. create : /etc/php5/conf.d/yaz.ini

Create this text-file with a text-editor (e.g. nano or gedit) and put in it the following content :

```
extension=yaz.so
```

2.10.3.3. restart Apache

E.g. with the command :

```
apache2ctl restart

OR : /etc/init.d/apache2 restart
```

2.10.3.4. Optional additional steps

You'll also need File_MARC:

```
pear install Structures_LinkedList

pear install File_MARC
```

You may need to specify versions if pear fails with 'beta channel' messages:

```
pear install Structures_LinkedList-0.2.0

pear install File_MARC-0.4.2
```

Fine-tuning Depending on your local conditions, e.g. using 32-bits or 64-bits Linux, the corresponding cgi-bin folder has to be identified: if using 32-bits : rename /opt/ABCD/www/cgi-bin to cgi-bin64 and rename /opt/ABCD/www/cgi-bin32 to cgi-bin (this will activate the 32-bits version of CISIS and set the 64-bits version aside).

2.11. Apache

Apache is the name of the webserver software very frequently used in 'open source' webservers. In fact we are talking about a software called 'HTTPD', which is only one product of the powerful 'Apache Software Foundation', which also provides other interesting products such as e.g. Lucene indexing (also going to be used in the next releases of ABCD), TomCat (a Java Servlet and Server Pages server) and the Derby DB.

Apache as a webserver seems to be the most widely used in the actual Internet, which is one of the (few) examples where FOSS dominates over the commercial solutions offered. All information on the Apache web-server and download files can be found at the URL : http://www.apache.org.

In many cases the Apache webserver software will already be installed on the server where ABCD will reside, as is probably also the case with PHP (and MySQL). For this reason ABCD will come as a package both inclusive of Apache and PHP and another one without these. In the case of an existing Apache webserver, expertise on Apache should be available in order to integrate ABCD with the existing Apache-based applications. E.g. a virtual server for ABCD could be set up with 'aliases' specifically for the ABCD system (htdocs home) and cgi (scripts folder). In the case of the full package, the latest 'stable' Apache httpd-version will be included, pre-configured to work with ABCD as 'localhost' (which means : the PC itself runs both the client and the server). A small script will launch the httpd (or Apache) service based on that configuration, so that installation and configuration efforts in principle can be kept to a strict minimum. In case of additional configuration still to be necessary, the user should be fully aware of the fact that Apache, as a Linux-based software, is case-sensitive for its parameters and file names (with path information) !
In the case of webservers, we should mention 'IIS' (Internet Information Services) of Microsoft, free software but not open, which is the webserver coming with Windows. Differences are mostly in the way how it should/can be configured and managed, rather than performance, security etc... The terminology is a bit different (e.g. 'aliases' are called 'virtual folders' and there is no easily approachable ASCII-configuration file as with Apache and its 'httpd.conf'.
ABCD runs perfectly with IIS, as with other web-server software (e.g. Xitami), but this manual does not support the implementation on IIS.

3. ABCD installation

3.1. Available installation versions

ABCD version 1.2t or 2.0 comes in the following versions :

1. non-assisted installation package for Windows : this is a ZIP-file containing all necessary files, which simply need to be unzipped into the root of (one of your) harddisk(s), e.g. C:\ or D:\. Since it encludes both Apache and PHP with their own ABCD-specific configuration files, after simply unzipping it should normally work ! Apache and PHP both come with their own configuration file (httpd.conf and php.ini) where port 9090 is activated to allow ABCD running next to possible other running Apache installations. Only for some specific uses, e.g. when there is a need to install additional extensions for PHP, it will be necessary to do some editing (of e.g. php.ini).

   ABCD installations from this package which will use the Advanced Loans module (EmpWeb) need to additionally unpack the most recent EmpWeb....zip package, where an extra main subfolder in \ABCD will be created and some additional files will be added to the ABCD Central directory.

   As from version 1.2transitional (and v2.0) ABCD comes also with the YAZ-module (for Z39.50 shared cataloging) enabled and active, but in some cases where the pre-installed YAZ is not compatible with the exact Apache-version (e.g. there are thread-safe and non-thread-safe versions, differently compiled), manual editing will be necessary. For YAZ we refer to a dedicated set of instructions in the YAZ-chapter of this manual.

   Summing up, the extra non-standard PHP-extensions coming with this installer are YAZ, APC, XSL and GD. For manual installations of ABCD these modules have to be added to the PHP configuration, meaning : downloading the extension's DLL, copying them into the 'ext'-folder of your PHP and activating (or adding) the lines in php.ini in the format 'extension=', e.g. 'extension=yaz.dll'.

   After adding extensions in PHP always Apache has to be restarted for the extensions to be loaded.

2. installation wizard for Windows : this is a self-installing executable which will first unpack the main folders of ABCD (including Apache and PHP) and then present a series of dialog boxes to create the correct configuration files. This requires mainly following the dialogs and instructions of the installer itself. This installer allows installation anywhere on the hard-disk, as defined by the user, with any specified http-port (default suggestion is 9090), responsible institution name, default language and OPAC-catalog.

   This installer creates the main configuration files based on the options given in the dialog boxes :

   - config.php for the Central module (in htdocs/central)

   - iah.def.php for the iAH module (in htdocs/iah/scripts) for paths and index.php (in htdocs/iah) for default database and language

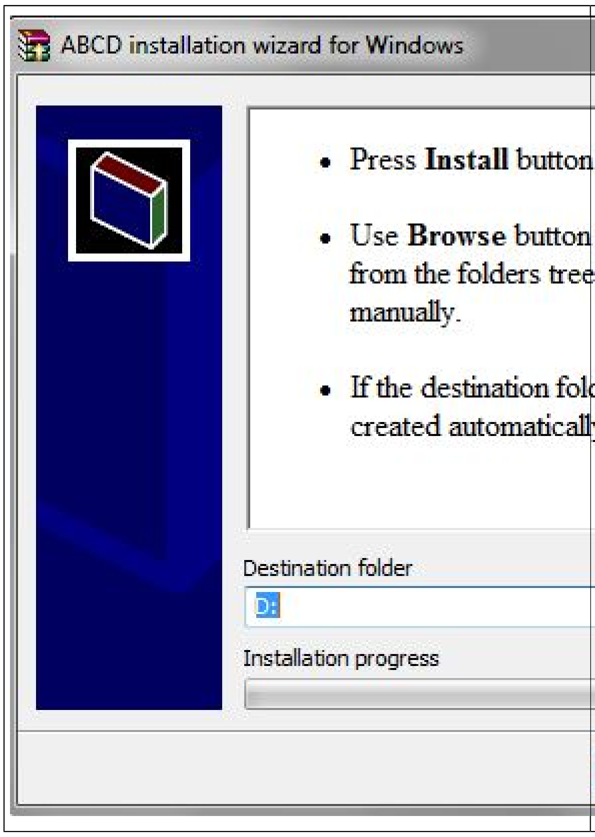
   - bvs-conf-def.php for the Site (in htdocs/site) for default language and paths.

   Remember that these files are simple text-files which can also be edited directly under your OS with a file-manager and a text-editor (e.g. Notepad for Windows or nano for Linux).

   This version of ABCD comes with a 'cache optimizer' for PHP : APC pre-installed, i.e. as an extension php_apc.dll in the folder ABCD/php/ext and listed in the list of extensions in php.ini.

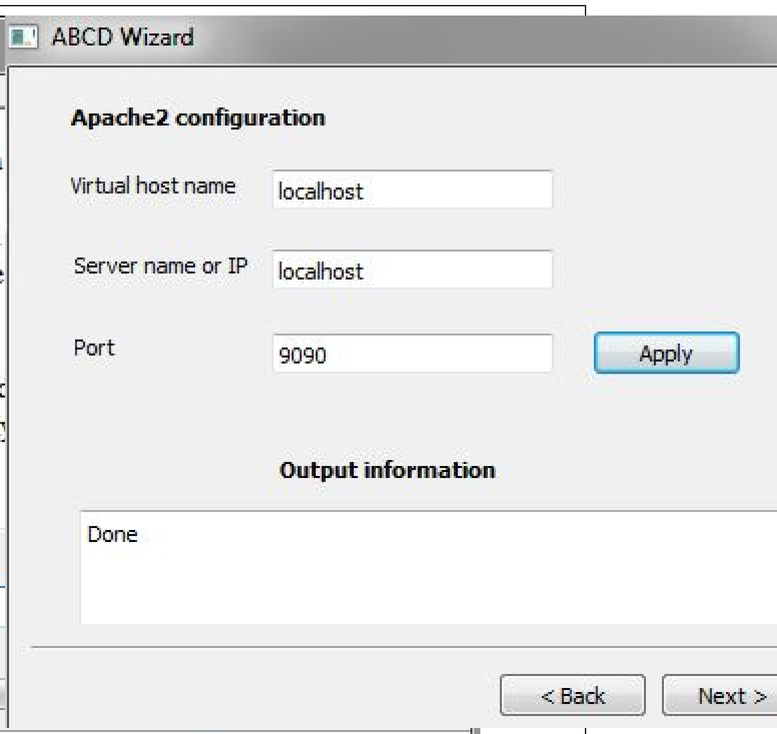
   This version also comes with EmpWeb Advanced Loans pre-configured, only leaving Java and MySQL to be installed separately. So this installer is the easiest and most complete ABCD-package.

Table 1.2. The ABCD Installer for Windows



- Press **Install** button
- Use **Browse** button from the folders tree manually.
- If the destination fold created automatically

D:



Apache2 configuration

Virtual host name: localhost

Server name or IP: localhost

Port: 9090

Output information

Done

3. non-assisted installation package for Linux : this is a .tar.gz archive for Linux systems which should be unpacked into the Linux file-system, depending on its organisation (definition on where such applications can be put). If the correct access-rights are granted (with the appropriate Linux-commands such as chown and chmod) ABCD can be installed, like in Windows, under the file-system root '/'. In Linux systems the assumption is that Apache and PHP are installed separately (e.g. with the dedicated tool like apt-get or Synaptic), so the package only contains the proper ABCD files in the 'www'-directory. Configuration of Apache (in /etc/apache2) with a virtual host (in /etc/apache2/sitesenabled) is to be done manually by someone who understands the configuration of Apache.

4. A Debian package, either for 32 or 64-bits, which has to be downloaded and then installed using the '*dpkg -i*' command in the terminal console. Since Ubuntu and Linux Mint are highly Debian-based we tested installation in these OS'es successfully, in addition to the Debian Wheezy (7) environment. The installation script of this package will check some dependencies and consequently install ABCD in the default (new) location for Linux :

   - all scripts will go inside a 'www'-folder as a subfolder of a folder ABCD, which in itself is a subfolder of /opt
   - all databases will go inside a 'bases'-folder as a subfolder of a folder ABCD, which in itself is a subfolder of /var/opt. A symbolic link to this bases-folder will be present in /opt/ABCD/www, as the module 'secs-web' necessitates the bases-folder to be a subfolder of the www-folder. Check this and if not present, create the link yourself with the command, given from the terminal when in the directory '/opt/ABCD/www' - note the . (dot) at the end ! : **ln -s /var/opt/ABCD/bases .**

5. The installation will check if there are already databases in the directory /var/opt/ABCD, and only if NOT so, it will copy the folder 'bases_examples' to /var/opt/ABCD. So this means that existing databases in that directory won't be over-written. The directory (or folder) 'bases_examples' remains available anyway in the directory /opt/ABCD/www and using the (new) feature of multiple database-directories (see infra) one can easily switch from e.g. demo-databases to operational databases.

   Make sure that your htdocs-directory has the access rights of '775' (full access except writing to non-members of the owner group) and the bases-directory (/var/opt/ABCD and /opt/ABCD/www/bases_examples) the rights of '777' (full access to everybody). Setting access rights in Linux is done by issuing the following commands at the terminal prompt from the level in which the directory to set is present :

(note : sudo is only for Ubuntu and Mint, in Debian use 'su', -R is to involve all subfolders recursively)

```
sudo chmod -R 775 htdocs

sudo chmod -R 777 bases
```

The 'ownership' of the files has to be put to the name under which your Apache is operating, mostly 'www-data', with a command like :

```
chown -R www-data:www-data ABCD
```

A package for RedHat-based systems (Redhat, Fedora...) is in the planning but not yet ready as of the writing of this manual.

3.2. Installation issues

This section deals with some specific installation issues for ABCD. In principle the installation is quite straightforward, either involving simply unpacking the downloaded archives into their proper folder (e.g. in Windows the root of a disk-drive) or correctly running the installation wizard (in Windows) or installing the Debian-package. Since ABCD has several totally different components, installation by definition encompasses quite some potential pitfalls. Three main reasons can be given for the installation to be complex :

1. ABCD is a combination of several software technologies : ISIS-databases, ISIS-scripts and ISIS-formats, a webserver, PHP-scripting, plus (in the case of the advanced Loans module) some JAVA and MySQL parts;

2. being web-based, which means a web-server has to be installed and special measures have to be taken about access rights : in principle the whole world - with access to the WWW - can interfere.

3. ABCD will be installed in quite different situations, varying from a simple stand-alone (even non-networked) PC up to servers in big networks with a webserver and often also PHP-scripting services already pre-installed.

Currently the installation packages can come in two types :

1. a full package, containing all ABCD-proper files plus the Apache webserver and PHP-scripting engine.

   In this situation an archive (.zip or .tar.gz) needs to be unpacked into a root-folder of the file system (which can be any operating system in which Apache/PHP and ISIS can run). After unpacking there will be a dedicated folder for Apache, another one for PHP, a cgi-folder (to contain the web-accessible executables) and a 'documents' folder (in Apache called ' htdocs') which acts as the homepage of the ABCD-application.

   • Apache comes with a pre-defined configuration file (httpd.conf in the conf subfolder of the Apache folder for Windows and in the /etc/apache2/conf directory in Linux) which defines the following specific parameters :

Apache parameter	explanation
ServerRoot "/ABCD/apache2.4"	the directory from where Apache runs
LoadModule php5_module "/ABCD/php/php5apache2_2.dll"	this instruction activates the capability of Apache to run PHP-scripts
AddType application/x-httpd-php .php	this instruction identifies the PHP-scripts as an application type
Listen 9090	the port used by ABCD, the default http-port being 80, but in order to avoid interference with other existing http-applications, if so desired, a different port can be used, e.g. 9090. In case of using a different port-number, some adjustments will have to be made in the ABCD_start.bat script and

Apache parameter	explanation
	in some OPAC-URL's. The installation wizard automates these changes.
PHPIniDir "/ABCD/php"	The folder from where PHP is running
DocumentRoot "/ABCD/www/htdocs"	The root-folder for all the files which are part of the application itself, so the ' homepage' (which is in fact : index.php in this folder).
ScriptAlias /cgi-bin/ "/ABCD/www/cgi-bin/"	The folder in which Apache will allow executables to run from instructions in the web-pages. As from version 2.0 subfolders of this cgi-bin folder can contain several other subfolders for specific versions of CISIS to be used, e.g. the Unicode or BigISIS versions.
NameVirtualHost localhost:9090	This instruction introduces a 'virtual host' with a name (in this case : localhost) running to its own port (in this case : port 9090).
<VirtualHost localhost:9090>.... </VirtualHost>	The actual definition of the variables for this *virtual host*. A typical implementation defines the DocumentRoot and ScriptAlias, each time followed by some permissions definitions for the folder defined, e.g. : **DocumentRoot "/ABCD/www/htdocs**" **ServerName localhost** **DirectoryIndex index.htm index.php homepage.htm** **<Directory "/ABCD/www/htdocs">** **Options Indexes FollowSymLinks AllowOverride None** **Order allow,deny** **</Directory>** **ScriptAlias /cgi-bin/ "/ABCD/www/cgi-bin/"** **<Directory "/ABCD/www/cgi-bin">** **AllowOverride None** **Options None** **Order allow,deny** **Allow from all** **</Directory>** Note that one can also create a 'virtual host' file in the 'extra'-folder of the apache-conf folder, with the same contents and 'included' into the main httpd.conf file with the instruction : **Include conf/extra/httpd-vhosts-abcd.conf**

- PHP comes with a predefined configuration in php.ini.

  The complete list of required PHP-modules can be derived from this command for the Debian Linux-environment :

  apt-get install apache2 libapache2-mod-php5 libxml2-dev libapache2-mod-proxy-html libpng12-dev libjpeg62-dev zlib1g-dev libtidy-dev libxslt1-dev curl php5-dev php-pear libyaz-dev php5-gd php5-xmlrpc php5-xsl

  PHP settings and php.ini

  Since ABCD uses PHP throughout with some additional PHP modules (YAZ, XSLTProcessor, GD...) Pear should be installed within the PHP- installation and some extra modules need to be copied into the PHP 'extensions' folder : php_yaz.dll, yaz.dll, yaz3.dll (these two serve the Z39.50 function of ABCD cataloging), iconv.dll, libxm2l.dll, libxslt.dll (for the XSLT Processor). The PHP-extensions folder needs to be present in the system's path environment variable (in Windows e.g. : go to 'My Computer (right-click) | Properties | Advanced | Environment Variables | System variables and edit the Path variable by adding, if not present : ';C:\ABCD\php\ext'). Also make sure your php.ini (in \ABCD\php) has the extensions mentioned here commented out (i.e. remove the leading ';' to activate the extension).

  extension=iconv.dll

  extension=iconv.dll

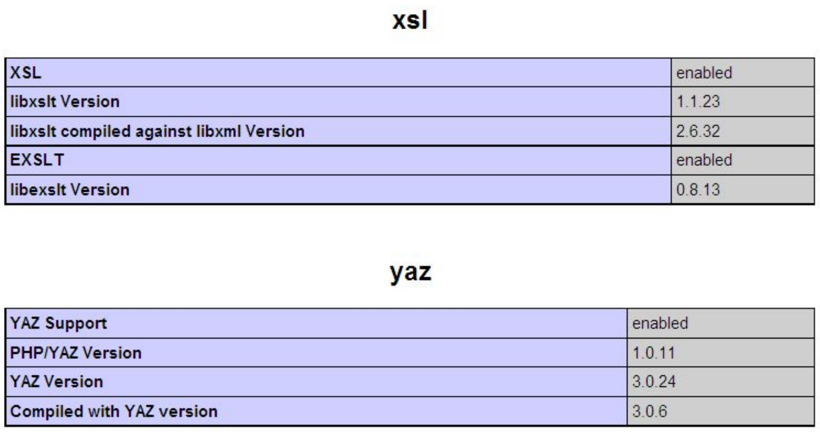
  extension=libxml2.dll

  extension=libxslt.dll

  extension=yaz3.dll

  extension=php_yaz.dll

  Be careful with possible other php.ini files existing, e.g. in \Windows or \PHP as these might disturb your ABCD-PHP. A PHP-test option is available with ABCD at the URL : http://localthost:9090/info.php. We are specifically interested in the following section below, where XSL and YAZ should be mentioned as running - if not check your path-environment variable and all paths again, as well as the 'extensions' section of your php.ini !



xsl

XSL	enabled
libxslt Version	1.1.23
libxslt compiled against libxml Version	2.6.32
EXSLT	enabled
libexslt Version	0.8.13

yaz

YAZ Support	enabled
PHP/YAZ Version	1.0.11
YAZ Version	3.0.24
Compiled with YAZ version	3.0.6

  The php.ini file contains a few more settings which need to be checked for ABCD to run correctly :

  - register_globals = On (default = Off)
  - extension_dir = "/ABCD/php/ext" (or adjust to the real path to your ABCD installation)

- default_charset = "iso-8859-1" (default = not active)
- extension_dir = "/ABCD/php/ext" => defines the extensions directory
- extension=yaz4.dll and extension=php_yaz.dll are listed in the => are added in the 'Dynamic Extensions' section in order to allow the YAZ-module for Z39.50 to work

2. an ABCD-only package, requiring Apache (or another web-server) and PHP already being installed. This is the default option for installation of ABCD in a Linux environment, because Linux expects the Apache and PHP packages to go into their own dedicated folders with the configuration into the directory /etc.

   In this case the assumption is that at least some expertise is available to understand the existing web-server installation and PHP configuration. Using 'aliases' for the ABCD-installation and cgi-folder, which can be put in a virtual host configuration file, ABCD can be installed anywhere inside or outside the existing home-folder for the web-server. So only the cgi-folder and htdocs-folder is included into this package. System managers should refer to the Apache and PHP manuals in case they are not sure about how to proceed with this type of installation.

   A dedicated installation tool, i.e. a 'wizard', has been created as part of the ABCD-software for Windows, but in essence still doing the same as described above, only after collecting some parameters for installation (like which disk to use, which port etc.) the wizard will create the correct configuration files for the different modules of ABCD as well as for Apache and PHP-configuration. For Debian Linux (including Ubuntu and Mint) there is a '.deb' (Debian) package, which can be simply installed with the command (as super-user) : 'dpkg -i abcd_2.0_amd64.deb' (in this case we use the 64-bits installation package).

   Alternatively one could also use prepackaged installations like EasyPHP or WAMP (for Windows) / XAMP (for UNIX/Linux). Again in this case Apache and PHP (and MySQL) will be automatically installed and the ABCD cgi-bin and htdocs folders have to be moved into the existing folder-structures (of Apache) and php.ini has to be edited.

3. Installation of ABCD in a pre-configured WAMP or XAMP environment.

   Combinations of Apache and PHP, mostly also with MySQL, are distributed as 'WAMP' or 'XAMP', 'EasyPHP' etc. with additional tools to facilitate configuration and maintenance (starting/stopping servers e.g.). ABCD can work as embedded into such an environment, but care has to be taken to do it properly. We discuss the principles under here, but note that since these packages also change now and then, it can never be a final 'full instruction'.

   Normally WAMP comes with its own full installer executable (or .msi file) which indeed installs the Apache webserver, MySQL (server and client) and PHP. On top of that WAMP offers – and this is a major added-value – a ‘WAMP-manager’ icon in the taskbar from where not only all the services can be (re-)started or stopped, but also the main settings for all three services can be accessed and managed. E.g. php.ini (PHP settings) and httpd.conf (Apache webserver configuration) and my.ini (settings for MySQL) are directly accessible, also version information for all three servers is available and direct access is given to phpMyAdmin.

   A standard installation of ABCD contains not only the ABCD-scripts and databases (in the www-folder) but also a folder for Apache and a folder for PHP. After configuring ABCD correctly as instructed under here with WAMP, both the PHP and Apache folder in principle can be removed from the ABCD-installation.

   a. Configuring Apache webserver with virtual hosts

      The example configuration discussed here is for a typical installation on disk C: with a MySQL version of 5.1.53 - change this for more recent versions.

      Please note that Apache settings are case-sensitive !

      The main solution we propose is to configure Apache webserver, as part of WAMP, with virtual hosts settings. This requires two steps, supposing the installed Apache version is 2.2.17 (change when appropriate to your actual version !) :

      i. Including the virtual-hosts settings file and moving the virtual host settings from the basic settings :

a. open (with any text-editor, e.g. Notepad) the file \wamp\bin\apache\Apache2.2.17\conf\httpd.conf

b. remove (or de-activate by putting the ‘#’ comment sign in front) the following statements, since we don’t want to keep them in the basic configuration for all applications :

```
Listen 80

DocumentRoot "c:/wamp/www/"

ScriptAlias /cgi-bin/ "cgi-bin/"
```

c. remove the ‘#’ before the ‘include’ statement in the following section towards the end, in order to activate the statement and make the virtual hosts configuration active :

```
# Virtual hosts

Include conf/extra/httpd-vhosts.conf
```

d. save the file httpd.conf

ii. Edit the virtual hosts settings in the file extra\httpd-vhosts.conf (with any text-editor) in order to make it contain the following two sections, one for the default host with port 80, one for the ABCD-dedicated port 9090 :

a. Create the two virtual hosts with their resp. ‘listening port’ :

```
NameVirtualHost *:80

Listen 80

NameVirtualHost *:9090

Listen 9090
```

b. Define the settings for default virtual host by adding the following lines :

```
<VirtualHost *:80>

ServerRoot "c:/wamp/bin/apache/apache2.2.17"

DocumentRoot "/wamp/www/"

<Directory "c:/wamp/www/">

Options Indexes FollowSymLinks MultiViews

AllowOverride all

Order Deny,Allow

Deny from all

Allow from 127.0.0.1

</Directory>

ScriptAlias /cgi-bin/ "cgi-bin/"

<Directory "cgi-bin">
```

```
AllowOverride None
Options None
Order allow,deny
Allow from all
</Directory>
PHPIniDir "c:/wamp/bin/php/php5.3.4"
</VirtualHost>
```

c. Create a section for the ABCD-port 9090 virtual host, by adding the following lines :

```
<VirtualHost *:9090>
ServerRoot "c:/abcd/apache2.4"
DocumentRoot "/ABCD/www/htdocs"
<Directory "c:/ABCD/www/htdocs/">
Options Indexes FollowSymLinks MultiViews
AllowOverride all
Order Deny,Allow
Deny from all
Allow from 127.0.0.1
</Directory>
ScriptAlias /cgi-bin "/ABCD/www/cgi-bin/"
<Directory "/ABCD/www/cgi-bin/">
AllowOverride None
Options None
Order allow,deny
Allow from all
</Directory>
#PHPIniDir "/ABCD/php"
# next 2 lines are only for ABCD EmpWeb
ProxyPass /empweb/ http://localhost:8080/empweb/
ProxyPassReverse / http://localhost:8080/
</VirtualHost>
```

d. Save the httpd-vhost.conf file and re-start your Apache service (the easiest way is to use the WAMP-manager icon and go to the Apache ‘restart service’ option).

iii. Configuring PHP

a. It is – most unfortunately – not possible to refer to 2 different PHP-settings by including the directive ‘PHPIniDir’ twice (or more) in the virtual hosts settings. Apache will only observe the first one and report the second one as an error. So one can either leave that directive into the basic configuration file httdp.conf or put it into one (and only one) of the virtual hosts sections of the httpd-vhosts.conf file :

PHPIniDir "c:/wamp/bin/php/php5.3.4"

One could put such statement e.g. before closing the virtual-host section (</VirtualHost>).

b. Add the php-extensions which ABCD needs but are possibly not yet active in WAMP, by removing the leading ‘;’ or adding if not yet present in the section of the file ‘php.ini’ (in WAMP with PHP5.3.4 you will find this in C:\wamp\bin\php\php5.3.4) labeled as ‘Dynamic Extensions’ :

```
;;;;;;;;;;;;;;;;;;;; ;

Dynamic Extensions ;

;;;;;;;;;;;;;;;;;;;;;

extension=php_gd2.dll

extension=php_pdo.dll

extension=php_pdo_mysql.dll

extension=php_xsl.dll

extension=php_yaz.dll
```

c. Other php.ini settings could be slightly different in between WAMP and ABCD, but after some testing we didn’t find them to be really important for either ABCD or WAMP, but rather ‘fine-tuning’ settings.

d. Copy the here added extension DLL’s from your ABCD-php extension directory (folder) to the one of WAMP, probably being resp. \ABCD\php\ext and \WAMP\bin\php\php5.3.4\ext.

Beware : don’t copy existing ABCD-extensions over existing WAMP-extensions, only add non-existing ones !

Warning : if the PHP version is indeed different, you will need to download the correct versions for the WAMP-PHP configuration. If e.g. your ABCD PHP is using version 5.2 but WAMP is using 5.3, you will have to load the correct versions for version 5.3, as that is the one we will be using.

Since WAMP installs its own MySQL server (in the folder \WAMP\bin\mysql) with its data-folder inthere as a subfolder, you might – but only if you use MySQL for the ABCD EmpWeb Advanced Loans module – have to reconfigure your MySQL configuration, i.e. the following directives might need to be edited in order to refer to the MySQL-folder used when installing MySQL for EmpWeb :

```
basedir=c:/wamp/bin/mysql/mysql5.1.53

log-error=c:/wamp/logs/mysql.log

datadir=c:/wamp/bin/mysql/mysql5.1.53/data
```

This file ‘my.ini’ also allows to set the port (default =3309) and password/login to access your MySQL.

iv. Restart all services to test the new set-up. If you have done all previous steps correctly, it should be possible to open both default port web-applications, e.g.

http://localhost/phpMyAdmin

and ABCD-port9090 applications :

http://localhost:9090/

4. Making ISIS/ABCD files types known to your Operating System

   One issue which could make life easier when being an ABCD-administrator, is to teach the Operating System (OS) how to deal with ISIS-files - mostly they are simple text-files which can be opened directly in a file-manager with a text-editor such as Notepad in Windows or Gedit in a Gnome-based Linux. Whereas ABCD already provides lots of functions to deal with such configuration files, lists and menu's from within its own interface (the WWW-browser), if one has access to the server itself quite often it could be handy to edit quickly some files directly from the OS, e.g. from a terminal or CMD-console. In the case of crucial files such as 'config.php' for the Central module, this approach is even mandatory.

   Modern OS's try to 'recognize' files and link them immediately to the most likely software for opening them by checking the extension of the files. Unfortunately typical ISIS-related extensions are not known to either Windows (which e.g. 'knows' .DOC or .DOCX files to open them with Word or .txt files to open with Notepad) or Linux. Therefore one has to, or can teach the OS to know some frequently used ISIS-text files for quick checking and/or reading.

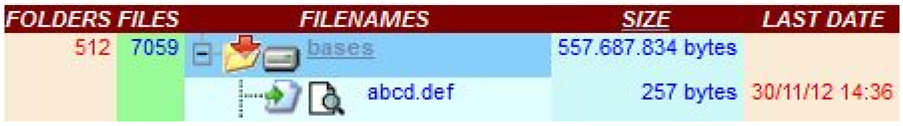
   Such linking is done by right-clicking on the file in a graphical file-manager (like simply 'My Computer' in Windows) and select the 'open with' option from the menu. Mostly some good candidates like Notepad will already be listed but also if not yet present they can be identified by browsing the file-system to identify the correct executable. Once selected make sure the option 'always use this' is active, which actually means that your OS will link the extension of the file to the chosen executable.

   It is safe to link the following types of files to Notepad in Windows or nano/Gedit in Linux :

   - .php in the htdocs folders with ABCD-scripts (the PHP-scripts, but know what you are doing !)
   - .FDT, .PFT, .FST, .STW in the database or database-definitions and formats folders
   - .TAB, .DAT, .DEF files in several ABCD-related folders.

   Note

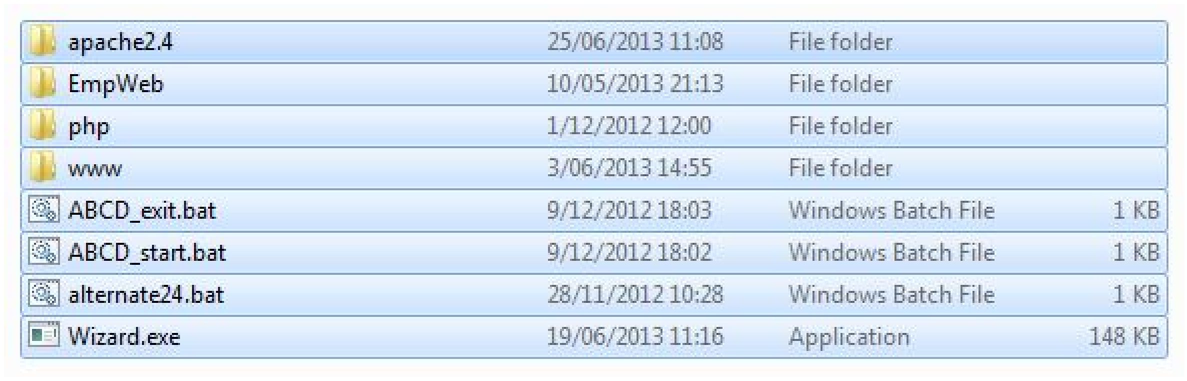
   There is one parameter in config.sys which, if active, allows administrators to 'browse' the file-system of the server at the level of the 'bases'-folder from within ABCD and viewing but also editing the text- and configuration files : the variable $dirtree. With this variable set to 'on' (1), quite impopular with network-administrators by the way (!), in the main menu there will be an extra option for such file-browsing with viewing/editing icons :



FOLDERS	FILES	FILENAMES	SIZE	LAST DATE
512	7059	bases	557.687.834 bytes	
		abcd.def	257 bytes	30/11/12 14:36

3.3. Directory structure : folders and files

After installation of ABCD the following folder structure will be created (in this case EmpWeb is included) :



apache2.4	25/06/2013 11:08	File folder	
EmpWeb	10/05/2013 21:13	File folder	
php	1/12/2012 12:00	File folder	
www	3/06/2013 14:55	File folder	
ABCD_exit.bat	9/12/2012 18:03	Windows Batch File	1 KB
ABCD_start.bat	9/12/2012 18:02	Windows Batch File	1 KB
alternate24.bat	28/11/2012 10:28	Windows Batch File	1 KB
Wizard.exe	19/06/2013 11:16	Application	148 KB

As can be seen, 3 (or 4 if EmpWeb is included) sub-folders have been created in the main folder /ABCD. The standard folders are resp. :

1. apache2.4

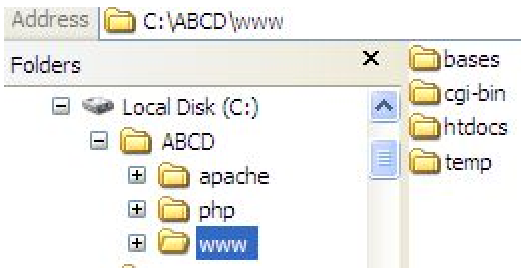
   The Apache folder contains the currently latest version of the Apache web-server software, which is in fact only one of many important softwares developed by the Apache Software Foundation. By default the Apache webserver is installed in another base-folder (e.g. in Windows : C:\Program Files\Apache Software Foundation\Apache2.2) and network-managers will probably have installed Apache on their server(s) according to their own preferences, but when installed from the 'full ABCD-package' Apache will run - with its configuration file httpd.conf adjusted for this situation - from \ABCD\Apache*xx*.

2. php

   THe PHP folder contains the current version of the PHP scripting software. Again, as with Apache, in many instances this software will be installed in its own right, e.g. in C:\PHP, or often also as part of a combined package containing Apache, MySQL and PHP, e.g. with EasyPHP or WAMP-server. When installed as part of ABCD however PHP will run from here with the necessary adjustments done in the main PHP-configuration file php.ini.

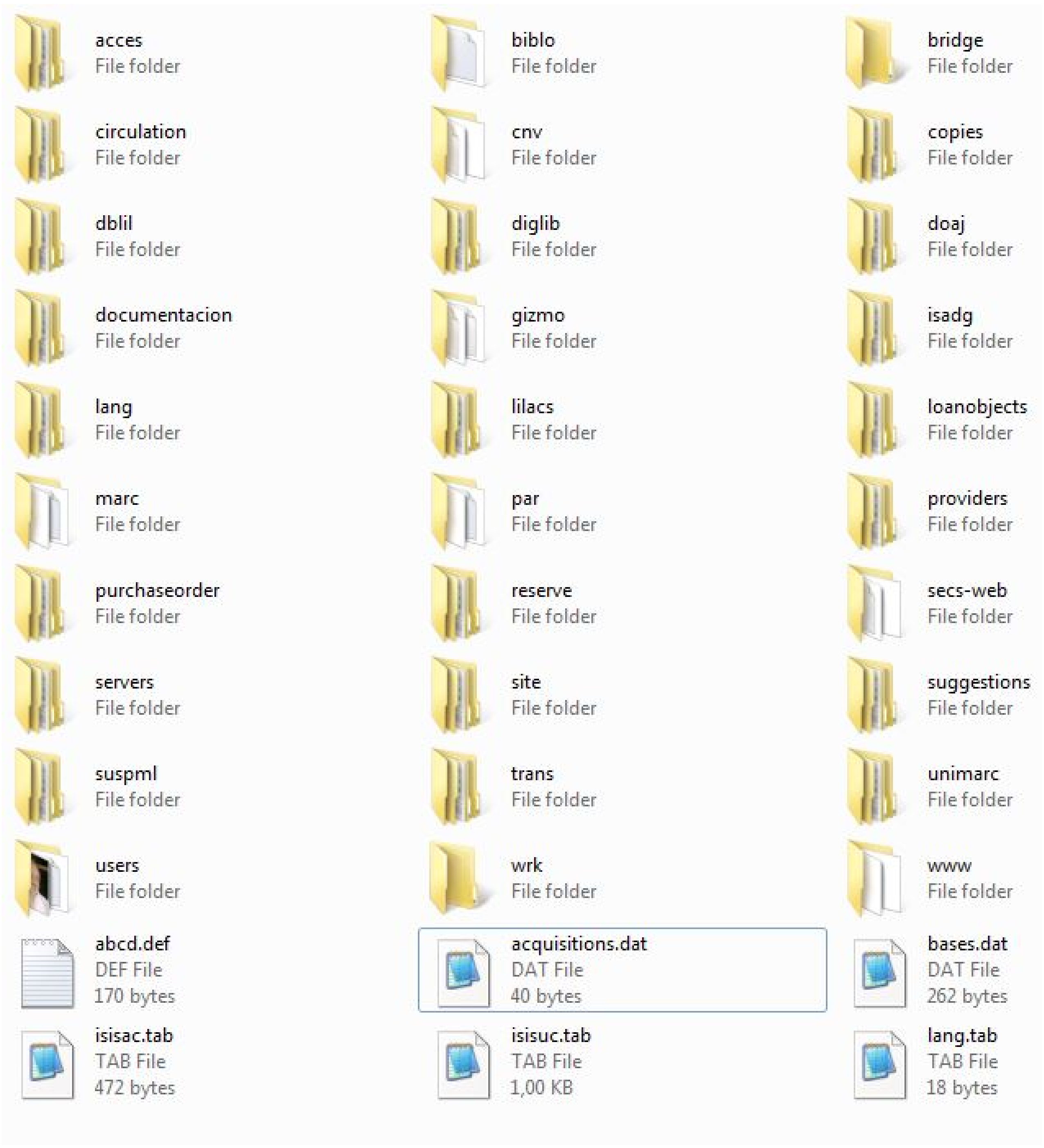
3. www

   The www folder contains the whole ABCD system, which is subdivided in 4 folders :



   a. **bases**

      The bases folder contains the databases of your ABCD installation, which one dedicated subfolder (with many subfolders in its turn) for each database. When an additional database is copied or created using ABCD, the system will create such a dedicated extra subfolder here. A typical list of database-folders in the /bases folder looks as follows :



acces File folder	biblo File folder	bridge File folder
circulation File folder	cnv File folder	copies File folder
dblil File folder	diglib File folder	doaj File folder
documentacion File folder	gizmo File folder	isadg File folder
lang File folder	lilacs File folder	loanobjects File folder
marc File folder	par File folder	providers File folder
purchaseorder File folder	reserve File folder	secs-web File folder
servers File folder	site File folder	suggestions File folder
suspml File folder	trans File folder	unimarc File folder
users File folder	wrk File folder	www File folder
abcd.def DEF File 170 bytes	acquisitions.dat DAT File 40 bytes	bases.dat DAT File 262 bytes
isisac.tab TAB File 472 bytes	isisuc.tab TAB File 1,00 KB	lang.tab TAB File 18 bytes

As can be seen, many databases exist (but not as many as there are tables in a relational setup, since ISIS does not practice 'normalisation' into related tables), some of them - e.g. marc, biblo, dblil - are models coming with the installation of ABCD, others - in this case e.g. 'doaj' and 'isadg' - are created by ABCD in the author's installation only, while finally others are serving specific modules of the library system, e.g. 'providers' and 'purchaseorder', are used for the acquisitions module, 'suggestions', 'reserve', 'suspml', 'trans' and 'users' are used for the Loans Module. So each ABCD-bases folder will be different according to the actual databases used.

A special database is the database 'acces' which holds the users (with their login data) and their access-rights (authority level) to the databases.

Another special database-folder is the folder 'par' is not a database folder but it holds the .par files for each database known to ABCD. A .par file actually is a small text-file (so it can be edited by any TXT-editor like

Notepad) with on each line the full path reference to parts of the database concerned. E.g. a typical .par file for ABCD, but with also added references to 'copies' and 'loanobjects' and 'trans' (in order to be capable to retrieve information from these databases from PFT's for MARC) looks like this :

```
marc.*=%path_database%marc/data/marc.*              prologoact.pft=%path_database%www/
prologoact.pft       prologo.pft=%path_database%www/prologo.pft       epilogoact.pft=
%path_database%www/epilogoact.pft          epilogo.pft=%path_database%www/epilogo.pft
autoridades.pft=%path_database%marc/pfts/pt/autoridades.pft  isisac.tab=%path_database
%marc/data/isisac.tab       isisuc.tab=%path_database%marc/data/isisuc.tab       STW=
%path_database%marc/data/marc.stw        copies.*=%path_database%copies/data/copies.*
copies.pft=%path_database%copies/pfts/pt/copies.pft   inven.pft=%path_database%copies/
pfts/pt/inven.pft loanobject.*=%path_database%loanobjects/data/loanobjects.* loan.pft=
%path_database%loanobjects/pfts/pt/loan.pft trans.*=%path_database%trans/data/trans.*
```

Each element gets, after the equation sign, its path in the file-system. As can be seen, variables taken from the PHP Environment can be used, in this case %path_database%, which is substituted by the real pathname as defined in the main configuration file config.php (see infra).

While normally all elements, referred to here, belong to the database in question, elements of other databases should also be added if they are used in 'REF'-statements of the formats used in this database, since ISIS will have to know where to locate such external database element if called from a format - and will look for its path here ! So in this example marc.par copies, loanobjects and trans references were added.

[NEW] As from v2.0 ABCD will also accept, as did Micro CDS/ISIS for DOS and WinISIS before, the file '**syspar.par**' which contains references for all databases, so esp. when some databases are referenced to from different other databases, e.g. 'copies' and 'loanobjects', they can be put here.

An example syspar.par looks like :

```
maxmfnrl=800000     marc.*=%path_database%marc/data/marc.*loanobjects.*=%path_database
%loanobjects/data/loanobjects.*       prologoact.pft=%path_database%www/prologoact.pft
epilogoact.pft=%path_database%www/epilogoact.pft
```

In this example the first line is a special one : .par files can also contain instructions for the CISIS-database technology, in this case defining the maximum length of records to 800Kb (which only makes sense when using one of the 'FFI'-type versions with large records up to 1Mb).

[new] Another innovation in version 2.0 is the use of the file 'dr_path.def' into the proper database folder of the bases-folder. For the database concerned it gives the 'root'-folder to denote where the 'documents repository' (dr) for that database will be stored. E.g.

```
ROOT=c:/abcd/www/bases/biblo/dr/
```

indicates that for the biblo-database whenever files will be uploaded to be linked to from records in this database, the browser will start (as 'root') from this folder, where the pictures, PDF's etc. will be stored, or in selected subfolders hereof. The browser function for this also allows the creation of sub-folders (but only at a 'lower' level) to contain specific files in order to allow a good organisation of the repository.

b. **cgi-bin**

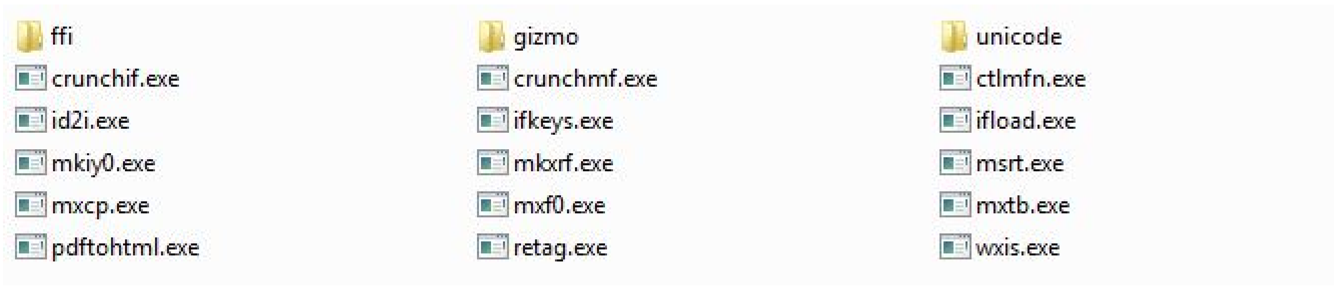
The cgi-bin folder contains the executables which ABCD will call from its web-pages and which therefore should be authorized to run by the webserver (Apache) using the CGI-protocol. This is what the 'ScriptAlias' instruction in the Apache configuration actually is for. In the case of ABCD the main executable is the wxis.exe ISIS-server, which does the main part of the job. Some other CISIS-tools are however also included for specific tasks.

The wxis-modules subfolder here contains scripts (with .xis extension) for the wxis-server, while the 'gizmo' folder contains some small ISIS-databases which define strings to be substituted by another one, e.g. for changes due to different environments used (DOS/ASCII, Windows/ANSI, WWW/XML.

As from version 2.0 ABCD allows the use of dedicated special versions of CISIS to be used for any specific database. For this to work the name of the database has to be added in a separate line of the file 'abcd.def' (in ABCD/www/bases), followed by '=' and the name of the special version, e.g. for the Digital Library database 'diglib' one could define 'ffi' to be the special version (with large records) to be used as follows :

```
diglib=ffi
```

A typical cgi-bin folder then looks like this :

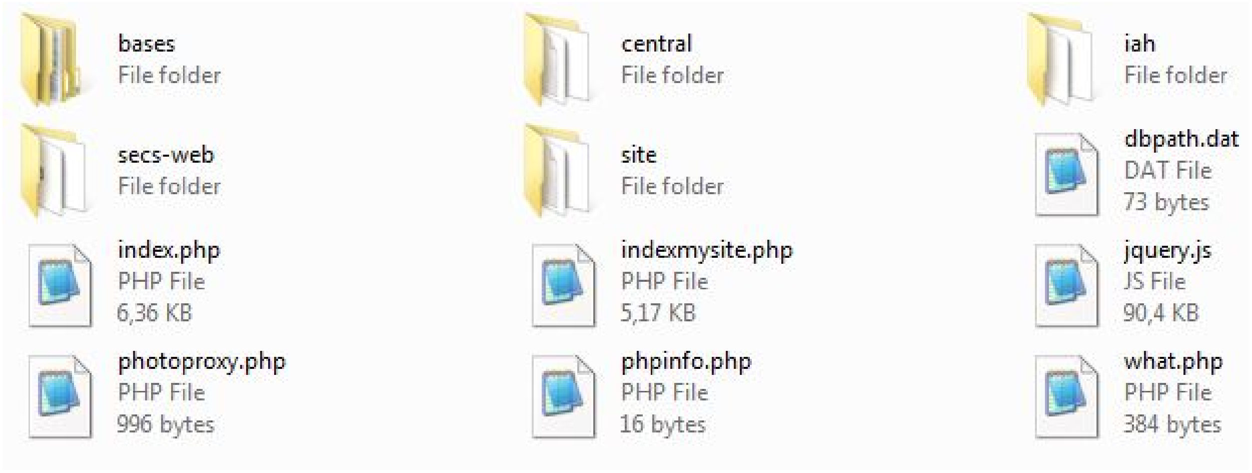


ffi	gizmo	unicode
crunchif.exe	crunchmf.exe	ctlmfn.exe
id2i.exe	ifkeys.exe	ifload.exe
mkiy0.exe	mkxrf.exe	msrt.exe
mxcp.exe	mxf0.exe	mxtb.exe
pdftohtml.exe	retag.exe	wxis.exe

As can be seen in this case a folder 'ffi' and a folder 'unicode' have been added to allow databases with large records and with Unicode charactersets (see infra) to be used. One has to obtain the available CISIS-versions by downloading them from the BIREME-website (http://http://reddes.bvsaude.org/projects/cisis/browser/trunk/utl/)

c. **htdocs**

The htdocs (we use the traditional Apache 'hypertext documents' folder name) is the 'home-folder' of the web-site served by the ABCD-Apache server. So therefore it contains all the software elements (except the basic external technology such as Apache and PHP and the Java-based EmpWeb module for advanced loans) specifically produced for ABCD :



bases File folder	central File folder	iah File folder
secs-web File folder	site File folder	dbpath.dat DAT File 73 bytes
index.php PHP File 6,36 KB	indexmysite.php PHP File 5,17 KB	jquery.js JS File 90,4 KB
photoproxy.php PHP File 996 bytes	phpinfo.php PHP File 16 bytes	what.php PHP File 384 bytes

As can be seen, the main parts of the ABCD-suite can be identified here : Central, iAH, Secs-Web and Site. Since ABCD is a 'suite' of different functions, each one has its own homepage, i.e. the 'index.html' file located in the appropriate subfolder.

The main starting script is present within this homepage folder : index.php (which is the default home-page indeed, allowing the URL of ABCD only to refer to the server-part) and the [new] 'what.php' script for including the info contained into the file 'abcd.def', such as the institution's name, version info and URL-references for the footer-part of the ABCD-screens. Some more scripts are located at this level : homepage.php and inicio.php are the starting pages, which read into memory the main configuration parameters defined in config.php (or config.loans.php for the Loans module). [!!] If EmpWeb with its 'mySite' functionality is installed, additional initial scripts will be found here too : iniciomysite.php, homepagemysite.php, photoproxy.php and empwebavailibility.php (see the documentation on EmpWeb).

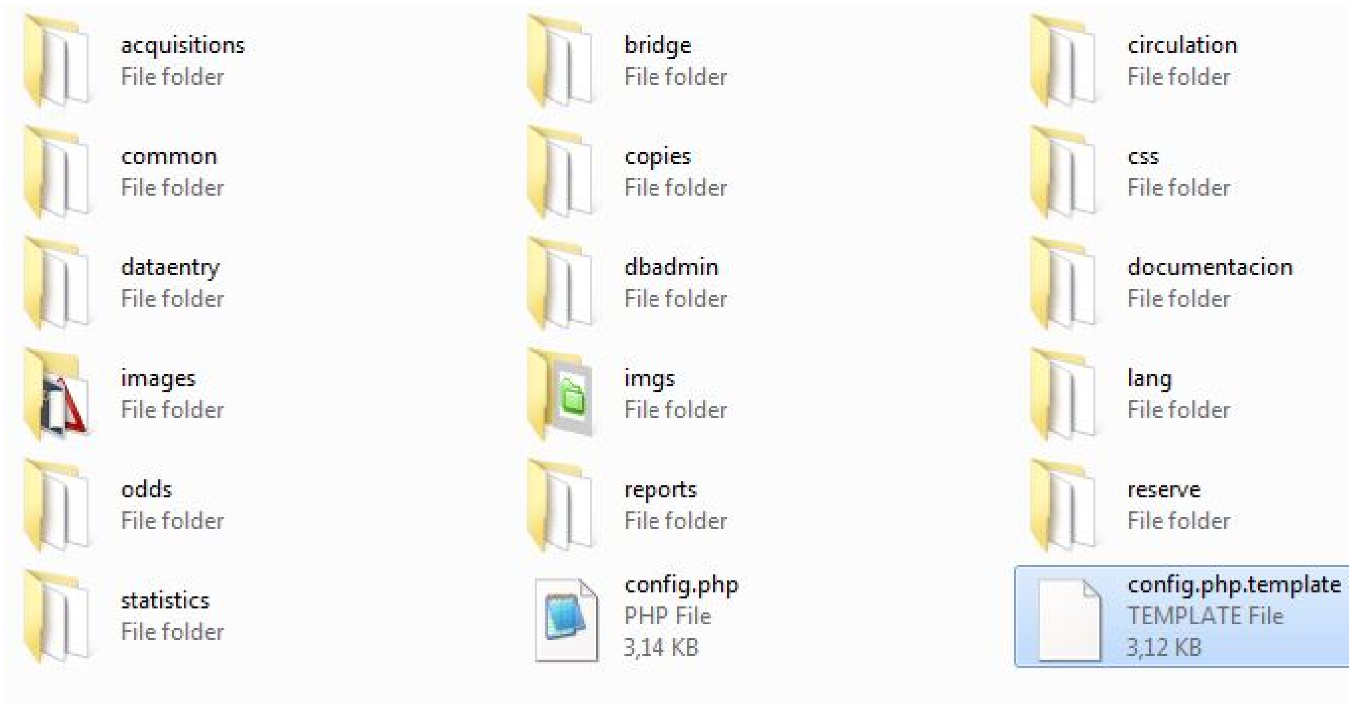
The main folders of the ABCD-system in the htdocs-folder are briefly described below here :

i. bases

   Here for each database (in a dedicated subfolder) external files linked to from the records in the database, e.g. full-text PDF's or images, can be stored. Since this folder is 'under' the htdocs-folder, i.e. the DocumentRoot of the web-server, ABCD can access it easily, but (as from version 2.0) the 'dr_path.def' file also allows defining other folders, also outside the DocumentRoot for this purpose. E.g. the user images can be stored here in a subfolder 'users', so the photos of the user will be shown whenever a loans-system user is presented.

ii. central

   This is indeed, as suggested by the name, the 'central' part of the system where most of the database administration and many core-activity scripts of the software are included. We will therefore deal with the important subfolders contained in here :



The basic configuration file for Central : CONFIG.PHP

The parameters defined in this crucial file 'config.php' are, without fixed order, amongst others (see also the discussion on this in the first section of the ABCD Central modules discussion in Chapter 2) :

- $open_new_window : if 'on' (Y) a new main ABCD browser window will be opened
- $context_menu : if 'on' (Y) right-clicking will issue a menu with e.g. a 'back'-button (beware : going 'back' artificially without following the ABCD software's buttons could result in trouble !)
- $config_date_format : the format for date-values, e.g. "DD/MM/YY"
- $app_path : the name of the folder in which the Central module is installed, by default this is 'central'
- $inventory_numeric : if 'Y'(es), leading zero's (at the left side) of the barcode (or identifier) of copies will be automatically removed to make it a numerical value
- $max_inventory_length : the fixed number of positions for the value of the barcode; missing positions will be filled with zero's, e.g. with value 6 the barcode 456 will be changed into 000456

- $max_cn_length : same as max_inventory_length but for the Control Number of the catalog records
- [new] $log="Y"; When put as 'Y'es, and given a subfolder 'log' exists in the www/bases-folder (as it has to be 'writable' by the software), ABCD will log all operations into a text-file for later analysis. For each single day a separate text-file (named as 'log_yyyymmdd.log' will be created containing the date/time and operator info followed by all URL's sent by ABCD Central as instructions to the engine, e.g.

  ```
  **Friday 1st of March 2013 12:04:31 PM Operador: abcd /central/dataentry/inicio_main.php /ABCD/www/htdocs/central/dataentry/wxis/login.xis IsisScript=/ABCD/www/htdocs/central/dataentry/wxis/login.xis&base=acces&cipar=\abcd\www\bases\par/
  acces.par&login=abcd&password=adm&path_db=\abcd\www\bases\&cttype=s
  ```

- $db_path : the path to the folder where the databases are stored
- $msg_path : the path to the folder where the message database (lang) and the online help-pages are stored; by default ABCD takes for this path the $db_path, so that for all databases the same msg_path can be used
- $img_path : default folder for all files containing images or texts (e.g. PDF) referenced to in the records for this database; this default can be changed in the file 'dr_path.def' in each database-folder
- [new] ABCD takes the dedicated CISIS-version, $cisis_ver, if not the default one, from the 'abcd.def' file (in the main databases folder) with the following code in this config.php :

  ```
  $cisis_ver="";

  if (isset($arrHttp["base"])){ if (isset($def[$arrHttp["base"]])) $cisis_ver=
  $def[$arrHttp["base"]]."/"; }
  ```

- $Wxis : the path to the executable, i.e. the ISIS-server, called directly with the GET-method.
- $wxis_URL : the URL for the ISIS-server, which can be left empty in order to not use the CGI-calls but rather the direct executable calls from PHP. This method uses the 'POST' protocol of HTTP allowing much longer URL's and is therefore preferred. If present, this method will be given priority over the use of the direct Wxis-call as defined in the variable $Wxis. So leave it empty if you don't want the POST-method to be used with the $Wxis-variable. E.g. :

  $wxisUrl="http://localhost:9090/cgi-bin/$cisis_ver"."wxis.exe";

  Also note the use of the right Apache-port and the *$cisis_ver* variable into the URL but also the path of wxis.exe.

- $xWxis : path to the wxis scripts
- $lang : default language
- $lang_db : default language for the database administration module
- $institution_name : the name of the responsible institution is no longer (as in v1) defined here but is now taken from the file 'abcd.def' in the main databases-directory
- $change_password="Y"; if this parameter is present and set to 'Y' (es) the login screen will provide an option to change the password
- $FCKEditorPath : the path to the HTML editor built-in into ABCD
- $FCKConfigurationsPath : the path to the HTML editor configuration

- MD5 : use of encryption for passwords (1) or not (0); WARNING : if changing this value all existing passwords will no longer be valid and have to be re-assigned !
- $dirtree=1: show (1) or hide (0) the icon that gives access to the exploration of the bases-folder; some network- or server-managers will not allow such function on their system !
- (new in v1.2t) $ext_allowed=array("jpg","gif","png","pdf","doc","docx","xls","xlsx","odt"); : this lists the file-types allowed for uploading; the list given here is the 'default' list as defined in the script htdocs/central/dataentry/upload_image.php, but other extensions can be added or exisiting ones can be disallowed by removing them.

The remaining folders here deal with one specific function or module of ABCD by storing the PHP-scripts with lots of additional elements (images and style-sheets for the webpages etc.) : acquisitions, dataentry, dbadmin, loans, odds, statistics and usersadm.

The names of the folders are sufficiently self-explanatory in these cases except for the [new] new ODDS-service : online document delivery service.

Here we would only like to underline the presence of a module 'database administration' which allows creation of any ISIS-structure to deal with any type of textual data, allowing ABCD to be more flexible than most other systems and more than just a library system.

Special folders, not dedicated to specific functions in ABCD-Central, are the following :

- common : in here there are some crucial php-scripts which are needed by all modules, e.g. 'header' and 'footer', but also 'wxis-llamar.php' (which allows using either the cgi-method of calling executables (safer) or direct executable calls from PHP (faster). The institutional_info.php script defines the name of the resonsible institution of the ABCD-installation, which will be called upon in many pages.
- documentacion : obviously this folder contains scripts to deal with the online-help functions of ABCD.
- images : contains small images used in many pages (mostly .png and .gif)
- css : contains the Cascading Style Sheets used in this central part of ABCD
- lang : contains for each module a script to facilitate language switching or reverting to the default '00'language

iii. iah

iAH is the original name of the **interface for a**dvanced web- for end-user searching (**'h**arvesting') of BIREME which acts as the OPAC of ABCD.

iv. secs-web

This module allows ABCD to offer advanced serials management tools within the web-environment : **Se**rials **C**ontrol **S**ystem.

v. site

Finally the 'Site' module combines advanced OPAC searching (with meta-search possibilities) with a 'portal' service, offering the search option within an environment of other networked information resources and communication with users. The structure and the contents of this portal can be edited online with a built-in ABCD Content Management System.

d. **EmpWeb** (only if installation of EmpWeb was added !)

This folder contains most but not all files necessary to run EmpWeb, i.e. the Java Jetty server and the Java scripts. EmpWeb however additionally needs also added scripts in ABCD Central (this allows the Advanced Loans to be compatible with the built-in Loans system of ABCD) and - since it uses an SQL-database for storing the transactions - an installation of one of the common SQL-databases (MySQL, PostGres, Oracle...), which needs to be done separately - use the installation instructions for the SQL-solution chosen. A separate part of the ABCD Manual deals with EmpWeb as it is quite different from the rest of the system.

3.4. Access rights

In both modern Windows versions as in all Linux-based OS's folders and files have 'properties' which allow control of who can do what with these files. In Linux this security policy has been maintained from the beginning in a more strict way than in Windows : folders and files belong to a specific user ('owner') and for each of three categories (the user, his group or 'others') it can be defined whether the file (or folder) is 'readable' (r), 'writable' (w) and 'executable' (x). These properties can be set at the Linux command-line with the commands 'chown' (change the owner) and 'chmod' (change the modus). In Windows one can 'right-click' on any folder/file and find these properties in the Properties|Security tab with options to change them.

To keep things simple - but network-managers might prefer to go into more refined access-right policies - we suggest the following settings :

- the database-folder (in Windows : \ABCD\www\bases, in Linux : /var/opt/ABCD/bases) gets the setting '777', meaning everybody can read and write the databases;
- all the other folders, i.e. \ABCD\www\cgi-bin|htdocs|temp, should be readable by everybody but writable only by the owner or his group : '775'.

In Windows users might only be confronted with the problem that as from Vista on, Windows will apply some more rigid access rights control, meaning that files outside the 'home'-folder for users, i.e. not in the C\users *username* (sub-) folders don't have full access rights. To change these (as we prefer ABCD to be installed in a root-folder instead) one should use the following options :

- right-click on the folder in which your ABCD-installation is put
- select 'properties' (mostly the last option in the pop-up menu)
- select the 'security' tab
- click on 'edit'
- select the 'users' category
- - tick the box for 'full' access for this category in the lower part of the window.

If ABCD was installed in a subfolder of the 'users' folder, no need for this.

This is equal to setting access-rights to '777' in Linux, which we recommend for the bases-folder.

3.5. Migrating from older versions

When installing ABCD (version 1.2t or 2.0) on a computer/server where there is already a running ABCD system, special precautions have to be taken. This chapter will give the main instructions to be considered, but remember that ABCD can be installed in an almost unlimited number of configurations : WIndows/Linux, with or without pre-installed Apache and PHP, on any hard-drive (C:, D: ... or in Linux : in /var, /opt etc). So always the system manager has to check carefully for any local variations.

3.5.1. Installation in Windows with existing ABCD-folder

By default ABCD in Windows was installed in a root-folder named 'ABCD'. If this is not the case, please change all references to \ABCD to the path indicating the correct path for your installation.

In principle, ABCD v1.2t and 2.0 are compatible in the sense that databases from earlier versions can fully be kept in the newer versions : the Master and XRF files but also the Inverted File files are exactly the same. The newer versions only offer extra features which are then not used in the older versions. So, as far as the databases are concerned - and this involves all files in the 'data' subfolder of each ABCD-database (e.g. \ABCD\www\bases \marc\data for the MARC21 database) - it suffices to keep a copy of these files somewhere else and re-copy them into the same folder after installing the new version. If the existing database was locally created with another name (and doesn't appear in the list of database-folders in \ABCD\www\bases), this is even not necessary since the database files won't be deleted or overwritten.

Some more files to backup for later merging into existing ABCD-files are :

- ABCD\www\bases\abcd.def (if it already exists, depending on your installed version) with institution name, special CISIS-versions for specific databases, path to the mx-executable, header- and footer info (like version, links)
- ABCD\www\bases\lang.tab : the list of languages used in your installation
- ABCD\www\bases\bases.dat : the list of databases used in your installation

Still in principle the module iAH (OPAC) will only be affected if the installation path has changed. Check the paths in the .def files for each of your OPAC-databases, which can be done from within ABCD Central using the 'Configure database in IAH' option of the 'Update database definitions' of the main Central menu. The Secs-Web module only uses relative paths in its scripts from where it is started and therefore is not affected.

The ABCD Site however uses a fully new version of the BVS-Site technology and therefore needs more interventions to preserve compatibility. This issue is discussed in a separate section under here.

3.5.1.1. migration of ABCD Central

In addition to the principles described above, for migrating the ABCD Central module to the new version 1.2t or 2.0, please consider the following.

3.5.1.1.1. bases.dat

The main measure to be taken, as far as the databases are concerned, is to keep a copy of your file 'bases.dat' (in \ABCD\www\bases) and merge it into the new one of the new installation, by adding/deleting lines in the file directly with a text-editor. There is a way to do the same from within ABCD : the option 'list of available databases (bases.dat)' in the 'update database definitions' option of the main Central menu, for those who don't have direct access to the physical file bases.dat on the server.

3.5.1.1.1.1. changes in FDT's and worksheets

In ABCD, differently from previous ISIS-implementations, the worksheets are integrated into the Field Definition Table, however keeping the possibility to create and use several worksheets for specific purposes, e.g. specific document types. As such the FDT can be considered to be a 'repository' of fields to be used (or not, due to the flexibility philosophy of ISIS) with their corresponding HTML-representation in web-pages.E.g. a repeatable subfielded field (a 'group') can be represented in the worksheet as a table, where each subfield is a column and each occurrence is a row in the table. As we will see later on, many additional possibilities have been created in this area for data-entry, e.g. fixed length subfields, character counter etc.

The new version of ABCD has reduced the number of field-types but expanded the list of possible data-entry HTML-elements for worksheets. No manual changes are necessary to be made : when opening an old FDT the software will correctly interprete the old elements and even save as new elements when the FDT or worksheet is saved again in the new version. So users might prefer to review all of their FDT's and FMT's by opening them and saving them again, to make sure all of them have been upgraded to the new standard.

3.5.1.1.1.2. configuration files

As mentioned above, the ABCD configuration files (1 per module) contain mainly definitions of paths of the installation and some default values for language and databases. This means, from the perspective of migration from an older to a newer ABCD-installation, that possibly some changes have been made in existing configuration files, which will be lost after installing the new version. Therefore we recommend keeping copies of the following files in order to be allowed to merge any changes in the old files into the new ones :

- ABCD\www\htdocs\central\config.php
- ABCD\www\htdocs\site\bvs-site-conf.php
- ABCD\www\htdocs\iah\scripts\iah.def.php and ABCD\www\htdocs\iah\index.html
- (in case already existing) abcd.def in ABCD\www\bases
- (in case already existing) db_path.dat in ABCD\www\htdocs

3.5.1.1.2. the database folders

Finally the databases themselves, all grouped together in one folder with the name of the database concerned, should be secured as far as they have been changed by local use. Just make sure you have backups (the easiest way is to simply re-copy the database-folder somewhere else) for each non-demo database. After installation of the new version you can then copy the folder back into the bases-folder of the new installation, add the name of the database into the file 'bases.dat', if necessary change the path into the corresponding .par file (in the par -folder of the bases folder) and that should suffice to allow the use of your database in the new installation.

More sophisticated possibilities exist now by using the new feature of multiple bases-folders, e.g. you could keep the demo-databases in one folder and your operational ones into another one, and have both folders referenced to in the (new) file 'db_path.dat' in the htdocs folder.

E.g. your file 'db_path.dat' with the following two lines (for a typical Windows installation) :

\ABCD\www\bases\|Demo

D:\library\bases|Operational

would allow, by selection of the first option in the login-screen, the use of the demo-databases, while the second option would give access to the real (operational) bases of the library.

3.5.1.2. ABCD Site

As mentioned above, ABCD from v1.2t on, uses a new version of the BVS-Site technology : version 5.3.1.

The configuration for correct reference to the paths of flies of the ABCD OPAC Site is done by editing the file /www/htdocs/bvs-site-conf.php.

Then the following steps need to be performed carefully :

1. copy the files from the old folder */htdocs/site/local/* to the new ABCD installation, same folder
2. copy your parameters for images and stylesheets in the folders htdocs/css and htdocs/images from the old to the new installation
3. copy the database 'site' from the old ABCD bases-folder to the new bases folder. This will preserve existing links to e.g. OPAC's and other resources.

Finally enter the Site Administrator submodule of ABCD-Site, by entering the URL (supposing port 9090) :

http://localhost:9090/site/admin/

In this new version the default or demo user and the password are: User: admbvs Password: adm@bvs

Check all links and adjust if necessary the paths in the configuration dialog windows.

Enter each one of the components which form your site, request to edit them and proceed to save them again. This will cause the components to be generated for the new version, creating the necessary .html and .ini files replacing those which come in the installation package. Each time you save a component, verify with “Preview” to see whether the conversion has been done. If your component does not appear, press Ctrl+F5 because sometimes the cache memory can cause a mistake.

Some more advanced XML-based features (e.g. RSS-feeds) might need being redone fully because of incompatibilities of the new XML-parser used in this version.

3.5.2. Installation in Linux with existing ABCD-installation

The same principles as for Windows apply to installation in Linux, however with one difference : the default directory of ABCD-installation has changed from /var/www to /opt. This means that - unless for some reason your ABCD is already installed in the /opt directory - your old configuration will not be affected by installing the new one and in fact they can co-exist, even using both bases-directories with selection of either one in the (new) login page of ABCD Central.

Normally it suffices to copy files which are specific for your ABCD-installation (see the list of files mentioned above for Windows, as they are the same) into the new installaton folder. E.g. remember to copy the 'Site' database folder from old to new to preserve the contents of your Site, and the CSS and Images folders for preserving the style and graphical presentation.

3.5.3. Migration of existing ISIS-database(s) into ABCD

ABCD Central provides an easy way to migrate existing ISIS-databases to ABCD : it is one of the three options when creating a new database. So by selecting 'Create from WinISIS database' in the first screen when having selected the 'create database' option of the main ABCD Central module, ABCD will guide you to migration with the following steps :

- copy the FDT from the existing ISIS-database
- copy the FST from the existing ISIS-database
- copy the default PFT from the existing ISIS-database

after which ABCD will automatically all initial folders and files for the new database. Check the results for errors or warnings however to make sure the creation of the empty database has been successful.

What then remains is 1) to 'fill' the database with the records from the existing database and 2) re-index the full database. The first job is done by 'importing' ISO-records (to be created by your existing ISIS application) from the 'utilities' submenu of ABCD-Central once having entered into the database, and in the same utilities menu you will find the option 'full inverted file generation' to fully re-index your database.

However in case of larger databases, the http-protocol with its overhead in client-server communication and time-outs (as defined in the case of ABCD by the php.ini settings), we strongly recommend using command-line operation for such job, which results in a much faster execution of the steps but requiring direct access (e.g. through ssh) to the server machine and its database-folders. These commands are discussed later on in this manual, but we are also working on a new 'utilities' page in which the commands can be created interactively from the ABCD-webpages while still being executed directly in the OS command-environment to preserve the speed.

Chapter 2. ABCD Modules

1. Introduction and general configuration

This chapter deals with the main functions of the 'Central' module of the ABCD system. As an integrated 'library automation software' the ABCD system offers tools for database management (both for bibliographic/document databases and administrative databases such as users, acquisitions and loans), data-entry, statistics, circulation, serials control and searching functions (OPAC in a 'portal' environment). Since ABCD aspires at being more than a library system, but rather a general document management system, a very important part of the Central module deals with creation and editing of database-structures, fully based on the ISIS philosophy.

These functions are presented in different parts of a suite, which are relatively independent from each other but not fully. Such parts or 'modules' are accessed by their own URL. Within one part several submodules can exist which also cooperate. For example the pre-cataloging information produced by the acquisitions submodule will be re-used in the copies-database for the inventory and loan-objects database for the circulation module, which in its turn uses bibliographic information from the catalogs. Statistics can be applied to any ISIS-database, not only the circulation databases, so this function will also re-appear at several instances within the software. The ABCD OPAC-technology, inherited from the 'Virtual Health Library' system of BIREME, can run on any ISIS-database, not only the own ABCD-catalogs, so it will be described as a relatively independent tool, as will be the case with the Serials Control.

Important

How to access the suite parts or modules directly ?

- The first six (sub-)modules together constitute the 'Central' part of the ABCD-suite. It can be accessed by the URL http://[serverURL]:9090/index.php. The 'index.php' part is optional if the webserver (Apache) has been told that index.php is one of the default pages in the folder (as is e.g. also 'index.html'). Default login data are resp. 'abcd' and 'adm' for user and password.
- The combined OPAC-with-portal (Site) can be accessed by http://[serverURL]/site/index.php, with the administrator page for this Site being http://[serverURL]/site/admin/index.php. Default login data for the Site Admin submodule are resp. 'admbvs' for user and 'adm@bvs' for password
- The Serials Control part should be accessed by the URL http://[serverURL]/secs-web/index.php. The login data are, as before, resp. 'admsecs' for user and 'admsecs' (the same indeed) for password.
- EmpWeb can be accessed, if installed, by the URL http://[serverURL]/empweb/ (note the trailing forward-slash here !). Login data : user = 'admin' and password = 'empweb'.

For all these parts or modules ABCD provides start login data as given above. Important : It is the responsibility of the system manager to take these login data out of the system and replace them by local - and locally controlled - login data.

ABCD manages the control of who can access the system and with which privileges through a system (introduced with version 1.0) of 'profiles'. Profiles are sets of allowed modules, databases and forms. ABCD users (not library patrons) then will be assigned to one of the defined profiles (see 'Users Administration' in the following section). Functions/actions and modules not in the profile will not be accessible to all users with that profile. Only administrators can deal with such profiles and operator-accounts.

1.1. Login configuration of ABCD Central

The most important configuration file for ABCD Central is the file 'config.php' in the /www/htdocs/central folder. We discussed most of its parameters earlier on (in the section about installation) as they deal with installation paths and default language to be used, but at the end some parameters are introduced which provide a recovery solution for the case when the general System Administrator login data have been lost (meaning the system cannot be entered by anyone anymore !).

These parameters are :

> //USE THIS LOGIN AND PASSWORD IN CASE OF CORRUPTION OF THE OPERATORS DATABASE OR IF YOU DELETED, BY ERROR, THE SYSTEM ADMINISTRATOR
>
> $adm_login=""; $adm_password="";
>
> //USE THIS PARAMETER TO ENABLE/DISABLE THE MD5 PASSWORD ENCRIPTYON (0=OFF 1=ON)
>
> $MD5=1;

By manually editing (e.g. with Notepad or another flat-ext-ASCII editor) the $adm_login and $adm_password parameters, one can create temporary login data, which will allow to create real logins again (using the database administration tools for the users-database, named 'access'). It is strongly advised to immediately thereafter again remove the login-data from this file config.php - for obvious security reasons.

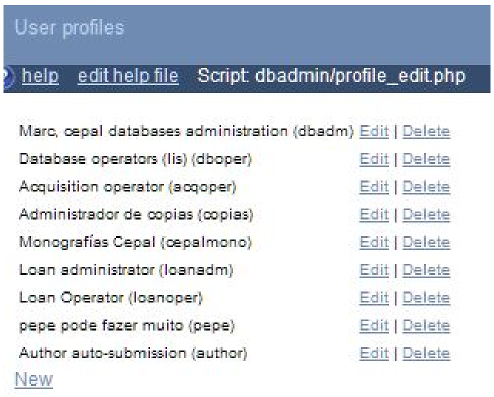
The $MD5 parameter will invoke password encryption (using the MD5 algorithm) when put to '1' or not if put at '0'. Therefore always after having changed this parameter (switched on or off), all existing passwords are no longer valid, since an encrypted password will be read as non-encrypted and v.v. This is an obvious situation where use of the temporary login-data in the 'config.php' file is necessary : login using these data, re-create passwords, at least for the System Administrator, and then re-login with these data for further processing the remaining passwords. This also means changing the use of encryption is a serious decision to be considered carefully !

1.2. Administration of the ABCD user profiles.

From the main Central menu option 'Users Administration' one can enter the 'Create/Edit profiles' option in addition to create/edit/delete system users. In fact this is mostly the first thing after finishing installation to be done by system administrators : to change the login data for the administrator and to create profiles and new operators linked to such profiles.

Some profiles come with the ABCD installation as examples, e.g. :

- System Administrator
- Database Administrator
- Database Operator
- Operator LIS



Marc, cepal databases administration (dbadm) Edit | Delete
Database operators (lis) (dboper) Edit | Delete
Acquisition operator (acqoper) Edit | Delete
Administrador de copias (copias) Edit | Delete
Monografías Cepal (cepalmono) Edit | Delete
Loan administrator (loanadm) Edit | Delete
Loan Operator (loanoper) Edit | Delete
pepe pode fazer muito (pepe) Edit | Delete
Author auto-submission (author) Edit | Delete
New

As can be seen from this screen, the administrator can edit existing profiles, delete them or create a new profile. For each profile the administrator has to enter the following data :

1. profile name : any (short) name for the new profile to be created
2. profile description : any description describing the profile

3. for each database listed : whether or not access is granted and which data-entry forms of that database can be used by this profile. If all, 'All' can be checked for simplicity. Database-related actions can be defined (allowed or not) for each database separately (as from version 1.2t).

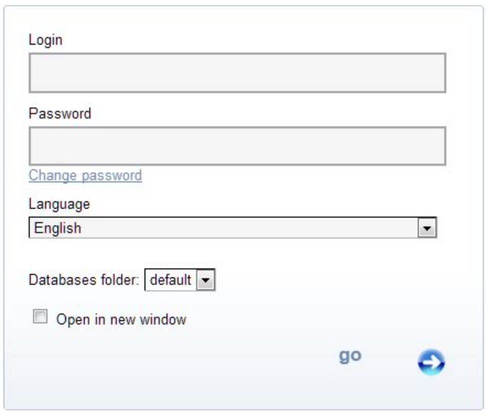
   Note

   Since some actions/options are defined by database, the icons for these actions cannot be adjusted (shown or hidden) at the initial main menu; the toolbar however will adjust since it appears only after database-selection.

4. Module permissions : for each module (Cataloging - Circulation - Acquisitions) the menu-option (or function) to which this profile should have access should be activated (in the small box).

1.3. Logging in into the system

After profiles have been created (or adopted from the default ones) and have been assigned to system users, any of the system users' logins can be used to enter the system. Unlike in previous preliminary versions of ABCD, it is obviously no longer necessary to select an authorization level when logging in, since this is now linked to the login itself. Therefore in addition to login-name and -password only the language to be used is to be selected (this list is taken from the file 'lang.tab' in the bases-folder of ABCD (default : /ABCD/www/bases). New features (discussed later) in v1.2t and v2.0 qre the choice of the main database-folder from a menu and the option to change passwords if the parameter $change_password is set to 'Y' .



Login

Password

Change password

Language: English

Databases folder: default

☐ Open in new window

go

This login-screen has one option 'Open in new window' ☐ Open in new window as a 'tickbox' for which the default value is defined in the general configuration file 'config.php' of ABCD-Central with the parameter-name '$open_new_window' which can be set to "Y" (meaning after login ABCD will use a separate window, which avoids interference with other tabs or wrong use of the 'BACK' button of the browser) or "N" to simply open the ABCD window in the same area; the other parameter linked to this behaviour in 'config.php' is '$context_menu' which also can be set to "Y" or "N" to resp. allow or not the window-menu invoked by right-clicking on the page - again this can avoid wrong use of the 'BACK' button as this going back should preferably be performed through ABCD-interface buttons, not the ones of the browser who has no control on certain necessary ABCD-navigation issues.

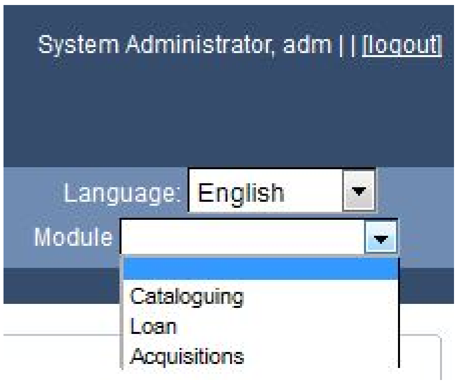
By clicking on the arrow ('GO') go the user will enter the main Central menu - adjusted to the profile granted. All options of this menu are discussed in the next sections of this manual. This includes the administration of users and linking them to a profile, since the users-database can be seen as just another ABCD-database for which general Database Management tools of ABCD are to be used.

2. Central module : database management

In this section we discuss the 'Central' module, which is the core of the ABCD system : this is the place where system managers and librarians will interact with the system most of the time. Such interaction is, technically speaking, about managing databases in one or more ways : creating them, maintaining them, filling them (e.g. 'cataloging') and using them for administrative purposes (acquisition, circulation, statistics).

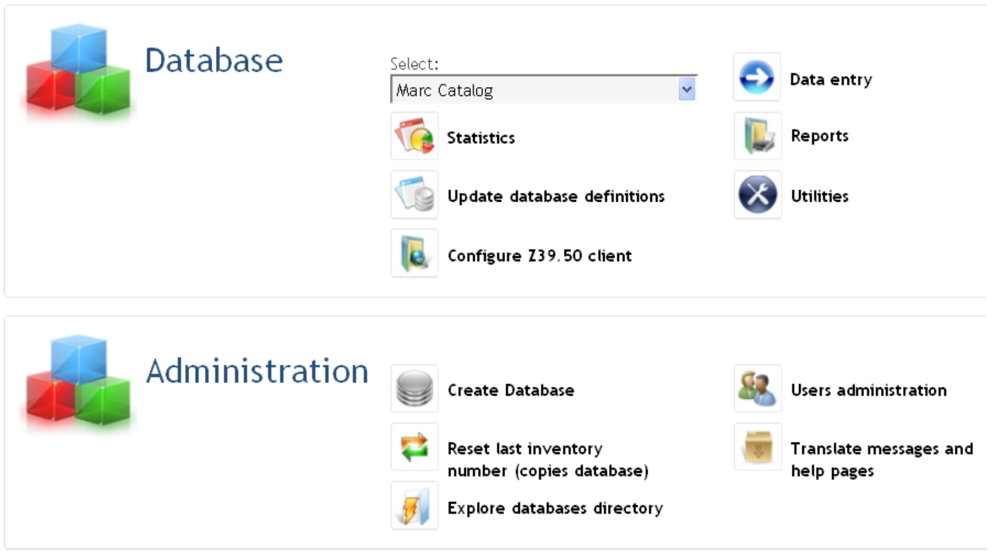
Within this Central module, there are three main sub-modules : Cataloging as part of database-administration, Acquisitions and Circulation. They can easily be opened from the main menu's right upper corner sub-modules menu, where also the language can be selected :

Figure 2.1. Sub-modules menu in Central



In terms of daily operations however Acquisitions, Cataloging and Circulation ('loans') will be much more important and take by far the largest share of 'operations time'. That is also why they got their own 'submodule' selection in this menu. Therefore we will deal with these more every-day's functions in separate chapters of this manual. So, in this specific system management section, we will review mainly the basic techniques of one of the most powerful functions of ABCD : creating new databases and modifying database structures. Since ISIS-databases don't require sophisticated 'normalized' relational structures and still can cope with elements in many-to-many relationships (like authors with publications), ABCD can be used to deal with any such 'locally' created database relatively easily. We recommend ABCD for environments where several such applications, like e.g. Institutional repositories, cultural heritage collections, vocabularies and ontologies or even just 'snippets' (loose textual information units), are likely to be created and used, and therefore need a flexible management tool.

We discuss in the following sections each of the options given in the main menu of ABCD Database management :



Note

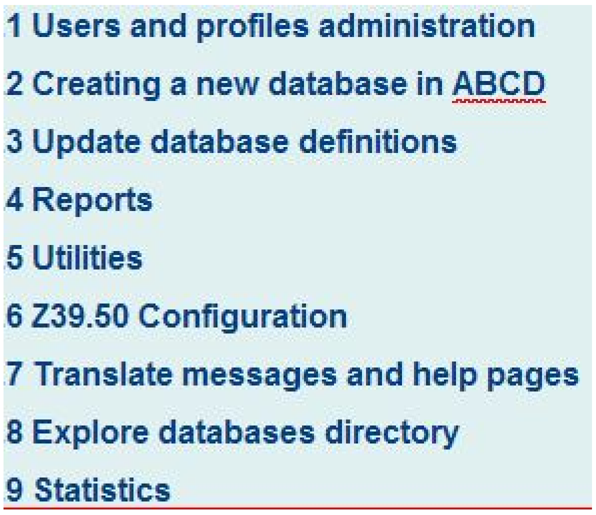
This menu is created in the PHP-script 'homepage.php' in the folder '\ABCD\www\htdocs\central\common' where each login level gets its own function to create the menu (e.g. function MenuAdministrador() for the system administrator), so if it is necessary to change the sequence of the functions of this menu, this file has to be edited by someone who understands the HTML-coding inside.

We prefer to discuss the options in this main menu in a slightly different sequence (which can be obtained also in the menu by editing the above mentioned 'homepage.php' script), because before doing anything else the Users Administration should be at least performed once to define a local System Administrator and probably (quite) some other system users. or 'operators' (as distinguished from library patrons or 'end users').

In view of the importance of the 'Data Entry' menu option here, which is in fact the 'cataloging module' of ABCD, a dedicated section of this manual will be devoted to it following the discussion of the other Central functions. Also the procedures to follow to create copies and loan-objects for the inventory resp. loan-databases will be explained there as they are part of the Data Entry function.

The resulting structure of this chapter of the manual largely co-incides with the main Central menu options, albeit that for the 'data-entry' section a separate chapter will be added.

Figure 2.2. Structure of Database Administration Chapter


1 Users and profiles administration
2 Creating a new database in ABCD
3 Update database definitions
4 Reports
5 Utilities
6 Z39.50 Configuration
7 Translate messages and help pages
8 Explore databases directory
9 Statistics

2.1. Users and profiles administration

We discuss the management of profiles and users separately even if they go together under one main menu option 'users management' : the users management requires profiles to exist first, since users have to be made members of a profile. Users can adhere to only one profile, but a profile can have mixed components (e.g. authorization for both acquisition and circulation sub-modules) and ABCD can maintain as many profiles as needed.

The screen following the selection of the 'Users Administration' option in the main Central menu will show 2 options :

- Users administration
- Create/edit profiles

The second option of managing the profiles is discussed first as these need to pre-exist before one can deal with operator records.

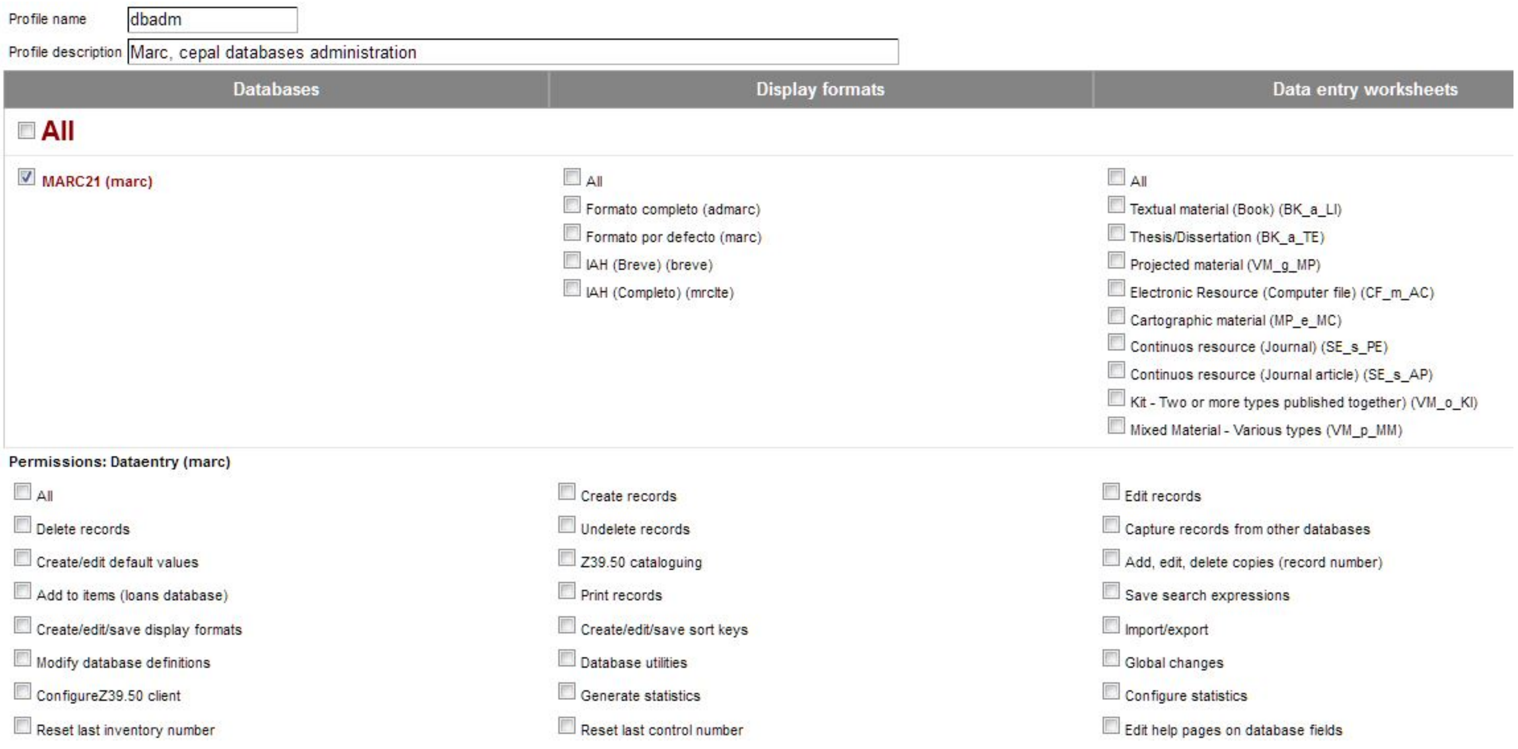
2.1.1. Profiles administration

An ABCD profile is a set of databases (with their worksheets and print formats) and actions of functions allowed for that profile. So the profile administration has two parts : a list of databases allowed to be accessed by members of this profile and a list of actions allowed to be performed by this profile. In fact the list of databases allowed to access will define the menu of databases to select from in the main Central menu, the authorized PFT's and FMT's (display formats and worksheets) will define the resp. lists which can be used in the right-hand upper corner menu's during data-entry. On the other hand, the list of activated 'actions' will define the menu items displayed when entering the system or opening a new menu.

For each available database - except special ones like the access-database itself which is only accessible by administrators - ABCD will present a list of display formats and a list of worksheets for that specific database, along with the database-specific 'actions' which can be allowed or disallowed in the profile. Whereas some special reserved databases (e.g. suggestions, loanobjects...) were not included in this list in previous versions from v1.2t, now all databases included in the 'list of available databases' can be selected or not for the profile. This creates some risks of which the administrator giving access to them should be aware : e.g. direct access to databases like 'copies' and 'loanobjects' allows operators to delete or add records without the built-in control mechanisms which ABCD applies to avoid creation of orphan records (e.g. a copy of a title which has been removed from the catalog is an orphan as it has no more parent in the catalog) or inconsistencies in between copies and loanobjects. In theory an operator can manually change the barcode of an object in the copies-database, but forget to do this in the loanobjects database. Such errors are normally not possible when using the special procedures to create copies from the catalog and loanobjects from the copies. The advice is here : beware of including databases in the list of available databases and profiles which are not needed really !

The illustration here shows an example of the MARC21 database, where the display formats, worksheets and actions can be defined for this database specifically.

Figure 2.3. ABCD Profiles administration



Profile name: dbadm

Profile description: Marc, cepal databases administration

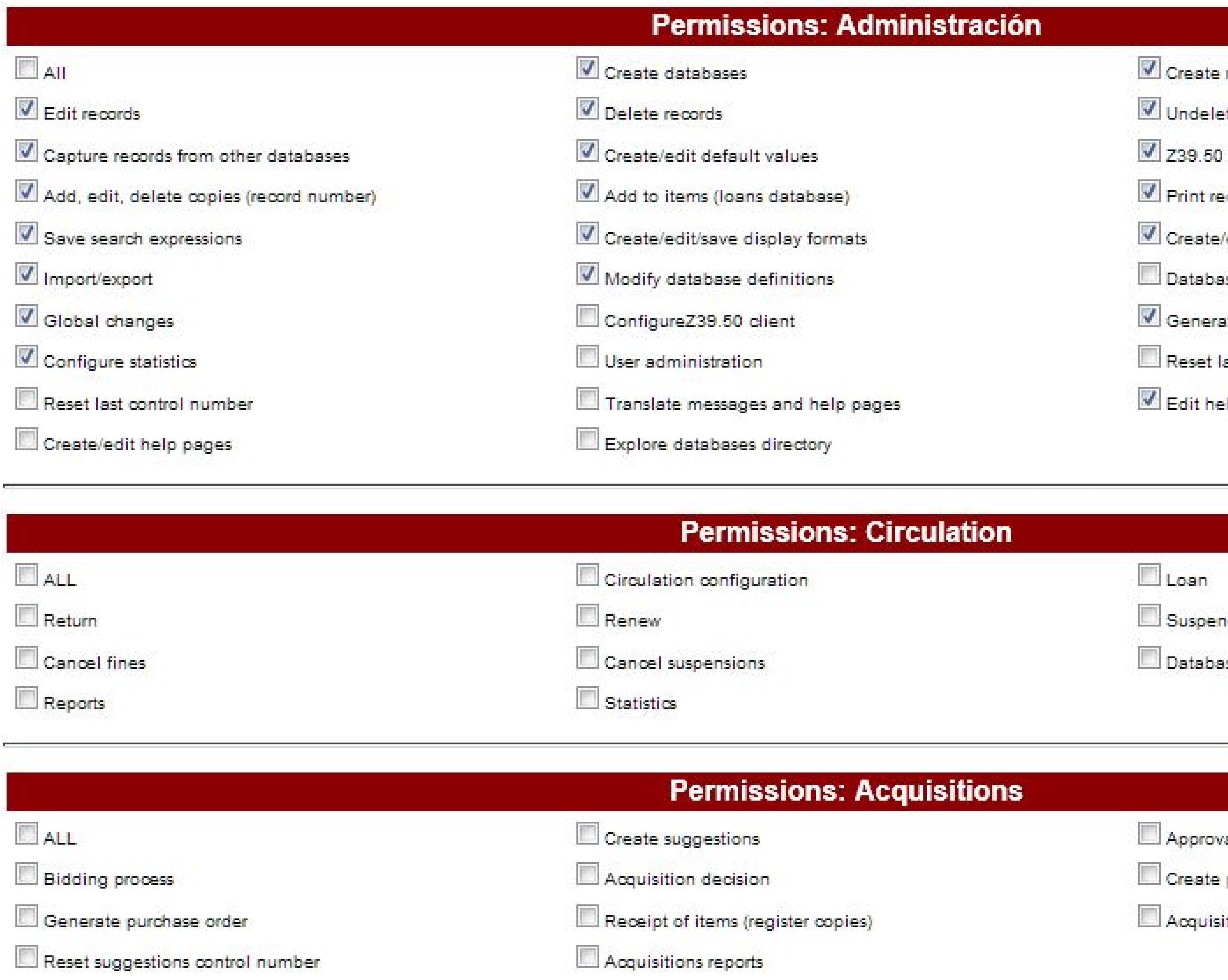
Databases	Display formats	Data entry worksheets
☐ **All**		
☑ MARC21 (marc)	☐ All	☐ All
	☐ Formato completo (admarc)	☐ Textual material (Book) (BK_a_LI)
	☐ Formato por defecto (marc)	☐ Thesis/Dissertation (BK_a_TE)
	☐ IAH (Breve) (breve)	☐ Projected material (VM_g_MP)
	☐ IAH (Completo) (mrclte)	☐ Electronic Resource (Computer file) (CF_m_AC)
		☐ Cartographic material (MP_e_MC)
		☐ Continuos resource (Journal) (SE_s_PE)
		☐ Continuos resource (Journal article) (SE_s_AP)
		☐ Kit - Two or more types published together) (VM_o_KI)
		☐ Mixed Material - Various types (VM_p_MM)

Permissions: Dataentry (marc)

☐ All	☐ Create records	☐ Edit records
☐ Delete records	☐ Undelete records	☐ Capture records from other databases
☐ Create/edit default values	☐ Z39.50 cataloguing	☐ Add, edit, delete copies (record number)
☐ Add to items (loans database)	☐ Print records	☐ Save search expressions
☐ Create/edit/save display formats	☐ Create/edit/save sort keys	☐ Import/export
☐ Modify database definitions	☐ Database utilities	☐ Global changes
☐ ConfigureZ39.50 client	☐ Generate statistics	☐ Configure statistics
☐ Reset last inventory number	☐ Reset last control number	☐ Edit help pages on database fields

After having defined PFT's, worksheets and actions by database, the sections for module-based permissions are presented, where again simply the actions can be selected to be allowed (or not) :

Figure 2.4. Module based permissions



Permissions: Administración

All	[✓ Create databases]	[✓ Create r]
[✓ Edit records]	[✓ Delete records]	[✓ Undelet]
[✓ Capture records from other databases]	[✓ Create/edit default values]	[✓ Z39.50]
[✓ Add, edit, delete copies (record number)]	[✓ Add to items (loans database)]	[✓ Print re]
[✓ Save search expressions]	[✓ Create/edit/save display formats]	[✓ Create/]
[✓ Import/export]	[✓ Modify database definitions]	Databas
[✓ Global changes]	ConfigureZ39.50 client	[✓ Genera]
[✓ Configure statistics]	User administration	Reset la
Reset last control number	Translate messages and help pages	[✓ Edit hel]
Create/edit help pages	Explore databases directory	

Permissions: Circulation

ALL	Circulation configuration	Loan
Return	Renew	Suspen
Cancel fines	Cancel suspensions	Databas
Reports	Statistics	

Permissions: Acquisitions

ALL	Create suggestions	Approva
Bidding process	Acquisition decision	Create
Generate purchase order	Receipt of items (register copies)	Acquisit
Reset suggestions control number	Acquisitions reports	

Using the combinations of database-access, database-print formats and -worksheets access and a rather granular authorisation of actions (e.g. there are separate authorisations for delete and undelete records), we believe this should allow administrators to arrange access to the system in a way well-suited to their workflow and organisation. Also it is possible to define e.g. an 'auto-submission' profile for authors in which a repository-database has active access (e.g. create/edit/delete records) but other databases only have 'viewing' (passive) access.

Note

The profiles are kept in the subfolder 'profiles' of the folder bases/par. One can find in that folder one file for each created profile, which is a text file with in there key-value lines. A list of profiles is kept in the file 'profiles.lst' and a list of actions in 'profiles.tab' (using the usual 2-column for internal and display-values for each action). Whereas ABCD normally takes care of these files, in principle it is possible to edit these files manually, if sufficient caution is applied to avoid logical conflicts or wrong parameters. A simple example of such key-value is the first line indicating the name of the profile : '`profilename=dbadm`'. There after one will find the list of allowed databases (e.g. '`db_marc=marc`'), with the definition of PFT and FMT's allowed (e.g. '`biblo_pft_ALL=on`' will allow all display formats for this profile and '`biblo_fmt_mm=mm`' will allow the 'mm' worksheet), followed by action-authorisations, e.g. 'CENTRAL_IMPEXP=Y' will allow import/export in the Central module.

In the folder bases/par/profiles two files can be found keeping track of the profiles defined : profiles.lst (with for each profile an entry like 'author|Author auto-submission') and profiles.tab in which all available 'permissions' are set, e.g. DELREC=Y to allow deletion of records.

A typical 'special' profile, requested by many ABCD-users, is a profile of catalogers who can also create copies for the inventory and loanobjects for the circulation system. For this to work, one has to add the copies-database to the list of allowed databases and the actions 'Add, edit, delete copies' and 'Add to items' (which are loanable objects).

Once suitable profiles are created in the system, the administrator can deal with user (operator) records to assign to the said profiles.

2.1.2. Users administration

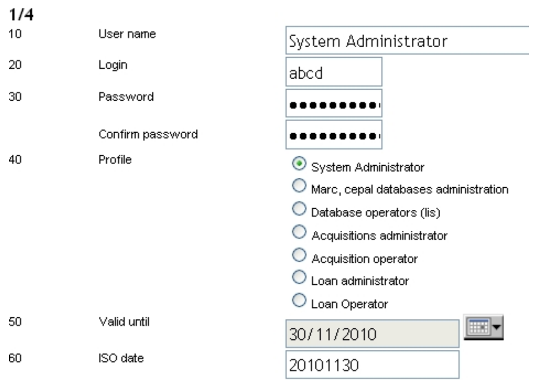
The Users administration option of the main ABCD Database Administration menu is a specific case of database-management, using mostly the general techniques discussed in this section, but for a specific database 'ACCES' in which only the System Administrator can create profiles and ('register') new users or edit them.

Note

IMPORTANT ! Before doing anything else, ABCD should get, by using this Users Administration option, a new, local System Administrator with his/her own login data ! The default login 'abcd/adm' will be widely known as it is published, so doesn't give *any* security indeed *!*

The option on managing users is presented by first showing the existing users (there should be at least one 'System Administrator' user !) and giving the options to either edit these, delete or add (create) a new user.

When clicking on the 'record edit' icon (first one of the three presented for each user :) the record with the user data will be shown in an interactive edit-form :



1/4

10	User name	System Administrator
20	Login	abcd
30	Password	••••••••••
	Confirm password	••••••••••
40	Profile	[✓ System Administrator] Marc, cepal databases administration / Database operators (lis) / Acquisitions administrator / Acquisition operator / Loan administrator / Loan Operator
50	Valid until	30/11/2010
60	ISO date	20101130

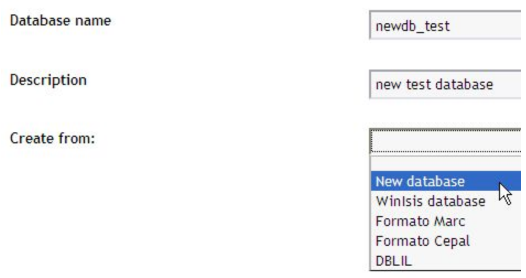
This edit form has the following parts :

1. the user name, which can be a full name
2. the login to be used in the login screen, mostly a shorter name
3. the password for this user (for encryption check the $MD5 parameter of config.php, discussed above)
4. the profiles which have been created and which can be assigned to this user. A set of demo-profiles is included with the ABCD-installation package.
5. the 'expiry' date for the current user, in the 'normal' date-format (as defined in config.php) and the mandatory ISO-date format which will be created automatically by the software itself.

Once the form is properly filled in and saved, immediately this user can login and access the authorized parts of ABCD Central.

2.2. Creating a new database in ABCD

After selection of the 'Create Database' menu option from the main menu of the Central module, the following 3 elements need to be specified :



Database name	newdb_test
Description	new test database
Create from:	

In the first box the software asks for the 'name of the database', which will be the real internal file name of the new database. These names no longer are confined to the old-style '6 characters' name of CDS/ISIS or WinISIS, but short names are still preferable. The name as presented to the users will be specified in the 2nd box : the 'description'.

Tip

Database names and descriptions can be approached directly in the file 'bases.tab' in the folder \ABCD \www\bases. In this file each database, provided to users, has one line with each two values : the 'name' and the 'description', separated by a pipe ('|') . A third pipe, followed by 'Y', is used to indicate if a database has been identified as a catalog for the Loans system of ABCD.

The 3rd box will always provide the options 'new database' - meaning creating a database from scratch - and 'WinISIS database' - meaning copying an existing structure of a (Win)ISIS database or in fact any ISIS-database with a FDT, FST and PFT. Then also the existing databases will be provided as models to be used as the basis from which to create the new database. We only deal with the first 2 options, as copying from an existing ABCD-database is quite straight-forward (ABCD simply creates the database by copying all necessary files into their appropriate folders and adding the new database to the list of existing databases).

The creation of a new database 'from scratch', meaning : not based on an existing model but starting from a zero-basis, involves understanding quite some ISIS-techniques, esp. the Formatting Language, because this will be used not only in the creation of the presentation format of the new database, but also in several ABCD-specific attributes of the fields (in both the FDT and data-entry worksheet) and the FST for indexing.

2.2.1. Creation of a new database from scratch

The possibility for ABCD administrators to create their own databases with their own structures is a not-so-common feature in library automation systems but is directly derived from the basic database-philosophy of the ISIS-software family : a non-relational, 'no-SQL' [see : https://en.wikipedia.org/wiki/NoSQL and http://nosql-database.org/] or 'schemeless' approach resulting in a high flexibility without giving up on power in those areas where it really matters in document-oriented systems : information retrieval and capability to deal with complex document-structures.

The users who want to create a database using ABCD need to be aware of the fact that some efforts will be needed : both in understanding the con's and pro's of many options to take (e.g. a simple bibliographic structure or a very complicated one like MARC ?) and in how to correctly apply these into ABCD. ABCD tries to make it as easy as possible but still real efforts will be needed. In view of the potential benefits, like e.g. the possibility to create local databases for specific projects next to the general library catalog (as practiced with CDS/ISIS software in a countless number of larger scientific institutes where the library uses a 'classic' - but 'closed' - system) and integration very different resources into one environment, as can be done with ABCD esp. using the Site. But it has to be acknowledged that creating a database from scratch is not a daily activity, most administrators will do this only once or twice in their career, so an additional effort to cope with the complexities and many error-prone actions in the process of creating a database, can be justified.

This feature at the same time also creates some complexity and risks for errors in the ABCD software as such : without knowing in advance which exact structure(s) will be used to deal with , all scripts and functions need to be programmed very carefully in an open way to accommodate all thinkable options and possibilities. Nevertheless this feature has now been used by quite some ABCD-users to good and quite stable results, which should encourage administrators willing to engage into database-creation.

Needless to add that for educational purposes, e.g. training students in librarianship or information science, this option without doubt is one of the most useful ones : students can be taught not only about bibliograhpic standards (e.g. MARC21, CEPAL) but also about principles of database structures, fields, validation, user-machine interaction and many more aspects of 'the art of information management'.

We will introduce some ISIS-database concepts here and explain how they are implemented into ABCD, but restrict thereafter the example database-creation to a bare-minimum, allowing the initialisation of a database. Further on in this manual there will be more detailed discussion with examples of more advanced database-elements.

2.2.1.1. The main hierarchical levels of databases

Databases in most cases come as part of a 'database management system' in which many databases can exist. This is the highest hierarchical level therefore. Within a DBMS databases exist, and within databases more or less structured information is grouped into units called 'records'. As in a programming language a record in memory management is a 'pre-structured' container of several distinguished pieces of information, the same concept exists in e.g. bibliographic records with as the pieces elements like title, author, publisher, keywords etc. In a typical administrative database a record will contain information such as name, address, age, sex etc. into combined units : again this is the idea of records.

Going down one level in the hierarchy gives us the 'fields' which reside into a record. In most databases fields are the smallest - lowest level - units inside a database the software can access, resulting in a 4-level hierarchy : database-system, database, record and field. Talking in terms of real physical units like people, one could say : people live in the Universe (the DBMS) on Earth (the database), within Earth they live in a continent and/or a country, within a country there is a municipality unit, in which streets are located and within the streets we have buildings. Within the buildings people can live e.g. on a certain floor in a certain room. So we have surpassed the 4-level hierarchy already by far.

So, in order to allow even more precise 'localisation' of information units, ISIS from the very beginning added at least one structural level : subfields. This idea fully complies with the ISO-2709 and MARC concepts or is taken from these standards as a matter of fact. Moreover, inside subfields (which 'live' inside fields which are parts of records as parts of a database) ISIS can also 'recognize' words, not as a structural part but at least in some indexing mechanisms words can be counted and their position used for proximity-based searching.

Only 'XML' (eXtensible Markup Language) goes even further in allowing sub-dividing into more hierarchical levels : in fact due to the aspect 'extensible' anyone can define any number of hierarchical levels, suitable for e.g. describing journals : from a journal title to volumes, going down to issues, inside the issue as one specific document there are 'articles' which can have parts, inside parts chapters, inside chapters maybe paragraphs... already reaching 7 hierarchical levels.

Each of the structurally identifiable elements offer 'handles' to get better grips to the information in the system and adds power to retrieval mechanisms, e.g. searching within specific fields, subfields or full-text indexing (=indexing by word). When defining or creating a database, administrators should consider all this, as the input-software, e.g. providing worksheets, defines the 'granularity' of the information stored and for both indexing and presenting the information this granularity can be used to a certain 'less or more' extent. In reality information can be stored as 'snippets' (unstructured text like in a 'personal information system' as the computer equivalence of a fridge with post-its sticked on) up to highly detailed units with hundreds of fields as in MARC. This manual is not a course on documentary information management but needless to state that its principles are applicable or even unavoidable to consider when working with ABCD as a database-administrator creating new databases.

2.2.1.2. The main constituting 'structures' of an ABCD database

Since ABCD-databases are simply full-fledged ISIS-databases, we inherit these main constituting parts from the ISIS-technology, and we will discuss them here to explain the principles and the basic part needed to allow database-creation - for more detailed discussions we refer to the section on 'Editing/Updating database definitions'. These main parts are respectively :

1. the Field Definition Table or FDT

   Before one can start using a database, one needs to know which elements can be used within the database. The concept of fields, generally used in all databases, perhaps is sufficiently explained by simply giving the usual examples of 'title, author, publisher' in a bibliographic record or 'name, street, phonenumber, e-mail' for personal address systems. But for the concept of subfields one has to engage into still ongoing debates on advantages and disadvantages of subfields, on which one has to take a stand when creating a structure. Generally speaking, and without going really into this intellectual debate, one can say that keeping elements together as subfields into repeatable units is the preferred way. E.g .in order to never mix first names (Karl and Vladimir) with last names (Marx and Lenin) it is safer to keep them together as Marx+Karl and Lenin+Vladimir whereas the '+' - or any clearly marked 'separator' for subfields, ISIS using '^' followed by a one-character identifier - allows the software to also deal with the elements separately, e.g. indexing as 'Marx, K.' but also as 'Karl Marx'. In many languages the author names are not that easily recognisable as for their first- and surname parts, so from a 'user-friendliness' point of view this certainly makes sense.

   Whereas in most databases fields are to be identified clearly and will always be present in the 'tables' (representable as rectangular 'matrices', so they need to keep their rectangular form by keeping the substituting elements constant), in ISIS one can consider the Field Definition Table (FDT) rather as a repository of possible fields, which can yes or no be present in the record. In fact one can even use fields - identified by their numerical 'tag' - in ISIS without having described them in the FDT, but then of course not allowing all possibilities of fields present in the table (e.g. knowing the 'name' of a tag). This is one element leading to the earlier mentioned flexibility.

   One more general characteristic of the ABCD FDT is that , differently from other ISIS-family members, it contains also the worksheet specifications for each field : which graphical form-element will 'take' input from the cataloger in which way, e.g. a 'normal' text/textarea box or a larger HTML-area with full HTML-editor buttons etc. Again one should consider the description in the FDT as rather the definition of the 'default' input behavior of the (sub-)field since it is always possible to create worksheets with different specifications for the same field. We refer to the discussion of the worksheets later on.

   Currently ABCD has opted for a spreadsheet-like presentation of the FDT, using an advanced JavaScript-library (dhtml-grid) which allows editing within 'cells' of a grid within the WWW-page.

   Remark : Unfortunately only Firefox and IE browsers support(ed) this advanced tool fully (e.g. Chrome and Opera presenting the grid incompletely) and it looks like even IE in its now new version 10 has dropped this full support. leaving only Firefox as a guaranteed correct handler of the grid. A new approach will need to be developed for this reason for subsequent new versions of ABCD. Again it can be stated that since database-creation and editing FDT's is not a daily activity, and only for System Administrators, this flaw should not be overrated and doesn't prevent the use of e.g. Chrome or IE for ABCD daily activities and end-user work.

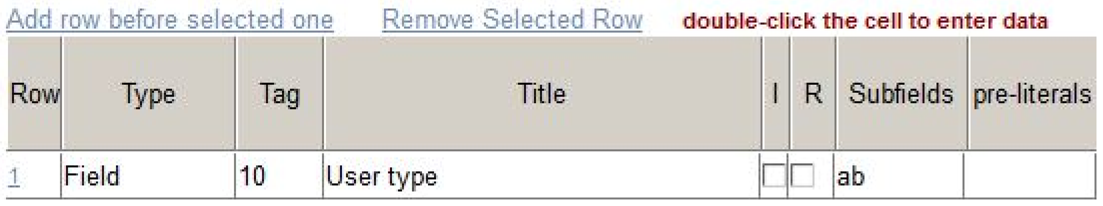
   ### Note

   For some features of the FDT to work correctly, the option '**allow pop-ups**' for the current URL (e.g. 'localhost:9090) has to be activated since the menu's within grid-cells are technically build as pop-ups. After having activated this option - most browsers warn if this is not the case - one simply should reload the page in order to see all elements correctly.

   Having raised all this as an introduction, let's now consider the respective elements of the FDT one by one, starting with the field characteristics *sensu stricto* and followed - in the discussion of the worksheets under here in no. 3. - by the worksheet characteristics for the given field.

   In the illustration here we only show the left-hand part which actually deals with the real FDT-elements - all elements or columns to the right of this part deal with worksheet elements.

Figure 2.5. ABCD FDT editing in a row


Add row before selected one
Remove Selected Row
double-click the cell to enter data

Row	Type	Tag	Title	I	R	Subfields	pre-literals
1	Field	10	User type	☐	☐	ab	

- first column : the row number, which is also the (blue-colored) link to open the row in a full form
- second column : field TYPE, being one of the following options :
  - Field : a normal field without any special features
  - Subfield : the row defines a subfield belonging to the previously defined Field
  - Fixed field : special fixed-structure field as used in MARC21 field 8
  - Date field : containing a MARC-date (v5)
  - MARC-leader field : a special field for the MARC-leader, see MARC references elsewhere
  - Group : a repeatable subfielded field - by far the most complex but also useful field for which we will give more details later on
  - Line : this option will only create a visible 'line' as a separator in worksheets based on this FDT
  - Heading : this option will only print the 'name' of the field as a heading subdividing the worksheet; collapsing/expanding sections of worksheets is based on these headings.

    Note

    The list of fields is reduced as compared to earlier versions pre 1.2t. This 'incompatibility' is coped for by the software and by re-saving an old-style worksheet the necessary move to the 'data-entry' type of incompatible field-types will be automatically performed.
- third column : the 'tag' of the field is the numerical identifier of the field, from 1 to max. 999 (higher values can be used for special internal and virtual fields, not explained here); no leading zero's are accepted here, so put '1' and not '001'
- fourth column : the 'title' or 'name' of the field which clarifies the use of the field since the identifier is only a number; in ABCD one should keep the ID's constant for the different language versions of the FDT, while adapting the 'title' for each language is possible here
- fifth column : the 'I'dentifier tick-box which denotes, only one for each database, the field to be used for the browsing lists (A-Z) in the Central module. This works together with the value of the prefix-column and 'List as' columns to construct the strings which will define the browsing lists
- sixth column : the 'R'epeatable field tick-box which indicates whether the field is repeatable or not; this will e.g. automatically create repeatable groups when the field is included in an automatically generated PFT (see infra)
- seventh column : pre-literals are mostly special characters (interpunction) like comma, semi-colon etc. which should be used to split incoming values into subfields. When the record is written, the pre-literals will be replaced by the subfield delimiters. This way the data-entry for fields with few subfields can be made easier, e.g. the name of an expert Name, Firstname will still be stored as Name^nFirstname or ^aName^nFirstname, depending on the data specified in this parameter for Subfields and Pre-literals.

2. the Field Selection Table of FST for indexing instructions

   Differently from other ISIS-versions, ABCD as being fully based on CISIS allows the use of more than one general index on a database. E.g. when defining the configuration for the OPAC (iAH) one parameter to be defined is the name of the index to be used, which often is just the default index with the name of the database (referred by the placeholder 'DATABASE' in the iAH-configuration). Therefore not only in theory but also in practice one can conceive different FST's for different purposes, e.g. one for detailed searches by catalogers/experts and another one with the popular search fields or even a virtual 'full-text' field for end-user searching. In reality ABCD offers already those options by providing a search grid for the catalogers in their own toolbar of the data-entry part of the software and one or two others for searching in the OPAC, but also preserving a 'Google'-like option of simple search with words across the records (the 'virtual' field idea referring to one prefix used for all selected 'full-text' fields).

   But ABCD wouldn't be a real ISIS-member if it would not allow to do all this still with one overall FST, combining entries for advanced searching in one interface with simple-search entries in another interface ! Still one should consider, especially in high-density multi-user cataloging environments, that creating such complicated indexes when saving a record becomes less trivial and in poorer server-environments (e.g. a PC) can lead to overloading the system. In such cases the solution is to provide a 'basic' real-time FST definition with fewer option (taking less load on the CPU and storage) with other FST(s) for batch indexing at quiet times (e.g. at overnight with auto-scheduled scripts).

   In concreto, creating an FST requires understanding of each of the three 'columns' of the FST itself : the index identifier, the indexing technique and the extraction format. For all three elements one has to basically understand that in the case of indexing in ISIS this means creating 'entries' of a dictionary, listing all searchable elements extracted from the database with or without additional elements. In ISIS applications such dictionaries or listings can always be presented as a 'menu' to select search keys from.

   a. the indexing IDentifier

      The FST ID is a numerical value identifying the 'origin' of the dictionary term. In most cases this is simply the field tag from where the value is extracted, but in two specific cases the identifier deviates from this idea for very good reasons :

      - if the term is constructed (by the format in the extraction column) from different fields, even from other database-values with the REF() function, the ID becomes an artificial or virtual field tag. It is best to use tags not present in the database to avoid confusion, e.g. '999' is often used for such purpose.

      - if one field is indexed more than once, with different techniques (2nd column), one can assign again a non-existant ID as a virtual one to distinguish the different entries. However mostly in ABCD the use of different 'prefixes' is preferred. E.g. a title field can be indexed as such (technique 1) with an identifier '245'(the MARC21 title-field tag) but also by each title-word separately, e.g. using an ID of '222'.

   b. the indexing technique

      As in all ISIS implementations, and therefore described in all ISIS-related manuals, there are several different ways of 'extracting' values from (sub-)fields in the databases. We confine ourselves to a very short description here, leaving today rarely used techniques in between parentheses :

      - techniques 0 and 1 extract the whole field as-is up to the length of the inverted file keys, in ABCD this is currently 60 characters. The difference in between 0 and 1 is not that important : technique 1 'knows' the subfields and will create separate entries, technique 0 does not.

      - techniques 2 and 3 deal with 'marked' terms from the field values, the marks resp. being the slashes ' / ' and the brackets '<' and' '>'. Esp. these last needs caution since the WWW-technology of the HTML standards has introduced, long after ISIS was designed, these markers for their essential tag-marking. Luckily ISIS has a way of forced preserving the brackets or hiding them with the Formatting Language. In reality these techniques are not used anymore that much, also because the marking itself of words/terms in fields in itself represents a quite labor-intensive job and people prefer the 'cheap' full-text indexing, see next.

- technique 4 : each word will individually be extracted from the (sub-)field and matched against a 'stop-words' list and if not in there retained into the dictionary for 'full-text' searching. Optionally - but specifically to be indicated by adding 'repeat' separators (e.g. '%' in case of the default repeat character in ISIS) - one can also count words and remember their position for 'proximity' searching, only selecting records where the two terms exist within a given distance of 'x' words, as known by their position in the field.

- techniques 5-8 are in fact the same as 1-4 but WITH the use of 'prefixes'. Prefixes are short text-tags which are added in front of the extracted strings (therefore 'pre-'fix) with the effect of putting the term in a specific position of the alphabetically sorted dictionary, keeping all entries with the same prefix together. Without going into details here this can speed up retrieval processes a lot esp. in case of very large databases and Boolean combinations where after the intial selection of the hits the software still has to 'match' all entries for the Boolean condition or a field-specification. That condition or field-identifier can be anticipated by the use of a prefix, eliminating the need for a 'second round' in the selection process. Typical prefixes in ABCD are e.g. TI_, AU_, DE_ etc. for respectively titles, authors, descriptors.

Note

In the ABCD iAH module the use of such 3-character strings as prefixes ending with underscore is mandatory.

c. the extracting format

Simply put the extraction format, which will create a 'string' to be put into the dictionary as an entry, is a 'PFT' and therefore it uses all possibilties - and they are many, too many to discuss here - of the ISIS Formatting Language. In reality it can be anything from the simplest format, e.g. 'v1' (which will just take the value from field 1) to any construction of strings possible with the Formatting Language, including making references to other records and other databases. Since in most cases this format is kept simple and short, we will confine here by giving the most typical example : a field value built by subfields, prefixed for authors as expected in ABCD iAH with two-character-plus-underscore prefix and generated by method 5 :

```
'/AU_/', (v100^a/)
```

which has to be understood as follows :

- first the prefix `AU_ is both quoted with single quotes and embedded in between a delimiter, which can be any character not occurring inside the prefix-string itself, in our case the slash ' / '`

- a comma, separating the prefix definition and the actual extraction format

- a repeatable group in between parentheses (and) to indicate that this extraction needs to be done for each occurrence of the author-field

- the actual format 'v100^a'' which will extract only the subfield a part of field 100

- the slash again, this time used as the Formatting Language 'repeat' command, in order to separate all author names as different entries in the vocabulary by putting each occurrence on a 'new line' - which is the official meaning of the slash / in the ISIS Formatting Language.

  Note that if you want e.g. title words (prefix : TW_) indexed with position kept, the format would be : `'/TW_/',(v245^a|%|)`

Most of the FST-entries will look rather similar but applied to different field tags. Let's conclude, for fun (not really...!), by looking at one still-not-too-complicated example from the demo MARC21 FST with some more programming-like content :

```
902 0 mpu,(if iocc<100 then |PR_|v900^r|%|/ else break fi)
```

Here you can see the identifier 902 is assigned to elements coming from v900, technique 0 is used for simple field extraction, and this is applied to v900^r but only for the first 99 occurrences (iocc<100) of that field. The occurrences need to be counted indeed so the occurrence separator '%' is added after each repeat. As said before, it can get much more complicated with more programming logics, replacements, external database lookups etc.

3. the worksheet or FMT descriptions

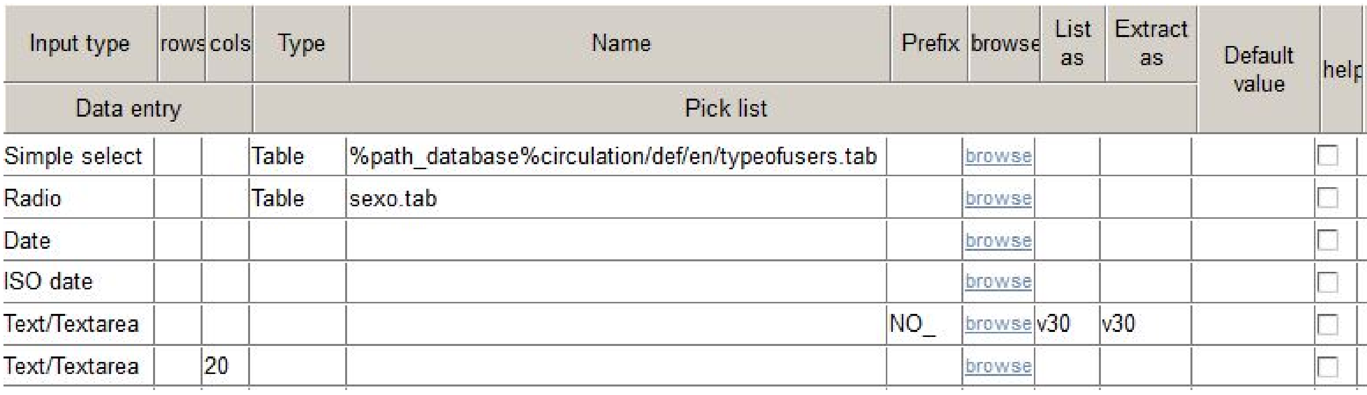
   Worksheets are 'forms' (in HTML-coded format) which are automatically presented by the ABCD software each time information is to be gathered from the operator or user in order to act on it, like mostly cataloging does. In ABCD the worksheet description can appear at two instances :

   - as part, i.e. the right-hand part, of the FDT for each row (field) in the grid
   - as a dedicated, separate worksheet

   The principle is as follows : in the FDT the 'default' data-entry characteristics of the fields are defined, whereas specific additional (non-default) worksheets can be defined at all times, containing subsets or all fields again, in the same or different sequences, but possibly with different characteristics. E.g a field which is presented as a table-based picklist by default, can be presented in another worksheet as a simple text-area box for direct editing. A new feature (as from v1.2t) is that when the administrator wants changes in a worksheet to be reflected in the FDT, a new last column allows 'linking to the FDT' for the field concerned.

   In both cases, whether in the FDT or in a FMT, the presentation of the 'rows' with all elements is the same, as is the way of dealing with the elements : double-click to activate a cell and edit. By clicking on the blue number in the first cell of each line, the line will be presented in a separate 'vertical' form presenting the same row-elements (columns) in a form for more detailed editing if so desired. We show both formats here as illustration.

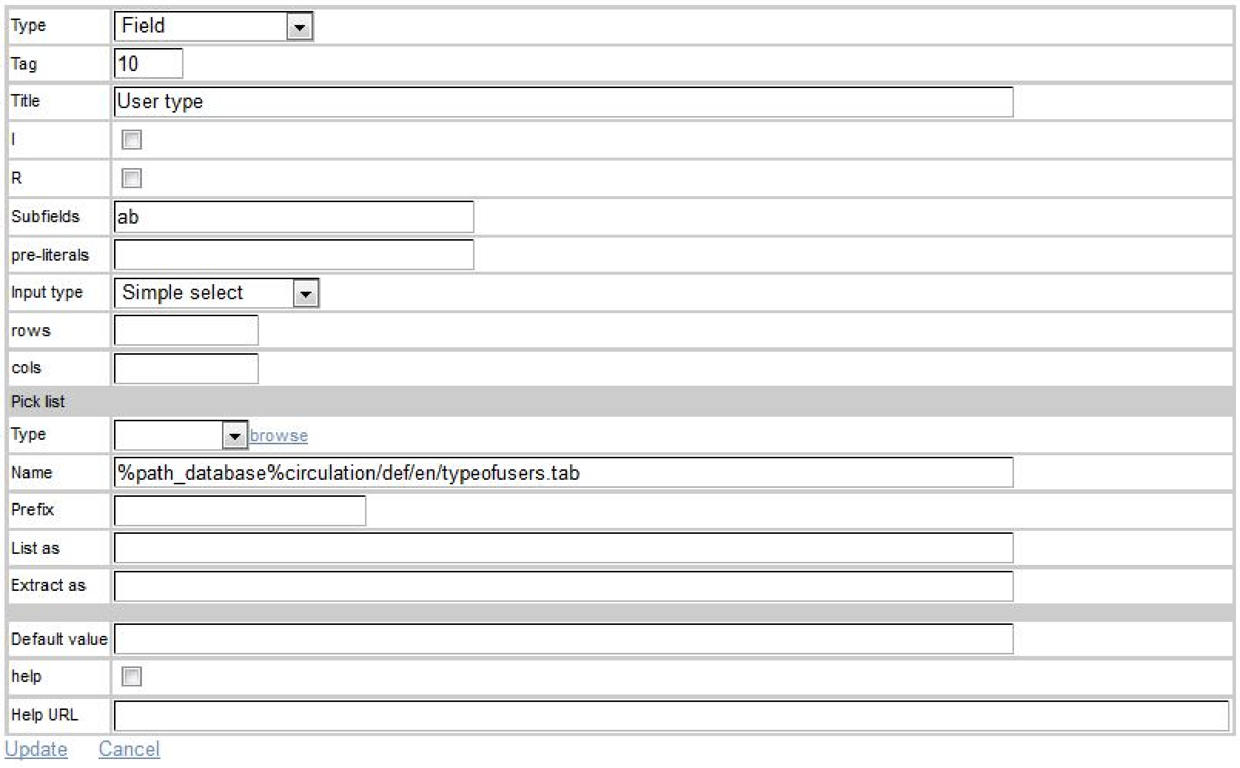
Figure 2.6. Worksheet row editing presentation



Input type	rows	cols	Type	Name	Prefix	browse	List as	Extract as	Default value	help
Data entry			Pick list							
Simple select			Table	%path_database%circulation/def/en/typeofusers.tab		browse				☐
Radio			Table	sexo.tab		browse				☐
Date						browse				☐
ISO date						browse				☐
Text/Textarea					NO_	browse	v30	v30		☐
Text/Textarea		20				browse				☐

When clicking on the far-left blue-coloured '1' the same row will be presented in a form as illustrated here :

Figure 2.7. Worksheet row-as-a-form editing presentation



Type	Field
Tag	10
Title	User type
I	☐
R	☐
Subfields	ab
pre-literals	
Input type	Simple select
rows	
cols	
Pick list	
Type	browse
Name	%path_database%circulation/def/en/typeofusers.tab
Prefix	
List as	
Extract as	
Default value	
help	☐
Help URL	

Update Cancel

If preferred one can do all editing work in this form and click on 'Update' to bring the data into the normal grid.

The elements themselves are explained further on when presenting the actual possibilities of 'Editing/Updating database definitions'.

4. the print formats or PFT definitions

   Following a 'law of documentary systems' which reiterates a well known political science law : the 'separation of the main three powers', ISIS and ABCD separate carefully the three main aspects of raw data input, the internal processing (indexing, formatting..) and the output. Unless a special built-in 'raw' display format, outputting records as they are without any processing, all output will be based on 'print formats'. Output in this context is to be seen as regardless to which 'device' : whether the screen (the real 'display'), a file on a storage device, a printer or another process (e.g. sorting, indexing, exporting), ISIS and ABCD will always 'mould' the output according to rather specific instructions as scripted in the Print Format Table or PFT's. A full 'language' has been developed to this end : the ISIS Formatting Language. The CISIS-Formatting Language, published separately, in its own right contains more than 40 pages, so we won't repeat the detailed discussion of this language with its commands and functions here.

   Basically the PFT contains instruction on what and how to display. Displayed elements can have two origins :

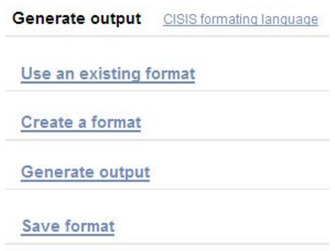
   - taken from a (sub-)field of a record in an ISIS-database, therefore the 'V'alue of a field with a given tag, e.g. `V1` is the value of field with tag 1.
   - defined by the PFT itself as a 'literal' (a string defined in the script itself) either as straightforward 'quotes', e.g. the word 'Title : ' before displaying an actual title, or as programmed strings, e.g. numbers of items in a looped list. In ABCD as a WWW-based software, the most used literals are actually HTML-tags : by quoting opening tags before and closing tags after a field value, the field value becomes an HTML element, e.g. `'<td>'v1'</td>'` will produce the same value of field with tag 1 inside a 'table-data' HTML-table, or as an example of the PFT defining 'how' to display : `'<b>'v1'</b>'` will display the value of field 1 in **B**old Face.

Especially in the Windows version of ISIS (WinISIS) many graphical attributes were added to the PFT language, e.g. defining colors, fonts, margins etc., but in ABCD these are replaced by WWW-equivalents as HTML-tags to be literally quoted.

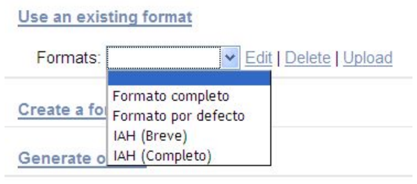
Actual ISIS PFT's are seemingly quite complicated concatenations of many such elements, each one in itself rather simple to understand as they are produced by a limited list of commands. With some efforts also non-computer experts can learn the language very well, as proven by so many marvellous PFT's developed by librarians all over the world.

The good news is that ABCD doesn't even require the librarians to make this effort as it generates PFT's according to some few pre-defined styles applied to a selected and ordered list of fields defined by the operator.

The PFT-Editor has 4 parts :



Use an existing format : a list of existing PFT's will be available to select from. It can be also deleted or uploaded from an external file if not yet available. The format then will be presented with an editor to make changes into it.

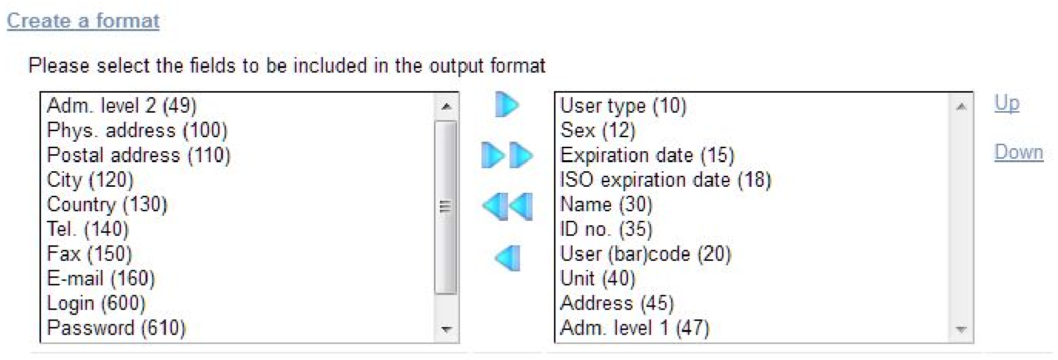


In reality the ABCD-operator has to run, for **creating a new PFT**, through 4 relatively simple steps :

a. selecting the fields to be displayed

   In case an existing worksheet needs to be edited, it can be selected (in the upper part of the interface) from a list - where possibly also an existing worksheet can be deleted is so desired. The main part of the worksheet presents the list of available fields with possibility to copy any field (▷) or all fields (▷▷) to the worksheet list at the right side.

   As was already the case in WinISIS, available fields are listed at the left hand and 'moved' to the right-hand panel either all together (with the 'double' arrow) or the one or more selected entries. Their sequence can then be adjusted by moving them up or down.



b. defining the general display style

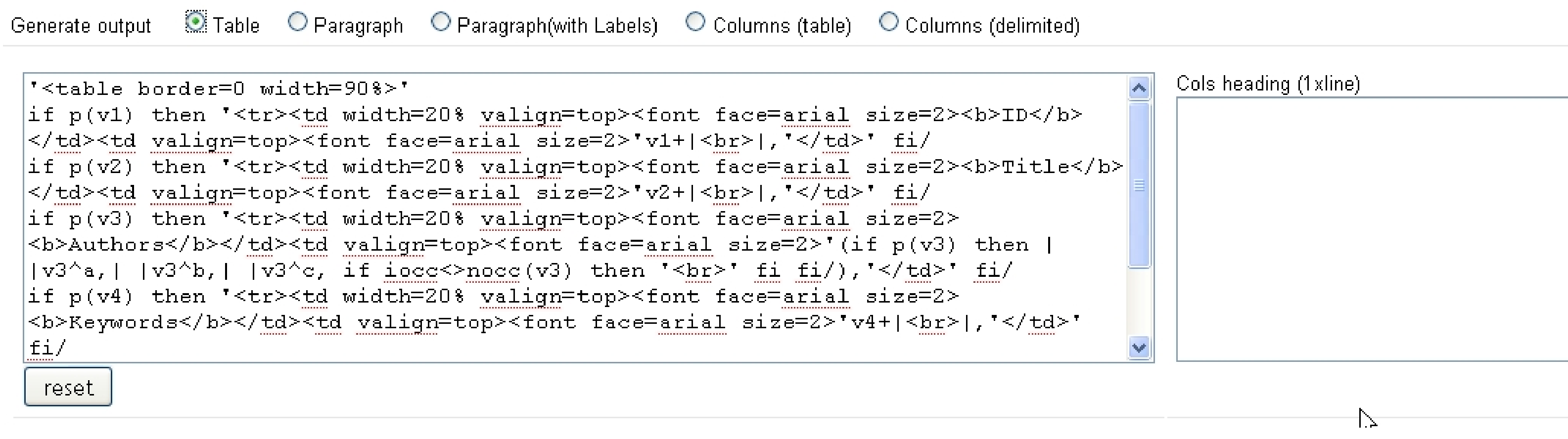
   The ABCD PFT-wizard offers either a general 'table-based' display (with field names and values in separate columns) or 'paragraph' (no columns) display, with or without field names. Immediately ABCD will generate the full PFT needed to display the fields selected in step 1.

   A special option here is to output the selected fields as 'delimited' data (separated by the ISIS-default pipe separator '|') for re-use in other software such as spreadsheets or statistical tools.

   When columns have been produced, the headers for the columns can be specified by putting them one by one each time in a new line in the dedicated box.

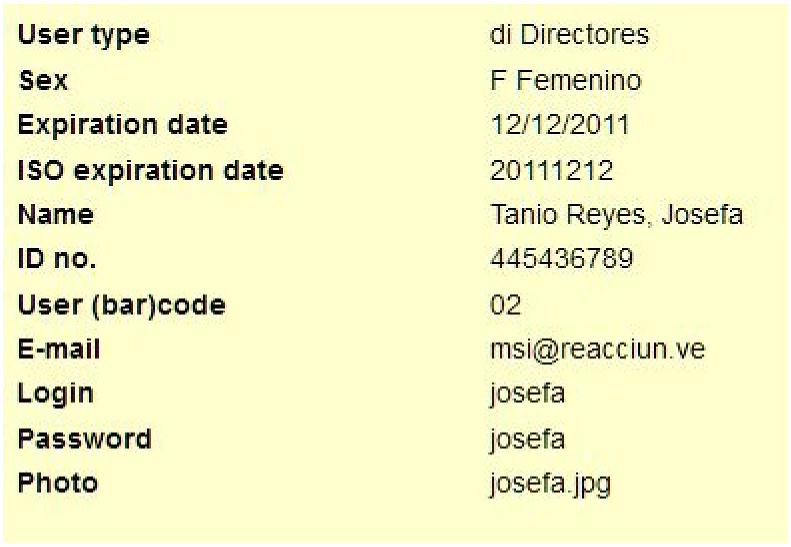
c. (optional) Generate output : this is mostly for testing purposes, where one can send the output produced by the PFT to either a word-processor (as default in the operating system), a spreadsheet software, to the screen itself ('preview') and as a text-file without formatting.

   As mentioned and shown above, generating output to test the PFT can be one of three pre-defined standard ways of presenting data from your database : either a 'table-formatted' web-page (in columns) or 'paragraph-formatted' webpage (no columns), or - alternatively for quite different purposes - a delimited format for export to other software. ABCD will immediately generate the necessary code, combining HTML-tags as quoted 'literals' with values from the fields (Vx).


Generate output
[✓ Table] Paragraph Paragraph(with Labels) Columns (table) Columns (delimited)
Cols heading (1xline)
reset

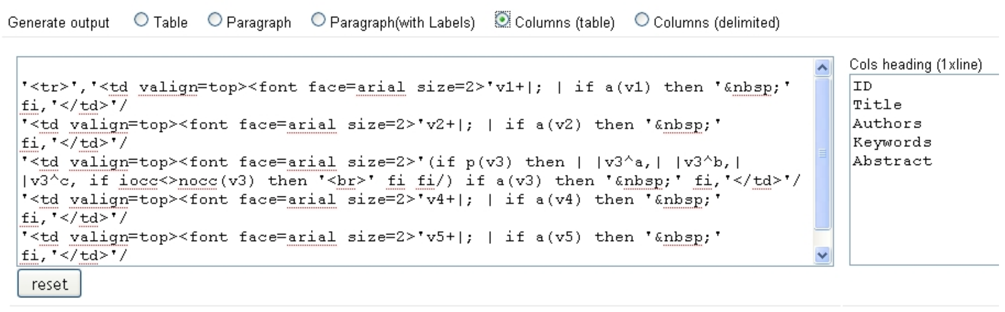
```
'<table border=0 width=90%>'
if p(v1) then '<tr><td width=20% valign=top><font face=arial size=2><b>ID</b>
</td><td valign=top><font face=arial size=2>'v1+|<br>|,'</td>' fi/
if p(v2) then '<tr><td width=20% valign=top><font face=arial size=2><b>Title</b>
</td><td valign=top><font face=arial size=2>'v2+|<br>|,'</td>' fi/
if p(v3) then '<tr><td width=20% valign=top><font face=arial size=2>
<b>Authors</b></td><td valign=top><font face=arial size=2>'(if p(v3) then |
|v3^a,| |v3^b,| |v3^c, if iocc<>nocc(v3) then '<br>' fi fi/),'</td>' fi/
if p(v4) then '<tr><td width=20% valign=top><font face=arial size=2>
<b>Keywords</b></td><td valign=top><font face=arial size=2>'v4+|<br>|,'</td>'
fi/
```

The resulting output looks as follows :



User type	di Directores
Sex	F Femenino
Expiration date	12/12/2011
ISO expiration date	20111212
Name	Tanio Reyes, Josefa
ID no.	445436789
User (bar)code	02
E-mail	msi@reacciun.ve
Login	josefa
Password	josefa
Photo	josefa.jpg

In case a 'column'-based format ('delimited' will allow export to softwares such as spreadsheets and statistical analysis tools) is selected, in the right square the (sequence of the) headers of the columns can be defined, by default they will be the field-names already defined.



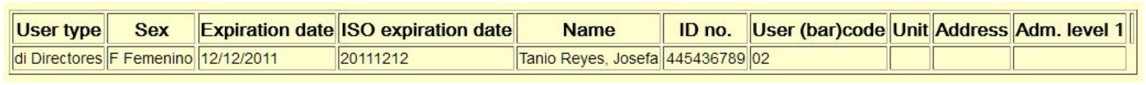
Generate output ( ) Table ( ) Paragraph ( ) Paragraph(with Labels) [✓ Columns (table)] ( ) Columns (delimited)

```
'<tr>','<td valign=top><font face=arial size=2>'v1+|; | if a(v1) then '&nbsp;'
fi,'</td>'/
'<td valign=top><font face=arial size=2>'v2+|; | if a(v2) then '&nbsp;'
fi,'</td>'/
'<td valign=top><font face=arial size=2>'(if p(v3) then | |v3^a,| |v3^b,|
|v3^c, if iocc<>nocc(v3) then '<br>' fi fi/) if a(v3) then '&nbsp;' fi,'</td>'/
'<td valign=top><font face=arial size=2>'v4+|; | if a(v4) then '&nbsp;'
fi,'</td>'/
'<td valign=top><font face=arial size=2>'v5+|; | if a(v5) then '&nbsp;'
fi,'</td>'/
```

Cols heading (1xline)

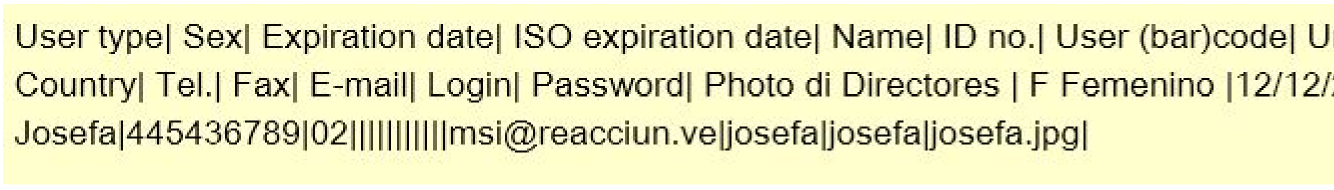
```
ID
Title
Authors
Keywords
Abstract
```

The 'columns with table' format will result, when 'previewed' within the interface as opposed to 'sent to a document or worksheet' (this last option is ideal when outputting data as 'delimited'), into a display format like the following :



User type	Sex	Expiration date	ISO expiration date	Name	ID no.	User (bar)code	Unit	Address	Adm. level 1
di Directores	F Femenino	12/12/2011	20111212	Tanio Reyes, Josefa	445436789	02			

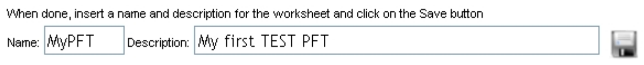
The last display 'style' is for export to 'delimited' formats for use with spreadsheet and statistical software. ABCD will produce the 'pipe' (|) as a delimiter which then could be changed into e.g. semi-colons or other characters not-present into the field values. The output will then look like e.g. :



```
User type| Sex| Expiration date| ISO expiration date| Name| ID no.| User (bar)code| U
Country| Tel.| Fax| E-mail| Login| Password| Photo di Directores | F Femenino |12/12/
Josefa|445436789|02||||||||||msi@reacciun.ve|josefa|josefa|josefa.jpg|
```

d. saving the format under a specific name

   Finally the generated PFT should be saved with a real file-name (default is the database-name) and a 'description', which is the name of the format as displayed.



When done, insert a name and description for the worksheet and click on the Save button

Name:	MyPFT	Description:	My first TEST PFT

Note

Remember that PFT's are saved separately for each language in the 'pfts' subfolder of the database-folder, as a text-file with extension '.PFT'

2.2.1.3. The actual process of creating a database

After having discussed the principles of ISIS database structures and the way ABCD implemented them, we can now proceed with the actual process of creating a database.

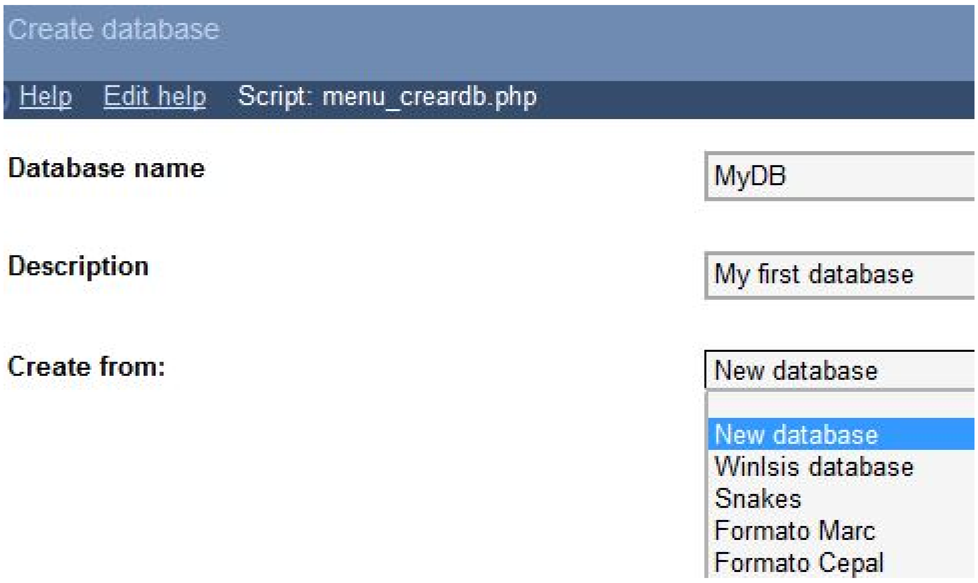
Important

General advice on the creation of a new ABCD-database : since 'validation' (e.g. about the bases-folder to be correctly accessible by ABCD to create new folders and files) is or can only be performed at the end of the database-creation sequence, described above, only after having run through all these steps the software can report on success or not. If no success - in case of an error - all steps have to be repeated all over again, as mostly nothing could be saved. Therefore we strongly recommend to always confine the creation process to a very basic structure, even with only one single field in the FDT, FST and PFT, and then immediately to create the database, check for errors in the listing and after successful creation of the base-structure to continue elaborating it by adding more fields, refining the display formats and indexing etc.

1. From the main Central menu, select the option 'Create database'

to get the initial database-creation screen :

   Figure 2.8. Database creation initial screen

   Create database

   Help Edit help Script: menu_creardb.php

Database name	MyDB
Description	My first database
Create from:	New database

   New database
   Winlsis database
   Snakes
   Formato Marc
   Formato Cepal

   As can be seen from the screenshot, three things have to be defined :

   a. the database name, which is a short name to be used as the internal name of the database; based on this ABCD will create a lot of (sub-)folders in the database main folder and files in there

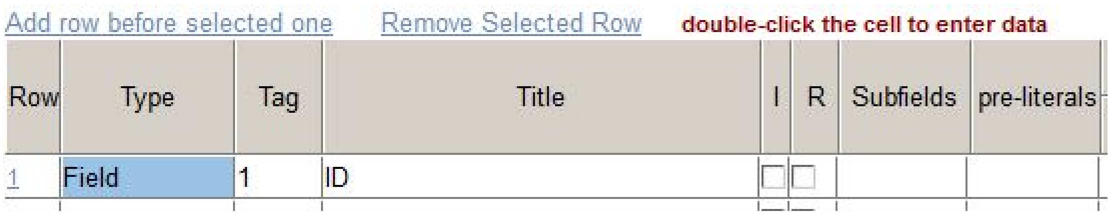
   b. Description : this is the name of the database as will be shown on screen, e.g. in the list of available databases

   c. Create from : this can be either (as selected here) a 'new database', an existing WinISIS database (see infra) or a copy of one of the available ABCD-databases.

2. The initial FDT

   After filling in the three elements above, the next screen is quite complicated as it avails the full FDT-editor, which will be discussed later. Here we suffice to create one simple field in the first row of the table (grid), which itself can be changed later on and complemented by (a lot) more fields later on.

   Figure 2.9. FDT for database creation

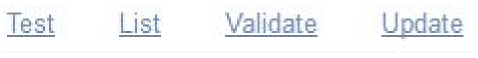
   

   Add row before selected one Remove Selected Row double-click the cell to enter data

Row	Type	Tag	Title	I	R	Subfields	pre-literals
1	Field	1	ID	☐	☐		

   Under the grid a menu (explained later) allows to 'update' the FDT and proceed to the next stage :

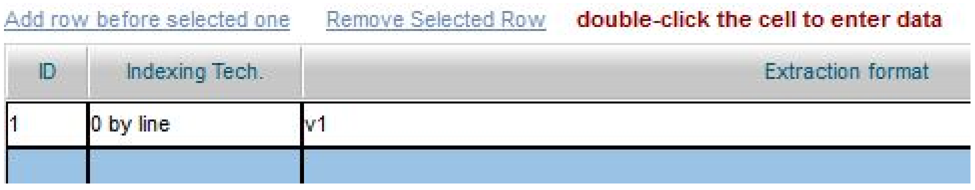
   Figure 2.10. FDT-editor menu

3. The initial FST

   This next step is about creating the indexing definition or 'Field Selection Table'. The FST editor always shows the actual table to the left hand of the screen, with a viewer of the available fields for indexing (taken from the FDt) at the right hand side. Again for the initial creation of a database, it is sufficient to add just one single line, almost like a dummy since the real design of the index can be done at any later stage.

   ## Figure 2.11. FST for database creation

   
   Add row before selected one
   Remove Selected Row
   double-click the cell to enter data

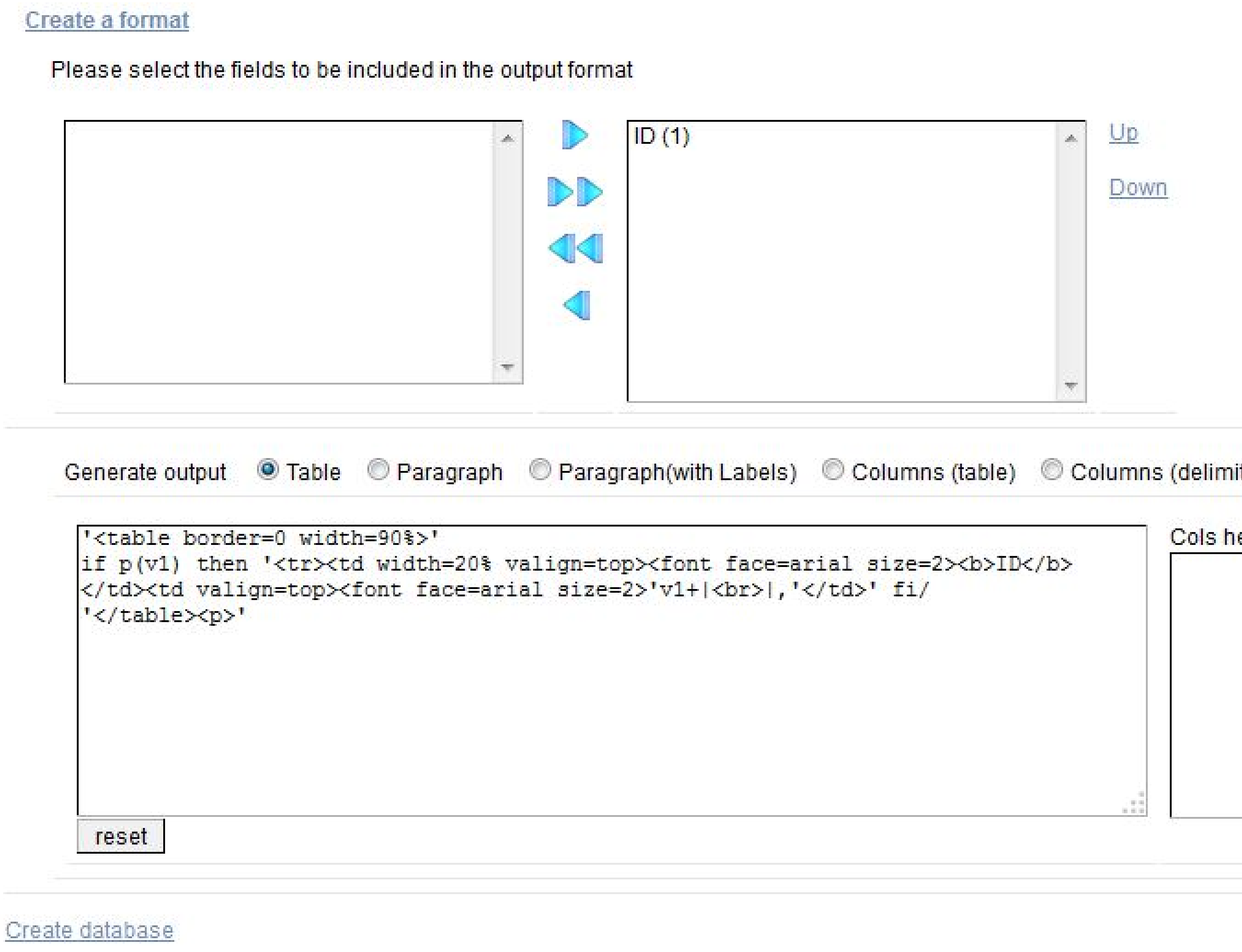
ID	Indexing Tech.	Extraction format
1	0 by line	v1

   These three columns have been explained in more details elsewhere, we keep it as simple as possible here. There is only one closing option here : update the FST to proceed to the last step.

4. The initial PFT

   The final step for database creation involves creating a (again : very simple) PFT, by selecting the one available field in the list, chosing one of the available styles, skipping the 'generate output' (since no testing still is necessary here) part and actually creating the database. This includes saving this PFT as the default PFT, i.e. with the internal name of the database itself.

Figure 2.12. PFT for database creation


Create a format
Please select the fields to be included in the output format
Up
Down
Cols he
reset
Create database

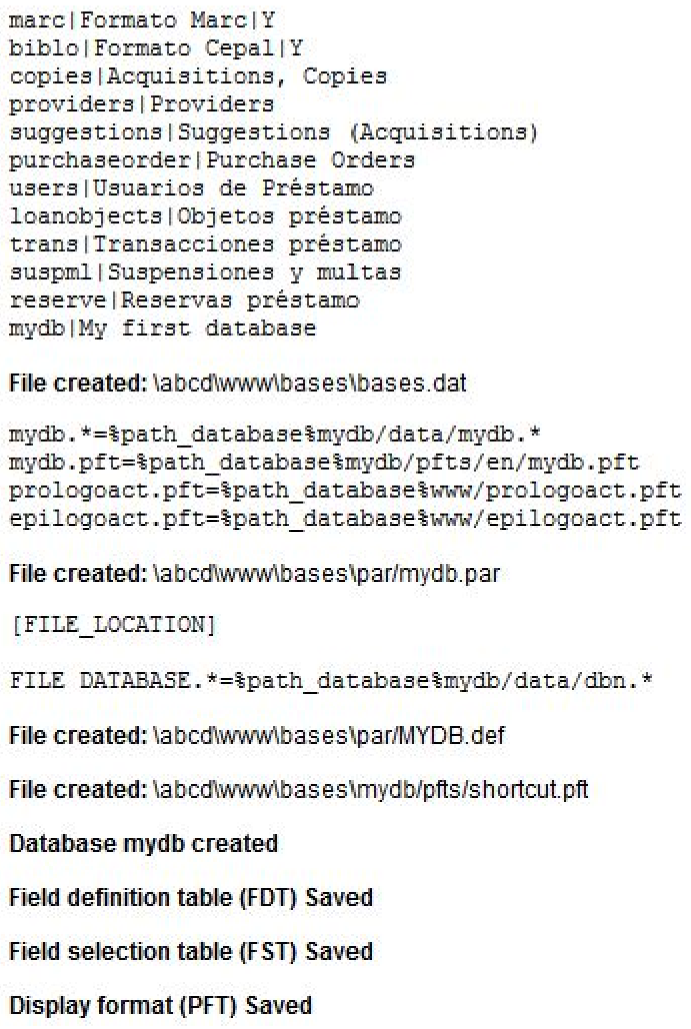
ID (1)

Generate output [✓ Table] Paragraph Paragraph(with Labels) Columns (table) Columns (delimi

```
'<table border=0 width=90%>'
if p(v1) then '<tr><td width=20% valign=top><font face=arial size=2><b>ID</b>
</td><td valign=top><font face=arial size=2>'v1+|<br>|,'</td>' fi/
'</table><p>'
```

As can be seen in the screenshot, the one-and-only available field was moved from the left to the right side to include it into the PFT, the 'Table' laout was chosen and ABCD has created immediately the necessary PFT-code for this layout, which involves mainly the HTML-table formatting in fact. The link at the bottom then results in all folders and files for the new database being written to the harddisk (in the ABCD-bases folder as defined by config.sys of the Central module). A list of files written is shown, if all went well (and we minimized risks...) there won't be errors in red color, just confirmations of the process having been successful and some instructions to proceed, e.g. to make the database also available for non-administrator profiles (otherwise only administrators can use the database). Also the new database has been added at the end of the list of available databases for opening it.

Figure 2.13. Database creation final confirmation



```
marc|Formato Marc|Y
biblo|Formato Cepal|Y
copies|Acquisitions, Copies
providers|Providers
suggestions|Suggestions (Acquisitions)
purchaseorder|Purchase Orders
users|Usuarios de Préstamo
loanobjects|Objetos préstamo
trans|Transacciones préstamo
suspml|Suspensiones y multas
reserve|Reservas préstamo
mydb|My first database
```

File created: \abcd\www\bases\bases.dat

```
mydb.*=%path_database%mydb/data/mydb.*
mydb.pft=%path_database%mydb/pfts/en/mydb.pft
prologoact.pft=%path_database%www/prologoact.pft
epilogoact.pft=%path_database%www/epilogoact.pft
```

File created: \abcd\www\bases\par/mydb.par

```
[FILE_LOCATION]

FILE DATABASE.*=%path_database%mydb/data/dbn.*
```

File created: \abcd\www\bases\par/MYDB.def

File created: \abcd\www\bases\mydb/pfts/shortcut.pft

Database mydb created

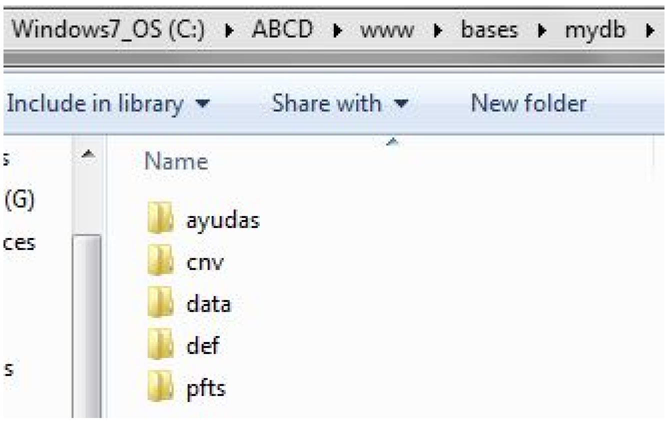
Field definition table (FDT) Saved

Field selection table (FST) Saved

Display format (PFT) Saved

The new database has now been created, i.e. in the bases-folder a new folder (in this case with name 'mydb') with subfolders and all necessary files have been written.

Figure 2.14. New database folder created



Note

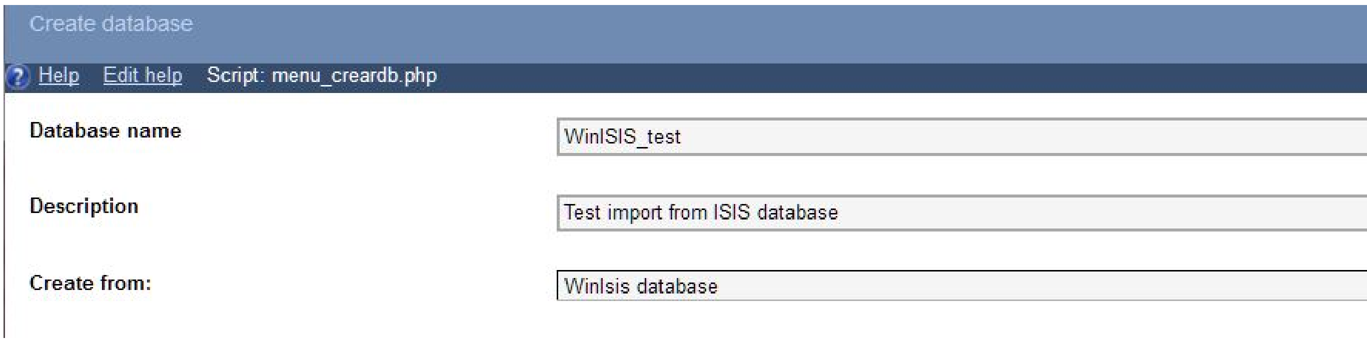
New from v1.2t on : this database creation process also creates a new folder in the htdocs/bases folder to make sure there is a writable folder for uploading files related to this database. Using the new mechanism for uploads (see infra) however this folder can be moved to other locations.

2.2.2. Copying an existing WinISIS database

For creating an ABCD-database from your existing WinISIS database, here are the steps to be followed :

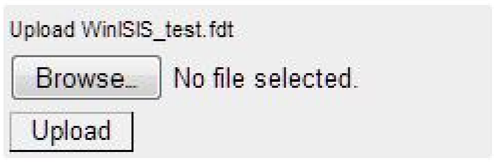
1. Export your existing records into an ISO-export file using WinISIS (or another ISIS-tool allowing ISO-export); remember where you have saved this .ISO export file, normally it will reside in the WORK-folder of your WinISIS installation. No specific parameters need to be set, unless of course you would only want to use a subset of the records in that database (by using MFN-range minimum and maximum or a search result) or you need to 'convert' (reformat) the records before entering them into ABCD by the use of a 'reformatting FST'.

2. Assign in ABCD, after having selected the 'Import from WinISIS' option, a name and a description - as with a new database, see supra. Then select your WinISIS database using the list in the 'Create from' part of the dialog.

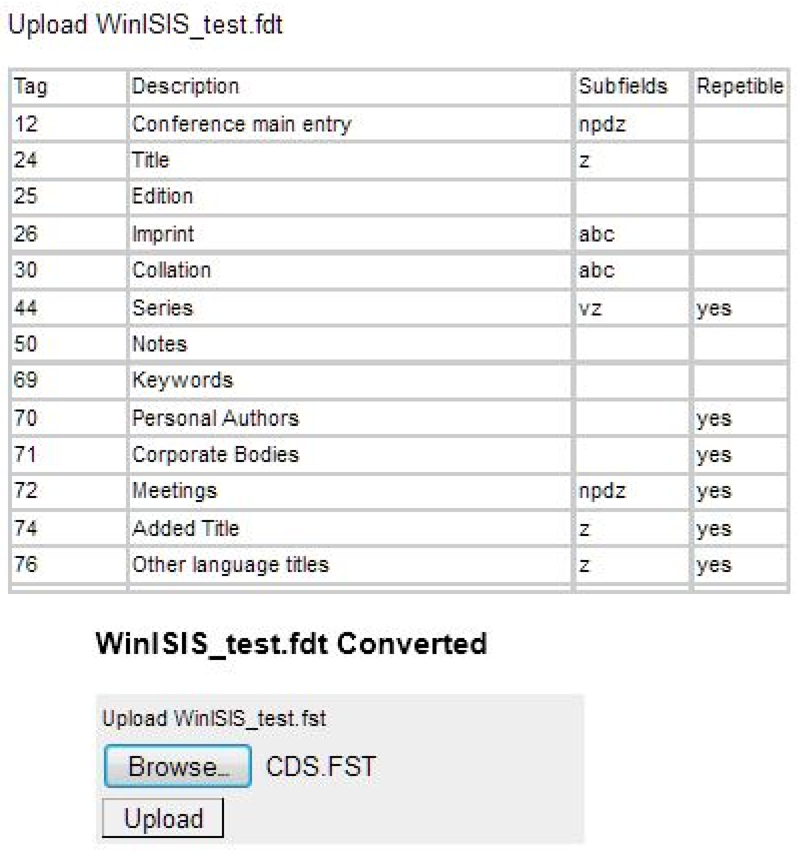


Database name	WinISIS_test
Description	Test import from ISIS database
Create from:	WinIsis database

3. Select the FDT belonging to that database and click on 'Upload' in order to have the FDT loaded into the ABCD environment of the new database.



4. Select the FST belonging to that database and click on 'Upload' in order to have the FST loaded into the ABCD environment of the new database.

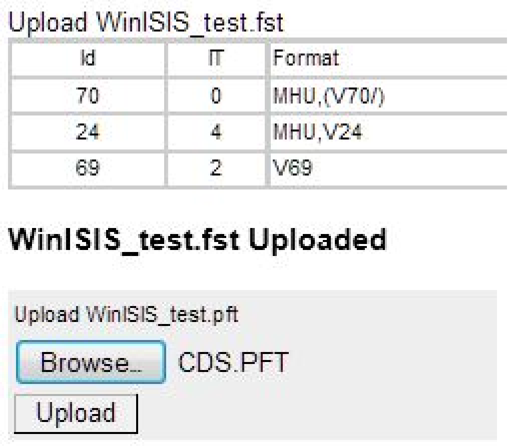

Upload WinISIS_test.fdt

Tag	Description	Subfields	Repetible
12	Conference main entry	npdz	
24	Title	z	
25	Edition		
26	Imprint	abc	
30	Collation	abc	
44	Series	vz	yes
50	Notes		
69	Keywords		
70	Personal Authors		yes
71	Corporate Bodies		yes
72	Meetings	npdz	yes
74	Added Title	z	yes
76	Other language titles	z	yes

WinISIS_test.fdt Converted

Upload WinISIS_test.fst

5. Select the PFT belonging to that database and click on 'Upload' in order to have the PFT loaded into the ABCD environment of the new database.


Upload WinISIS_test.fst

Id	IT	Format
70	0	MHU,(V70/)
24	4	MHU,V24
69	2	V69

WinISIS_test.fst Uploaded

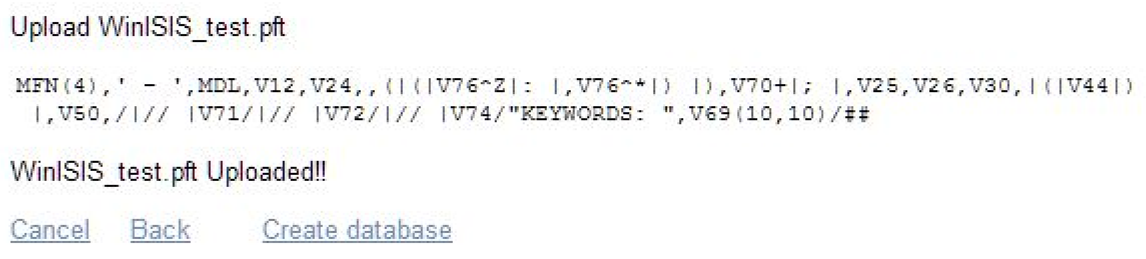
Upload WinISIS_test.pft

Warning

Most WinISIS databases use a default PFT (with the name of the database) which contains typical Windows (as opposed to HTML) codes, such as e.g. 'BOX', 'FS' etc. These will result in a 'grammatical' error when later opening this in ABCD, so it is better to avoid this by selecting a PFT without such typical Windows-elements ! If not available, remember to re-create a HTML-based format within ABCD to replace the default PFT for your new database.

6. Click on the 'Create Database' option in order for ABCD to start writing the necessary folders and files for your new database. A message about successful creation (or not, in case of problems) will be displayed on your

screen. Also you will be reminded of the fact that without assigning this database as accessible to at least one user, you won't be able to use this database.



Upload WinISIS_test.pft

```
MFN(4),' - ',MDL,V12,V24,,(|(|V76^Z|: |,V76^*|) |),V70+|; |,V25,V26,V30,|(|V44|)
 |,V50,/|// |V71/|// |V72/|// |V74/"KEYWORDS: ",V69(10,10)/##
```

WinISIS_test.pft Uploaded!!

7. Now you can open the new database, as it has become part of the list, in the main database management window.

8. As the database can be opened but with 0 records, the first thing to do is to import the ISO-records created in the first step of this series. To this end, click on the 'Utils' icon in the main toolbar of this data-entry screen (as described in the section dedicated to this) and select the option 'ISO import'. This procedure further, as can be expected, involves the selection of the source ISO-file, which then should be 'uploaded'.

9. Now, a bit strange, the ISO-file is ready for being effectively imported into the database. For this, click on the 'Utils' icon again in the toolbar and select 'ISO-import', where now the uploaded ISO-file is available for effectively importing (converting) into your ABCD-database. The software will now ask you if it is o.k. to indeed start importing the ISO-records from the selected file. The list of imported records will be shown on your screen to monitor its progress and success.

10. If your newly imported records don't immediately show up in the database, re-open the database from the main menu, this will refresh the database parameters.

11. Now the records should be visible and editable as normal records, only they have not yet been indexed into the Inverted File, so use the option 'Inverted File' update in the 'Other utils' section of the 'Utils' screen.

Warning

If your imported series of records is quite large (e.g. above a few hundreds), it is possible, depending on the system you are working on, that the process will be too long for the web-server (Apache in most cases) and it won't be allowed to finish. For this reason it will be necessary in such cases to perform the Inverted File generation action not from ABCD (as a web-environment) but directly from the command-line, using the dedicated CISIS-tools (for which another section of this manual will give the details).

For the somehow more cumbersome procedure of importing ISO-records, mostly by batches because of the slow processing through the http-protocol and PHP-timeouts, there will be an alternative option in version 2.0 which creates a call to the Operating System and loads the ISO-records in a much faster one-run command using CISIS directly.

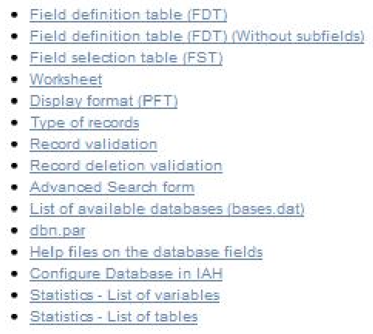
2.2.3. Copying an existing ABCD database

This last option is the easiest to perform, as only a new name and description need to be entered, after which ABCD will simply re-produce all necessary files into a new folder structure for the new database. The source databases from which you can choose are the ones listed in the database-menu, in other words : the database descriptions listed in the file 'bases.dat' in the ABCD bases-folder.

The system will simply - as above - list all files copied and created in their proper folder-structure and that is it ! An empty database, as a copy of the existing ABCD-database but with new name and description, will be available for normal use.

2.3. Update database definitions

From this option it is possible to edit all existing 'structures' or definition tables related to a database in ABCD. In comparison with 'normal' ISIS-databases, and in order to support some more advanced features in ABCD, there are some more of such database definition elements, as can be seen from the following 'database definitions' menu :



- Field definition table (FDT)
- Field definition table (FDT) (Without subfields)
- Field selection table (FST)
- Worksheet
- Display format (PFT)
- Type of records
- Record validation
- Record deletion validation
- Advanced Search form
- List of available databases (bases.dat)
- dbn.par
- Help files on the database fields
- Configure Database in IAH
- Statistics - List of variables
- Statistics - List of tables

In fact only the first four tables are used in other ISIS-environments : the Field Definition Table (FDT), the Field Select Table (FST), the FMT or edit-worksheet and the Print Format Table or PFT. Since we needed these also in order to create a 'new' database in ABCD, they were discussed in the according section above, the only difference being that instead of an empty table a pre-filled table with the already existing definitions will be presented by ABCD.

Let us deal with the definitions for ABCD-databases now in more details.

2.3.1. Field definitions and worksheets

Earlier on in this manual, we already dealt with the basic elements of the FDT to define a 'repository' of possibly-used fields in the ABCD-database. We also noted that the definition of the worksheets is closely linked to the FDT (unlike in other ISIS applications) and defined fields can be presented in different ways from the default one as given in the FDT by using additional worksheets.

The FDT-editor screen is probably the most complicated one of ABCD, as it presents not only the FDT proper, but also defines the worksheet for data-entry (or cataloging), unlike in other ISIS-softwares where a separate but simple 'FMT' (data entry worksheet) is defined, and since in addition ABCD uses quite some more advanced data-entry features such as picklists and validations, this step is rather demanding.

Since version 1.0 of ABCD two interfaces are provided for editing the FDT : one 'full' and one (marked in red for the time being) 'abbreviated'. The abbreviated form will not show the subfields unless the field itself is selected by its link in the first column - they will then appear in the subsequent form where all the details of the subfields can be edited, e.g. the width of the columns as 'no. of columns' in case the subfields are to be presented in a table rather than separate entry-fields (which is the default type of entry : Text/Textarea). This abbreviated FDT-editor is quite practical - and faster - in case of large complicated (i.e. using many subfields) structures such as MARC. For other, simpler structures the full FDT-editor can be used.

In the case the full FDT-editor is selected, or when in the abbreviated editor a field is presented in a detailed format, the link at the first column can be used to show the field in a 'vertical' detailed way, so presenting just the selected field in a normal form. This then again is more practical to deal with the individual elements to be defined.

Tip

In order to 'edit' the form, double-click inside a cell of the table ! Simple-clicking will only select the row but not make the cell editable or invoke the menu attached to the cell.

We will now discuss some more advanced possibilities to be used when designing worksheets to illustrate the power of ABCD. Remember ABCD can be used for almost any type of ('mostly textual') information and is not confined to bibliographical or library records. In fact at the occasions of ABCD-workshops and courses we have seen students creating information management systems on the Olympic Games, their family genealogy, personal interest WWW-inventories, personal music collections, pet-collections etc. whereas ISIS always has been used also in quite strict scientific environments for vocabulary control, thesauri, experts and knowledge bases, species databases...

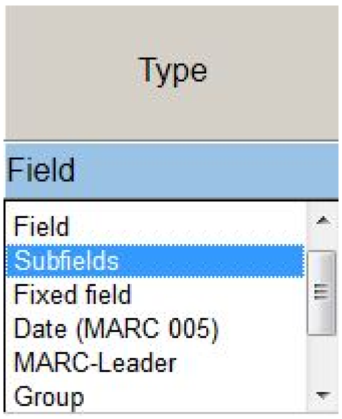
In v1.2t new features, allowing more sophisticated data-entry forms, have been introduced. We will also present them here.

2.3.1.1. field types

In v1.2t the list of possible field types has been simplified and options have been shifted to the input type column. E.g. the 'auto-increment' field type is now a normal field but with 'auto-increment' as its secondary characteristic in the 'input types' column.

Except for two simple 'separator' options to sub-divide the FDT presentations (line and header), at the bottom of the list (not shown) we have the following field types :

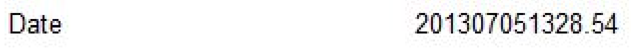
Figure 2.15. Field types



- Field : any field which is not a special MARC-type field, fixed, a date or a field-with-subfields ('group') is a normal field. So this will be used for most fields in reality.

- Subfields : whenever a 'group' has been declared, in the subsequents rows of the FDT the subfields of that group need to be defined and declared to be of the 'subfield' type

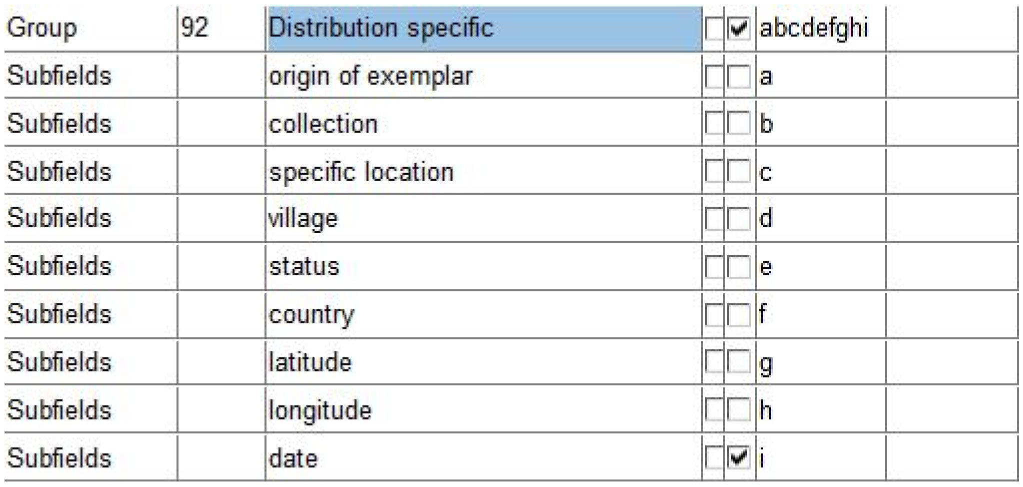
- Fixed field : this is a field with a fixed structure which can be presented with its structural parts separated; the MARC field 008 is a typical example, e.g. the first 6 positions need to contain the date (yymmdd), at specific positions near the end the language code has to be given etc.

- Date : a special date-field for field 5 of MARC can be used with this type; it will be automatically present in the worksheet with a detailed date-and-time stamp format :
  
  Date 201307051328.54

- MARC-Leader : this is the special MARC field meant to contain specific information on the record (see the MARC manuals)

- Group : a field subdivided in subfields. This is a very important 'structured field' since it can contain a lot of separate information units grouped together in one field : it is like a 'record in a record'. Simple examples are e.g. author names (with substructures like name, firstname, role, date of birth, date of death...) or addresses (building/housenumber, street, municipality, ZIP-code, country, phone/fax-numbers, e-mail addresses...), but the more complicated example we will be using here comes from a real-world scientific database to manage information published on a certain animal species.

Figure 2.16. Group field example



Group	92	Distribution specific	☐	✓	abcdefghi	
Subfields		origin of exemplar	☐	☐	a	
Subfields		collection	☐	☐	b	
Subfields		specific location	☐	☐	c	
Subfields		village	☐	☐	d	
Subfields		status	☐	☐	e	
Subfields		country	☐	☐	f	
Subfields		latitude	☐	☐	g	
Subfields		longitude	☐	☐	h	
Subfields		date	☐	✓	i	

As can be seen here, this field will contain details about the distribution of the species described in the literature to keep track of exactly where the studied exemplar was found. The interesting part follows later when for each of these subfields different 'data-entry input types and parameters will be defined.

Since the next FDT or FMT columns (tag, title, browsing Identifier, Repeatable, subfields and pre-literals) don't need further explanation on top of what is already discussed earlier on, we skip to the next interesting column for further detailed explanations.

Note

ABCD, unlike WinISIS and other ISIS-variants, allows creation of FDT for each language used, so field titles can be language-dependent !

At the end of the table options are provided to save the table, but also to test and validate it

Test List Validate Update

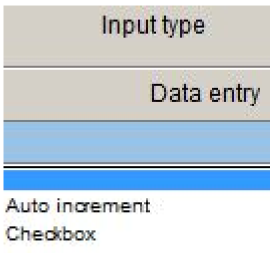
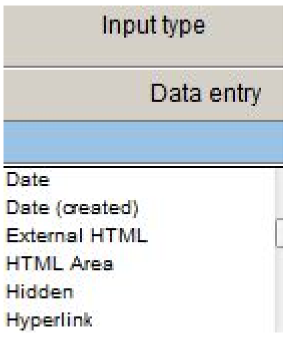
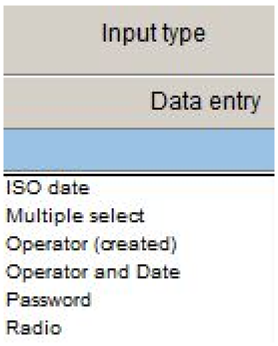
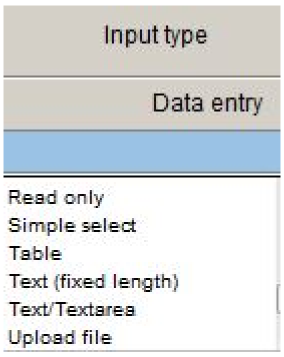
Here the 'Test' and 'Validate' options will resp. display the resulting form for getting an idea about the result - interestingly this is an 'interactive' display so it can be really tested ! - and display the table in a different window with a message indicating wether any logical or grammatical errors are present in the table. It goes without saying that such errors need to be corrected first before 'saving' or 'updating' the FDT with the last option presented here.

The 'List' option provides a listing of the table in a separate window, e.g. allowing printing it or saving it as a separate file.

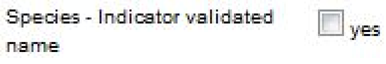
2.3.1.2. Input types

In v1.2t the following input types, which are mostly HTML-form elements but also some extra special ABCD-types (to which in v2.0 will be added for the digital library feature : DOC and URL), are available (in the list the options are alphabetically sorted) :

Table 2.1. ABCD input types

Input types	Input types continued	Input types continued	Input types continued
 Input type Data entry Auto increment Checkbox	 Input type Data entry Date Date (created) External HTML HTML Area Hidden Hyperlink	 Input type Data entry ISO date Multiple select Operator (created) Operator and Date Password Radio	 Input type Data entry Read only Simple select Table Text (fixed length) Text/Textarea Upload file

First we briefly mention the 'standard' HTML form elements which don't need much additional explanation as they are well-known or simple to understand.

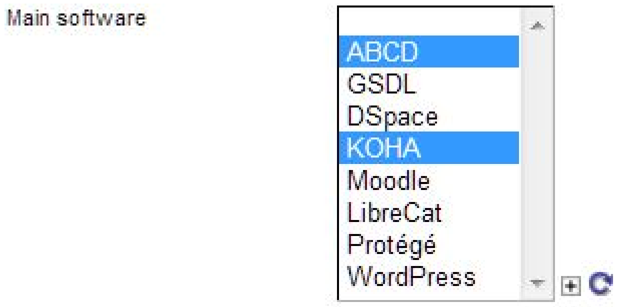
- checkbox : a simple small box to 'tick' in order to make this active (1) or not (0), e.g. to denote whether a record is active (1) or not (0) in the database 
Species - Indicator validated name ☐ yes

- radio : several round buttons of which only one can be selected as alternating possibilities

- hyperlink : the value entered here should be a URL as it will be formatted as a hyperlink

- simple select : a list of options as in a menu, of which only one can be selected

- multiple select : a list of options as in a menu, of which one or more can be selected (with shift-click and ctrl-click for selecting subsequent or not more entries); this is probably one of the most handy ways of facilitating data-entry if the list of options can be limited to just a few from which more than one can be selected, e.g. :

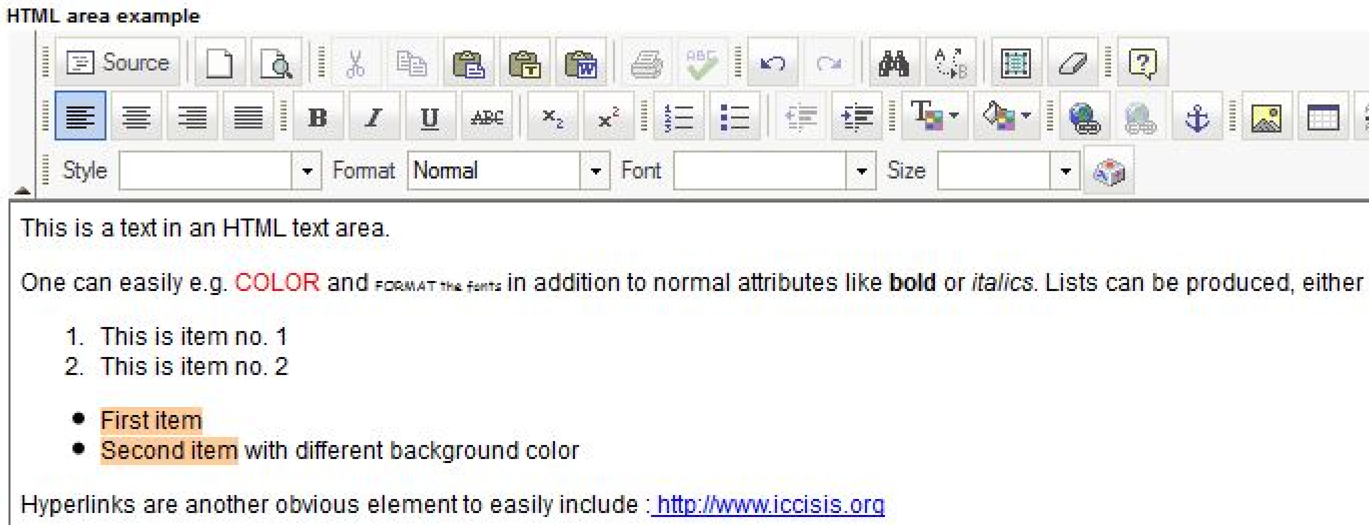

Main software
ABCD
GSDL
DSpace
KOHA
Moodle
LibreCat
Protégé
WordPress

- password : typed characters will be hidden by asterixes

- text area : this is the standard, default one-line empty box in which text can be typed; width and height can be defined/changed by specifying the number of rows and positions resp. in the columns 'rows' and 'cols' in the FDT

In addition to these more or less standard elements, ABCD is adding the following input types for specific purposes :

- HTML Area : a larger editing window with accompanying HTML-editor toolbox will be shown to allow editing of full HTML-encoded documents :


HTML area example

This is a text in an HTML text area.

One can easily e.g. COLOR and FORMAT the fonts in addition to normal attributes like **bold** or *italics*. Lists can be produced, either

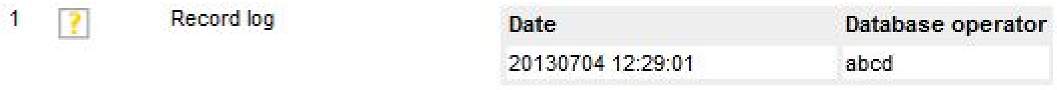
1. This is item no. 1
2. This is item no. 2

- First item
- Second item with different background color

Hyperlinks are another obvious element to easily include : http://www.iccisis.org

Needless to say this HTML-editing control (loaned from the rich PHP-library of controls) adds a lot of power and can make records quite attractive since most HTML-attributes are available here. One can also switch in between the source-code (the HTML-tags) and the 'WYSIWYG' result. For educational purposes this feature even allows ABCD to be used to teach/learn HTML !

- operator : the username of the logged-in operator will be automatically entered in this field
- operator and date : both the operator's username and date of saving the record will be stored, as illustrated here :


1 Record log

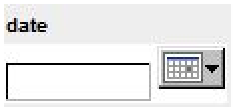
Date	Database operator
20130704 12:29:01	abcd

- read-only : the value given (as default) cannot be edited
- text fixed-length : a text with maximum length as specified in the column 'cols'

Note

If in addition to this max. length another value preceded by a slash is given, ABCD will show a counter for remaining characters, e.g. the value '50/200' in the cols column means that a box of 50 characters width is shown and the maximum number of 200 characters will be counted down.

- date : a field in which the operator can easily put a date using a calendar tool (as shown by the icon) :



- ISO date : a date field entered with the calendar tool but automatically formatted into the ISO format YYYYMMDD :

ISO date 20130705

- date (created) : a field in which the software will automatically add the current date/time-stamp, e.g. :

Date created 20130705 02:08:18

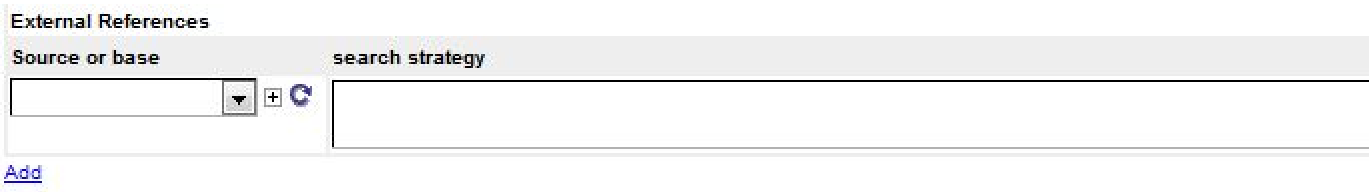
- upload file : ABCD will present next to the input box an icon to open a browser to folders from where a file can be selected to be uploaded into the server's domain and a link will be generated to open the related file :

- auto-increment : the software will assign the next available numerical identifier automatically, based on the last stored value in the file 'control_number.cn' (in the data-subfolder of the database); overruling this mechanism is allowed by clicking on the 'assign'-link, which then means the operator takes full control but also responsibility about the number being and remaining unique for that database (!)
- hidden : the field can be entered but won't be shown
- table : ABCD will present input elements as cells of a table, each with their own attributes, making possible some quite sophisticated input worksheets. This needs some more explanation, given immediately here.

Note

This 'table' is quite different from the 'table' option in the picklists of the FDT, where a table is just a list of options or a menu in fact.

Let's first give a not-too-complicated example : a table with only two parts (or columns) : we want to take two subfields of a 'group' into one table, one is taken from a menu (indeed : a table picklist inside a table input-type...), the second one is a normal text field :



In this example the 'source or base' subfield is defined as a picklist in the first column of the table, the 'search strategy' subfield is a normal textarea in the 2nd column.

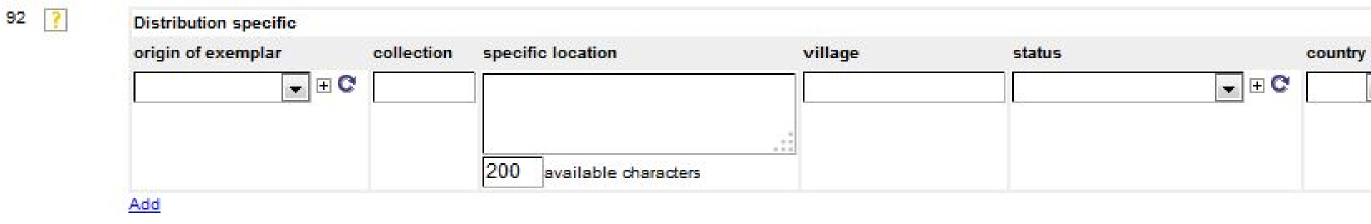
Warning

When using picklists in such a 'table' input, they should be defined as 'select simple' (or single-item selection).

Note two more options (both new from v1.2t on) :

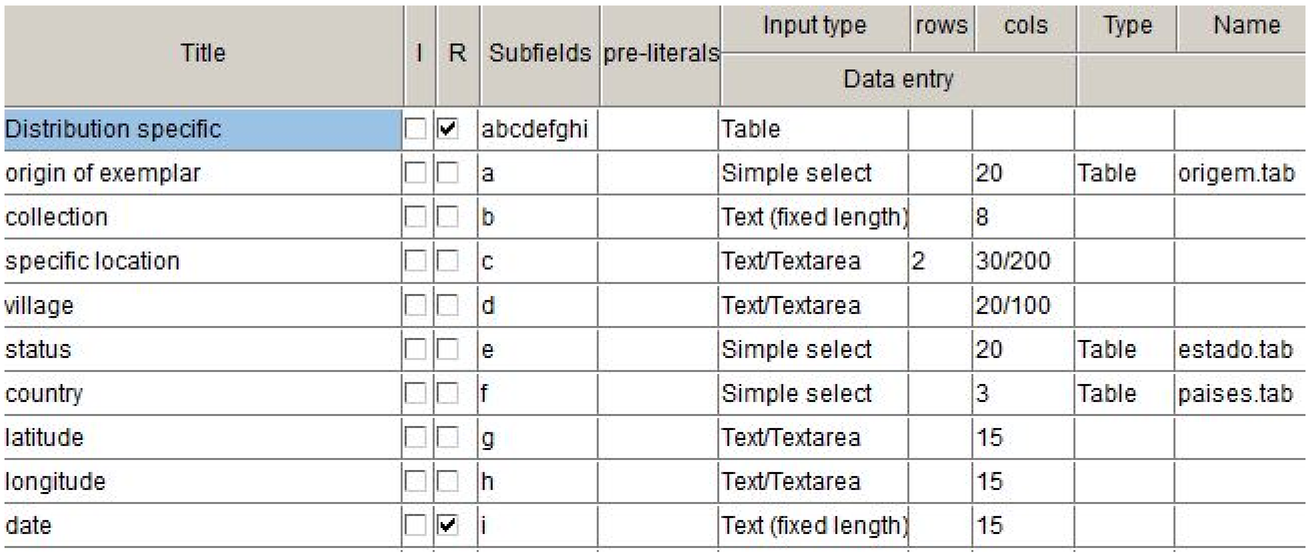
- the 'plus' sign with 'refresh' icon near the picklist : clicking on the plus - which will only be shown to administrator operators - allows the actual list of values for the picklist to be edited on the spot, while 'refreshing' will immediately show the newly edited list in the worksheet
- the 'Add' blue hyperlink will allow to add a new occurrence to this repeatable group, duplicating all elements of the previous row of the table to take new values

Next we show the group used earlier on as an example of a subfielded group with more subfields. Here one additional feature is illustrated, i.e. the use of a fixed field with character counter.



In the third column here ('specific location') a counter is added, counting down from 200 available characters.

This table was defined in the FDT, in the part 'data entry' as shown here:



Title	I	R	Subfields	pre-literals	Input type	rows	cols	Type	Name
					Data entry				
Distribution specific	☐	✓	abcdefghi		Table				
origin of exemplar	☐	☐	a		Simple select		20	Table	origem.tab
collection	☐	☐	b		Text (fixed length)		8		
specific location	☐	☐	c		Text/Textarea	2	30/200		
village	☐	☐	d		Text/Textarea		20/100		
status	☐	☐	e		Simple select		20	Table	estado.tab
country	☐	☐	f		Simple select		3	Table	paises.tab
latitude	☐	☐	g		Text/Textarea		15		
longitude	☐	☐	h		Text/Textarea		15		
date	☐	✓	i		Text (fixed length)		15		

As can be seen, for the third column (specific location) 2 rows were defined, presenting a box of width 30 (raising this number will make it wider) and a counter going down from 200 maximum allowed characters. For other subfields specific tables were created and named here as to provide picklists when editing that column. These picklists can be edited-in-loco by administrators (as shown by the plus-sign) and the newly edited list can be refreshed.

2.3.1.3. Rows and columns

Rows and columns as part of the ABCD-FDT and FMT (worksheet) definitions can have different meanings according to which context they are used in. Generally rows define the ' height' or vertical dimension of the input element, e.g. the number of lines in a box to be shown. The columns define the 'width' or horizontal dimension, e.g. number of characters in a text-box or number of columns in a table (= the number of subfields of a group to show in the table).

As already seen above, the cols-value can be given as x/y, where x is the width value and /200 means creating a counter from 200 down.

2.3.1.4. Picklists definition

The next section in the FDT's or FMT's is the section which defines the use and type of picklists. The following parameters need to be addressed :

- type of picklist, which is either :
  - DB : a database (the same or an external one) from which the dictionary (inverted file or index) will be used to build the picklist
  - thesaurus : a database with specific thesaurus structure from which the picklist will be built
  - table : a simple list of values - one per line - which serve the options of the picklist; this table has to be built manually for the purpose

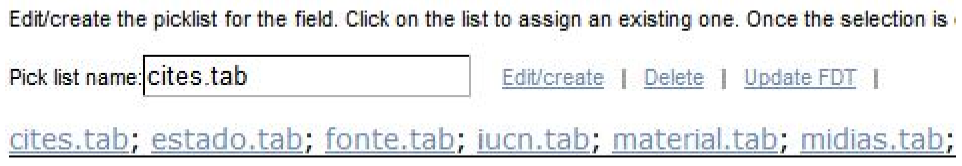
Note

At the time of creation of a database, since the path to the new database is not yet created, no table (which needs to reside inside that path) can be referred to, so this option is not available at the stage of database creation; after creation of the database however one can return to this worksheet element to select and define the table normally

- name of the picklist : name of the DB or thesaurus, or path and name of the file containing the table-values; when using the 'browse' link in this worksheet section to create or edit a table, the path/name will be automatically added. It is good practice to give all picklists the '.tab' extension.

- prefix : in case the picklist is built from an Inverted File (or index), since ABCD prefers to group entries there by prefixes, give the prefix here; e.g. to provide a list of publisher-names which have been included into the index with prefix 'PU_', enter 'PU_' here to only show that part of the dictionary

- browse : this is a link which opens a new window where the picklist table can be created, edited or deleted and



Edit/create the picklist for the field. Click on the list to assign an existing one. Once the selection is

Pick list name: cites.tab   Edit/create | Delete | Update FDT |

saved : cites.tab; estado.tab; fonte.tab; iucn.tab; material.tab; midias.tab;

All ABCD-tables have 2 (or more) columns which are text-values separated by the pipe '| '; the first column is the 'code' to be stored into the record, the second one is the description used to list the code in the table; e.g. to enter a user-category 'academic staff' one could put the following entry in the table (please note that the codes here hare case-sensitive) : `ac|Academic Staff`

After the FDT, FST and FMT for the database has been defined, the last step involves to produce the default display format for the database, i.e. the PFT which will be stored with the name of the database itself. The procedure is the same as what will be used later on for creating additional display formats, i.e. in three steps :

- list as : the (optional) ISIS-PFT which defines how to create the entries in the list, if different from the value-as-is in the dictionary, e.g. for a subfielded field one can extract here only one subfield for listing the entries, or change their sequence etc.

This display format (or 'List as' format) denotes the PFT which defines how the values in the list will be displayed with the Formatting Language. Here either an 'inline' PFT can be given, e.g. a simple one like '`v11`', or a reference to an external format can be given as '`@myformat.pft`'. This external format has to be written following a pre-defined pattern in order to be correctly interpreted.

See the example here used for the authority files of the MARC database : @autoridades.pft:

```
select e3

case 1: v1

case 100: v100^a,`$$$`v100^a

case 110: v110^a,`$$$`v110

case 111: v111^a,`$$$`v111

case 245: v245^a,`$$$`f(mfn,1,0)

case 260: v260^a," : "v260^b

case 270: v270

case 340: v340

...

endsel
```

Note

In case the PFT contains pipes (|) it CAN NOT be put inline into the FDT but has to be put in an external PFT referred to from this cell (this is because the pipes are also used as separators for the column values of the FDT table as stored in ASCII-format).

- the 'list as' PFT can also be used (as from v1.2t) to define an authority list during data-entry : in that case rather than producing the MFN of the record to open (as is the idea of the main index in the A-Z listing), the picklist should produce the value of the field itself to put it into the edit-element from where it is invoked. In this case the PFT should have a format like, with e3 representing the field-tag :

  select e3

  case 12: V12,'$$$' if e4=1 then f(mfn,1,0) else v12 fi

  case 18: v18,'$$$' if e4=1 then f(mfn,1,0) else v18 fi

  ... (add more fields if necessary)

  endsel

  The 'e4' is defined in the section 'LISTA_AUTORIDADES' of the script dataentry/wxis/ifp.xis : `if p(v2030) then e4:=1 fi`

  Remember the part coming after the $$$ is in fact the 'extract as' part, so if e4 is '1' it will result in the MFN of the record to open, otherwise in the value of the field to put into the edit-element. If this statement is lacking instead the MFN will be sent to the edit-element, which is mostly not the aim or idea.

- extract as : the (optional) ISIS-PFT which will re-format the selected entry before actually storing it into the field, e.g. if a name was 'listed as' `Name, Firstname` one can still store the value ^bFirstname^aName by putting here `'^b'v1^b,'^a'v1^a`

  The format defines, again with the Formatting Language, how the contents of the field needs to be exactly extracted from the field values in the record to which the entry in the list (as an Inverted File posting) points. If this value is omitted, the values will be kept in the format defined as 'list as format' in the previous column. If the display format is a pre-defined format (@xxxx) and follows the instruction to separate the display format from the extraction format by $$$, this part should be left empty.

2.3.1.5. Additional elements for fields in FDT and FMT

At the far right side of the grids for FDT and FMT we find some remaining additional elements, which are easy to understand and are just briefly mentioned here.

- default value : a text which will be available automatically when opening the worksheet for that field
- help : a tick-box indicating whether or not a help-file is available for this field which the operator can open from the worksheet; the help-pages are stored in the folder bases/*dbn*/ayudas, where *dbn* represents the name of the database.
- help URL : the link to the actual help-text (as HTML-page) for this field
- link FDT : this new element (as from version 1.2t) means that, if ticked, changes for this field in the FDT (except deleted/added fields) will be also adjusted in the worksheets to remain consistent; there are many cases however where one doesn't want this, e.g. to create alternative worksheets where a field is presented in a different way from the default one as defined in the FDT. E.g. a 'read-only' or even hidden field could be included into a special worksheet with normal editable properties.

2.3.2. Indexing mechanisms

Indexing mechanisms in ABCD are defined through the ISIS Field Selection Table, discussed above. Such index definitions can vary from very straightforward instructions like 'just put the field in the index' (with a simple statement like 'v1') to quite complicated instructions, typical for ISIS applications, which however follow closely librarian's logics, like e.g. 'if the record is about a book, then do this, if it is instead an article, do that'. Such instructions can be written in logical routing instructions which are close to normal human language - mostly preferred by non-technical librarians with statements like 'IF (condition x) THEN (do this) ELSE (do that) FI - or if preferred, more advanced programming logics like 'check a value over a list of possibilities and perform the action listed next to that possibility if there is a match', which is indeed the logics of a 'SELECT CASE.... ENDCASE' construction. Even real programming constructs with e.g. a 'local counter variable' can be used in 'WHILE (i>=0) {....}' loops.

Let's deal with some specific indexing tricks here, bypassing the 'normal' formats for extracting (simple) values to be indexed.

As the Formatting Language (ISIS' most powerful feature) can be used here in its full power (without the graphical presentation features, as this is not for display but indexing purposes), values can be processed before they enter the dictionary, e.g. 'N:', f(mfn,1,0) produces the recordnumber or MFN, formatted (f) as a string, but more complicated examples can be given, e.g. .

- a combination of several fields or subfields with added punctuation
- formats using the REF-function to refer to external databases to take values from there after locating the MFN with the L-function - doing so e.g. codes can be converted into values for the dictionary (see the example above)

  After having edited the FST, it can be tested with any record of your database to check whether the actual values which will be indexed indeed comply with what was intended.

-

2.3.2.1. Indexing substituting codes for values

For consistency or multi-linguality reasons sometimes we prefer to store codes into the databases, rather than the more user-friendly textual equivalents of these. Using a 'referring' technique of ISIS however we can actually put the equivalents in the index vocabulary for easier searching, either taken from the same or from another database. In this last case ABCD behaves a bit, like ISIS can do, like a 'run-time' relational system : taking values from another 'table' (or in our case 'database' which acts as a table).

E.g. suppose we want to index the month of some field-value (given in two positions, in this example as substring of 2 positions starting on position 4 of v65) in a more user-friendly way as (abbreviated) month-name, one could use the following instructions :

```
select s(v65*4.2)

case '01': ,'Jan',

case '02': ,'Feb',

case '03': ,'Mar',

case '04': ,'Apr',

case '05': ,'May',

case '06': ,'June',

case '07': ,'July',

case '08': ,'Aug',

case '09': ,'Sept',
```

case '10': ,'Oct',

case '11': ,'Nov',

case '12': ,'Dec',

endsel

Another example illustrates the use of 'referencing' values from another database, using the combination of the REF() function with the L() function. This function in ISIS 'REF'ers to another named database and takes values, with a PFT, from records there. In order to know which record to open for extraction values from, the L() function 'L'ooks up the MFN of a search key, which is the first hit in the Inverted File linked to the 'search key' passed with the function. An example :

Suppose our database uses sections based on the main category of a classification (Dewey, LC, UDC...) for each record. If that classification code is taken from the first 3 positions of e.g. v80^a, that would mean : `v80^a*0.3` . This is what we have in the database, but we want users to be allowed to search for e.g. 'forestry', with code 630. This can be reached by putting the following statement in your FST, supposing you have a list of such main codes (in v1) and their 'translations' in subjects (in v2) in an ABCD-database, let's say 'CLASSCODES :

```
REF(['CLASSCODES'] L['CLASSCODES'](v80^a*0.3), v2)
```

as this instruction would tell ABCD to extract v2 (the translation) of the MFN found when 'L'ooking up the first three characters of v80 (the classification code) in the 'REF'erred database CLASSCODES. Please check the exact grammar carefully as it is not obvious : e.g.the first time we refer to the database 'CLASSCODES', the second time to the Inverted File with that name. Both names need to be quoted inside square brackets. Now users can search for the search key 'forestry' even in your database there are only entries with the value '630' as the first 3 characters of a (sub-)field...

With such instructions in the 3rd 'extraction format' column of the FST, the index will contain month names, making them searchable by name, or main classification categories worded as a subject. This way there are hundreds of possibilities to include sophisticated mechanisms in your search options : after all what we do in preparing the FST is to prepare options for the users searching our databases.

2.3.2.2. Indexing numerical values from text fields and changing upper-case translation

Indexing techniques 4 or 8 (without or with prefixes) deal with 'full-text' indexing, i.e. each word is taken as such and put into the dictionary. A word is defined as a chain of alphabetical characters, so strings like 2013 (a year) or '1st conference' won't be indexed as expected. Here is how to solve this issue.

In ISIS whatever is to be considered as 'alphabetical' values is defined in a table 'ISISAC.TAB' (AC=alphabetical characters). In the default ISISAC.TAB (stored in ABCD in the main bases folder, but as always references in .par - including now also SYSPAR.PAR - can re-define the path), numerical ASCII-values such as dates (years) are not included, so they are not recognized as 'words' to be indexed. If one needs to do so however this ISISAC-table needs to be edited. Such editing is NOT an ABCD interface function and needs to be done within the OS itself.

Here is the list of alphabetical characters with added (in front) numerical values 048-057 (the numericals 0 - 9) :

048 049 050 051 052 053 054 055 056 057 065 066 067 068 069 070 071 072 073 074 075 076 077 078 079 080 081 082 083 084 085 086 087 088 089 090 097 098 099 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 192 193 194 195 196 197 199 200 201 202 203 204 205 206 207 209 210 211 212 213 214 216 217 218 219 220 221 224 225 226 227 228 229 231 232 233 234 235 236 237 238 239 241 242 243 244 245 246 248 249 250 251 252 253 255

E.g. omitted is the range 091-097 for characters such as [, } and /.

By adding the values 048 - 057 to the table, and re-indexing your database (don't forget to use 'mpl' in the extraction format), you will then get the numerical strings as well in the dictionary, making them searchable.

In the same way, by manually editing another table 'ISISUC.TAB' one can change the way how ISIS/ABCD transposes characters from their lower-case to their upper-case value. Normally variants like e, é, è or ë, even Ë, É, È etc. will be all converted to the same uppercase value for E (ASCII 069), but if one does not want this, in the second part of the table - representing the ANSI-values to which the characters of the first part will be converted when indexing - the values can be changed accordingly. For example : if one needs 'ñ' to translate to 'Ñ' instead of the default 'N', change the value '241' on position 241 of the ANSI-table to '209'.

2.3.3. PFT's in ABCD

We stated it already before : the ISIS Formatting Language makes ISIS so special and powerful, at least at the side of the local system managers, who can define to a high degree the exact behavior of their system without having to know how to program it. One can say of course that the Formatting Language is a programming language in itself, but we protest : the learning curve is much less as the FL follows the logics of librarians e.g. much more than a programming language does. Anyway thousands of librarians all over the world have already proven to be capable of doing marvellous things with this 'intermediate' layer of control : intermediate because in our view it comes nicely in between the end-user interface control (clicking on buttons, selecting from a menu etc.) and the real system programming.

All in all, there are only a limited elements to be understood and used, but by doing this for each and every relevant field in your database, the whole series of instructions starts looking quite complicated. Therefore the 'art' of reading or understanding, thus also writing a PFT is to split the scripts in the many mostly short-and-easy sub-statements and not being frightened by their multitude. Real PFT's vary from the most simple atomic statements like 'v1' upto lists of thousands of concatenated statements in files of several Kilobytes. One of the biggest PFT's used in ABCD is e.g. 'marclte.pft' (the format for MARC Lite), taking about 14Kb, which is the equivalent of several dense text-pages).

From a technical point of view, a PFT is a text-file, so it can be edited also directly in the OS with a text-editor (Notepad, nano...). The only difference then is that if working not within ABCD one has to also manually secure inclusion of new formats in the list of available PFT's, which is actually the file '**formatos.dat**' in the relevant language folder of the pfts-subfolder of your database. As with all ABCD 'tables' or 'data'-files they have 2 columns separated by a pipe : first comes the internal identifier, next the display-text for that value.

Whereas this manual on ABCD does not pretend to also be a course or manual on the ISIS Formatting Language, we need to mainly refer to the CISIS Formatting Language Reference Manual, published by BIREME. PDF-versions of this document (about 42 pages) can be found in several websites (e.g. of BIREME themselves) but also online-versions are available, e.g. http://biblioteca.enap.gov.br/phl8/format34.htm . All manuals are available for download at the URL http://wiki.bireme.org/en/index.php/CISIS .

In this discussion we want to focus on some special uses of the FL to add more functionality into ABCD : cross-referencing and adding JavaScripts.

2.3.3.1. Cross-referencing from record-displays to other search results.

This technique, abbreviated as 'cross-searches', involves presenting in a search results hyperlinks which, when clicked on, will result in a new search performed on the clicked term. This way e.g. author names or subjects could be presented, facilitating new searches for publications on the same author or on the same (or related) subject. With such techniques one can fully apply the philosophy of the 'FRBR' standard (Functional Requirements for Bibliographic Records) which promotes navigating accross different levels from its model (e.g. from 'works' to their concrete emanations in different expressions : movies, poetry, novels...).

In fact in ABCD two methods for this are implemented : a new search result presentation within ABCD Central or passing on the new search to the iAH OPAC module of ABCD to be opened there in a new window. Basically the same techniques are applied to reach these two results :

1. include a JavaScript in 'prologoact.pft' : prologoact.pft is a PFT (in the subfolder bases/www) which is called by ALL ABCD display windows as a 'prolog' to prepare the results window, so this is the ideal location to put code which should be 'known' by all following formats : (note that the whole text is cited in between single quotes as all non-database strings need to be quoted in the ISIS FL) :

   where, as can be seen, an anchor <a href> creates a normal HTML hyperlink passing the author-name and the prefix 'AU_' (used to index authornames) to the Javascript.

   ```
   ' <script>

   function CruzarABCD(Termino,Prefijo){
   ```

```
top.browseby="search"

top.Expresion="\""+Prefijo+Termino+"\""

top.mfn=1

top.Menu("ejecutarbusqueda");

}

</script> '
```

Then use this JavaScript inside your PFT by including a statement like the following one (here we use it for V100 as an author-field) :

```
(if p(V100) then `<a href='javascript:CruzarABCD("`v100`","AU_")'>`v100`</a>` fi/)
```

2. Use a JavaScript to send a search to the OPAC, again quoted inside the prolog-format prologoact.pft in the bases/www subfolder :

```
'<script>

function Cruzar(Termino,Prefijo,Bd) {

document.cruzar.exprSearch.value=Termino

document.cruzar.indexSearch.value=Prefijo

document.cruzar.base.value=Bd document.cruzar.submit()

}

'
```

and refer to this JavaScript inside your PFT as follows, in a completely similar way as above :

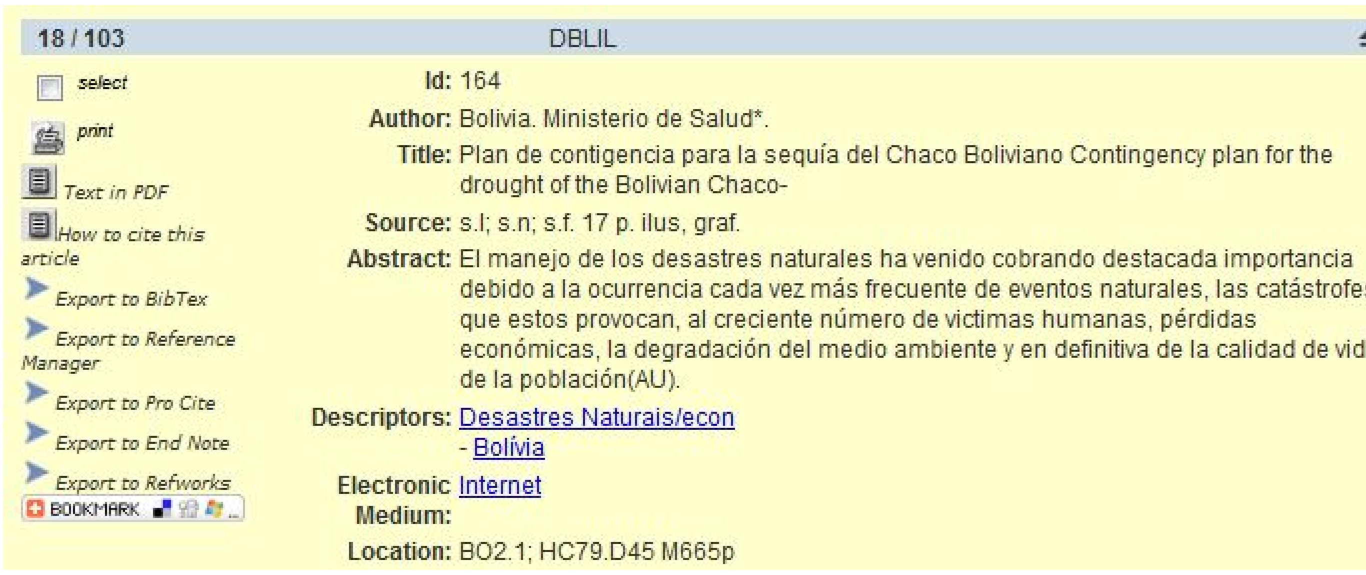
```
(if p(V100) then `<a href='javascript:Cruzar("`v100`","AU")'>`v100`</a>` fi/).
```

However for this to work one has to add a third step : add the following code to the 'epilogo.pft' (as prologoact.pft in the bases/www subfolder) :

```
<form name=cruzar action=/cgi-bin/wxis.exe/iah/scripts/ method=post target=_blank>

<input type=hidden name=IsisScript value=iah.xis>

<input type=hidden name=lang value=es>

<input type=hidden name=base>

<input type=hidden name=nextAction value=lnk>

<input type=hidden name=exprSearch>

<input type=hidden name=indexSearch>

</form>
```

When using such JavaScripts in PFT's for the iAH public search module of ABCD - so not in the Central module - the source files have to be put in the special PFT 'htdocs/iah/scripts<lang>/ahhead.pft'.

An example of such display of a search result record (in this case taken from the iAH interface but still using an ISIS PFT) shows some descriptors as hyperlinks :


18 / 103 DBLIL

select
print
Text in PDF
How to cite this article
Export to BibTex
Export to Reference Manager
Export to Pro Cite
Export to End Note
Export to Refworks
BOOKMARK

Id: 164
Author: Bolivia. Ministerio de Salud*.
Title: Plan de contigencia para la sequía del Chaco Boliviano Contingency plan for the drought of the Bolivian Chaco-
Source: s.l; s.n; s.f. 17 p. ilus, graf.
Abstract: El manejo de los desastres naturales ha venido cobrando destacada importancia debido a la ocurrencia cada vez más frecuente de eventos naturales, las catástrofe que estos provocan, al creciente número de victimas humanas, pérdidas económicas, la degradación del medio ambiente y en definitiva de la calidad de vid de la población(AU).
Descriptors: Desastres Naturais/econ - Bolívia
Electronic Medium: Internet
Location: BO2.1; HC79.D45 M665p

2.3.3.2. Adding JavaScripts into ABCD-formats

In fact the above described technique of 'cross-searching' already illustrates the use of JavaScripts inside ABCD-formats. Since JavaScript is a general-purpose language inside web-pages, and ABCD produces webpages all the time, we can use JavaScript for many other additional purposes.

An example is to use JavaScript to present repeatable subfields, something the ISIS FL does not do itself. The set is completely the same as the one used for cross-searches : include the script-code in the prologoact.pft to make it available to all ABCD PFT's, and call the scripts passing the correct parameters. Here is the example in concrete code, again quoted in single quotes :

```
'<script> function FormatearSubcamposRepetibles(fld,sep_sc,sep_occ) {

result=""

occ=fld.split("$$$")

limit=occ.length

for (ix_occ=0;ix_occ<limit;ix_occ++) {

fld=occ[ix_occ]

c=fld.split("^")

total=c.length-1

for (ix=0;ix<=total;ix++) {

if (c[ix]!="") {

result+=c[ix].substr(1)

if (ix!=total) result+=sep_sc

}

}

result+=sep_occ

}

return result
}

</script> '
```

This scripted function can be called as follows inside a PFT, e.g. to display a v100 with repeated subfields :

```
'<script>

salida=FormatearSubcamposRepetibles("'v100+|$$$|'"," -- ","<br>")

document.writeln(salida)

</script> '
```

In this call we use v100 again but add the string '$$$' to it artificially, so the JavaScript can use this special string to separate occurrences; next the same 'split' JavaScript function will be used to split into subfields with the separator '^' and count the 'lenght' of the subfield as value 'total' in the script. For each resulting subfield the occurrence separator passed (in this case : ' -- ') will be added to the resulting subfield and concatenated into one string as the final result.

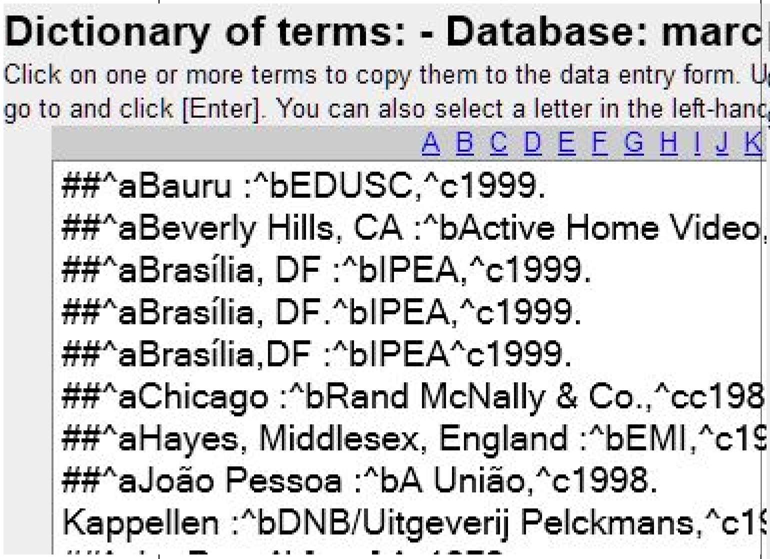
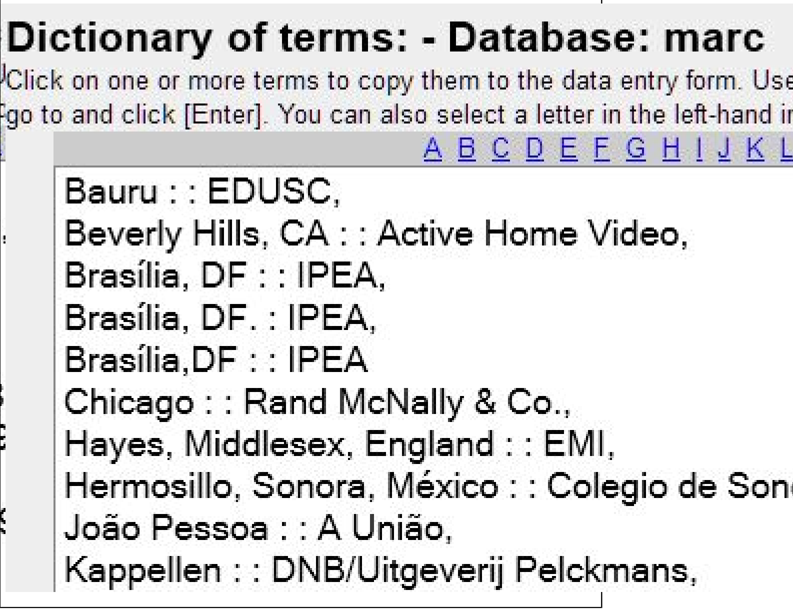
In ABCD v1.2t and later this technique is already provided to get another nice presentation trick : open or close (collapse) certain (groups of) fields in the display.

With the power of JavaScript combined with the ISIS FL 'the sky is the limit'. We have seen very nice example applications of this principle, e.g. alternating background and hovering-characteristics of lists of records displayed in the ABCD Opac, based on JQuery (another 'library' of JavaScript tools). So the basic idea is always : use the power of the FL for the formatting of data from the databases and add the enormous richness of the WWW-technology like JavaScript to make it more attractive.

2.3.3.3. Special PFT's to be used in ABCD FDT's and FMT's to format authority lists ('list as' and 'extract as')

A quite different use of special PFT's is dealt with separately and is supported only by internal ABCD-mechanisms (as coded into the ABCD-scripts). Such PFT's have as their only purpose to create specific picklists as defined inthe ABCD FDT or FMT. E.g. in the MARC-database demo the Imprint field (v260) has a 'list as' PFT stating : v260^a,if p(v260^b) then ` : `v260^b fi (which means : show not only the publishing place but also the publisher name, separated by a ' : ' , if present). The result would be as can be seen in the right-hand picklist, whereas if the 'list-as' PFT would only be 'v260' (the default), it would look like in the left-hand example :

Table 2.2. Comparison : without and with 'list as' PFT

List as : not specified	List as : PFT specified
 **Dictionary of terms: - Database: marc** Click on one or more terms to copy them to the data entry form. U go to and click [Enter]. You can also select a letter in the left-hanc A B C D E F G H I J K ##^aBauru :^bEDUSC,^c1999. ##^aBeverly Hills, CA :^bActive Home Video, ##^aBrasília, DF :^bIPEA,^c1999. ##^aBrasília, DF.^bIPEA,^c1999. ##^aBrasília,DF :^bIPEA^c1999. ##^aChicago :^bRand McNally & Co.,^cc198 ##^aHayes, Middlesex, England :^bEMI,^c19 ##^aJoão Pessoa :^bA União,^c1998. Kappellen :^bDNB/Uitgeverij Pelckmans,^c19	 **Dictionary of terms: - Database: marc** Click on one or more terms to copy them to the data entry form. Use go to and click [Enter]. You can also select a letter in the left-hand i A B C D E F G H I J K L Bauru : : EDUSC, Beverly Hills, CA : : Active Home Video, Brasília, DF : : IPEA, Brasília, DF. : IPEA, Brasília,DF : : IPEA Chicago : : Rand McNally & Co., Hayes, Middlesex, England : : EMI, Hermosillo, Sonora, México : : Colegio de Son João Pessoa : : A União, Kappellen : : DNB/Uitgeverij Pelckmans,

List as : not specified	List as : PFT specified
values are in 'raw' format	values are formatted according to PFT

The example used here has as its purpose to extract from an authority database (mostly not the actual database) a list of fields which will be automatically added to the current record during data-entry when an option has been selected from the picklist. Needless to say that again this can result in very powerful new functions, e.g. adding not only the name of a publisher house but also its address fields into the record, or as in the example below : adding details about a journal after having selected its name. The fields involved in the example are v1 (journal title), and v2, v3 etc. as automatically-added fields

- first step : add the database with journal titles and details as 'external DB' for the picklist in your FDT and specify the prefix used to index all journal titles
- in the 'list as' column of your FDT or FMT, enter the name of the script doing the trick, e.g. 'transfer.pft
- in your 'transfer.pft' put the following code :

  v1'$$$','_TAG11:'V1,'_TAG12:'V2,'_TAG13:'V3,'_TAG14:'V4,

  where : v1 is the 'list as' format (as simple as possible to list the journal titles, '$$$' is the separator in between the 'list as' and 'extract as' part of the PFT (as agreed in ABCD), followed by for each automatically transferred field : first the tag of the receiving field (the value after '_TAG'), a full colon (:) and the value from the referred database to put in that local-database field. So '_TAG01:'v1 will extract V1 from the external database (the journals list) and put it into the local field v11, while v2 will go to V12 etc.

2.3.3.4. PFT's for validation

When validating records or fields, ABCD uses the same ISIS FL to express the exact instructions of what to validate. The same elements are used : vX, IF...THEN...ELSE...FI constructs, substring functions and of course also 'literals', i.e. quoted strings in this case used to produce the resulting message (mostly an error message) on the screen.

One interesting example, illustrating the said elements, is the 'duplication check' validation, in this case we apply it to check if a user in the USERS-database already exists - this is derived from the very similar validation PFT used to check for ISBN duplicates to avoid unwanted duplication of titles in the catalog (and briefly discussed in the cataloging chapter).

The PFT goes as follows :

```
if a(v20) then User code missing' else if a(v3333) then proc('a3333~'f(mfn,1,0)'~') fi
if val(v3333)=0 then if l(['users']|CO_|v20^a) <> 0 then 'User already exists in MFN
',f(l(['users']|CO_|v20^a),1,0), fi fi fi
```

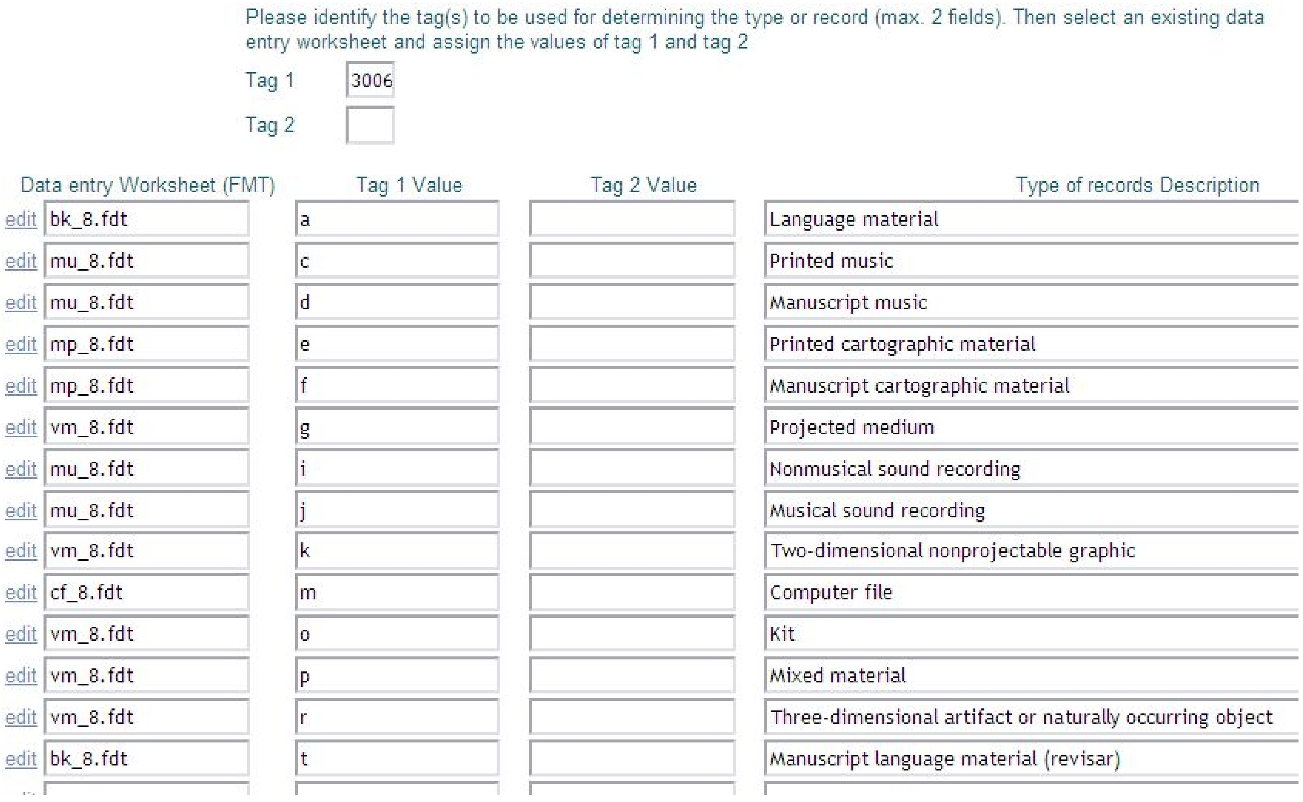
Explanation : first the presence of the users-code field v20 is checked and reported if absent, then the MFN-value is put into v3333 (an virtual field for internal purposes only) if it does not yet exist and then this value, preceded by the prefix 'CO_' is 'L'ooked up into the users Inverted File, and if found (meaning it already exists) the message reports in which MFN it exists.

2.3.4. Type of records

Some database-structures, such as MARC, require the 'type of the record' to be specifically coded into a dedicated field of the record. The software can then use this code to adjust many features to the specific needs for the type indicated, e.g. worksheets and print-formats can be different according to this type, or simply - as is the case with MARC - the format wants to be very detailed.

The 'type of record' information needs to be gathered at the beginning of the creation of a new record, so a list of types defined (and therefore 'available') will be presented as links, each one leading to the appropriate subsequent data-entry form, as can be seen from the MARC demo in ABCD.

[!!] In order to define such types, ABCD uses a table 'Typeofrecord.tab' - located in the 'def' folder for the actual language within the www/bases/[DB]/ folder. This file is - as is often the case in ABCD - an ASCII-file which contains, for each type defined, 4 values separated by a '|' (pipe-character), so this can be edited directly with an ASCII-editor (e.g. Notepad), but within ABCD an easier-to-use table format is presented :


Please identify the tag(s) to be used for determining the type or record (max. 2 fields). Then select an existing data entry worksheet and assign the values of tag 1 and tag 2

Tag 1 3006

Tag 2

	Data entry Worksheet (FMT)	Tag 1 Value	Tag 2 Value	Type of records Description
edit	bk_8.fdt	a		Language material
edit	mu_8.fdt	c		Printed music
edit	mu_8.fdt	d		Manuscript music
edit	mp_8.fdt	e		Printed cartographic material
edit	mp_8.fdt	f		Manuscript cartographic material
edit	vm_8.fdt	g		Projected medium
edit	mu_8.fdt	i		Nonmusical sound recording
edit	mu_8.fdt	j		Musical sound recording
edit	vm_8.fdt	k		Two-dimensional nonprojectable graphic
edit	cf_8.fdt	m		Computer file
edit	vm_8.fdt	o		Kit
edit	vm_8.fdt	p		Mixed material
edit	vm_8.fdt	r		Three-dimensional artifact or naturally occurring object
edit	bk_8.fdt	t		Manuscript language material (revisar)

The example above is the MARC record-type definition, which is kept in (internal) tag 3006 and has the following 4 columns, each containing values for each type :

- the name of the worksheet to be used for the given type of record - with the 'edit'-link next to this first column one can also immediately edit the worksheet
- the 'Tag1' value, which is in fact a one-character code to internally identify the type of record
- the 'Tag2' value, not used in this case
- the description of the type as it will appear in the list of available types.

  E.g in the CEPAL database in order to define a type 'Analytical in a monograph' he definition would be : Tag1 = v4 = M and Tag2 = v5 = am

Clicking on 'update' below the form will save the table with any changes made.

2.3.5. Record validation

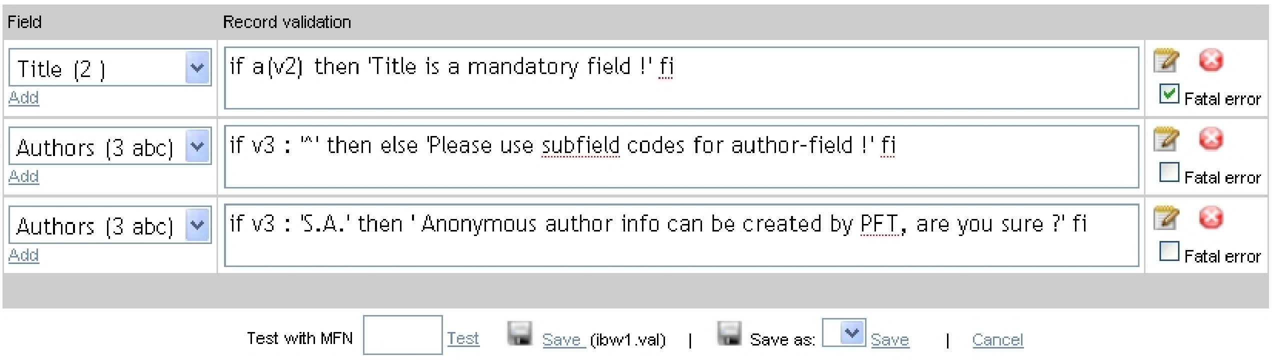
[!!} Record validation allows the database manager to define criteria against which input for a field can and will be checked before entering it actually in the fields, or after registration of the record. Record validation in ABCD can relate to one of the following elements :

- record validation : conditions can be given for each field
- begin format : code can be given to be executed when a (new) record is created, e.g. the date of creation can be added with the date() instruction
- end format : code can be given to be executed when the record is saved, e.g. the date of last update can be added with the date() instruction.

  After selecting on of the three options above, clicking on one of the listed (as pre-defined) record-types will show the editor where the validation conditions can be defined.

These criteria need to be formulated into - no surprise ! - the ISIS Formatting Language. With the Formatting Language one can check a condition and if (not) met, an error message, which will be shown on the next screen, can be produced by the format. [!!] If the validation criterion was defined as 'fatal' however the record will not be saved and the error has to be corrected first.

In this menu-option from the 'Update database definitions' menu each defined field will be listed with a box to enter the validation statement. E.g. :



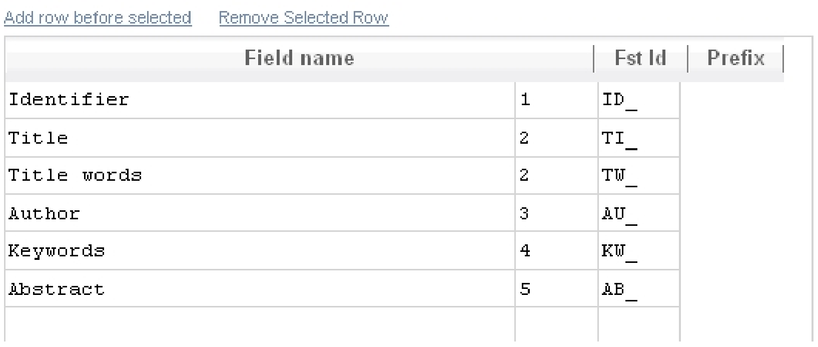
Field	Record validation	
Title (2) Add	if a(v2) then 'Title is a mandatory field !' fi	✓ Fatal error
Authors (3 abc) Add	if v3 : '^' then else 'Please use subfield codes for author-field !' fi	☐ Fatal error
Authors (3 abc) Add	if v3 : 'S.A.' then ' Anonymous author info can be created by PFT, are you sure ?' fi	☐ Fatal error

The format used here checks on the 'absence' of a value in field with tag 2 and if indeed absent will produce an error message that this mandatory field is missing. [!!] As stated above, errors can be marked as 'fatal' or not. With the 'ADD' link in the left-most part for each field. As shown in the illustration above, with the 'ADD'-link in the left-most column one can add more fields, even fields already used with another validation criterion to be applied.

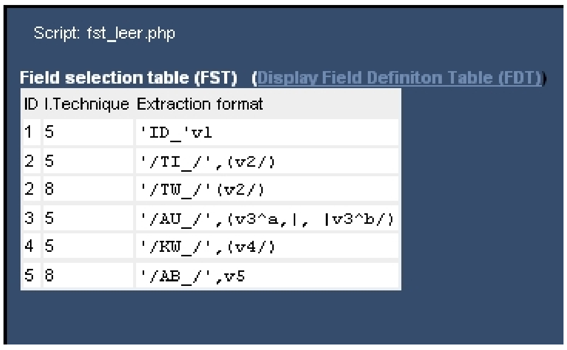
When clicking on the 'edit' icon to the right of the edit-window for this field, the box re-appears in a separate small window for editing and the statement can also be tested on a record to see if it is doing the right thing. After editing one has to click on 'send' to put the possibly edited validation format back to the main table.

2.3.6. Advanced search form

The advanced search form is the one used in ABCD at two locations : within the cataloging module to allow the cataloger to quickly and/or efficiently identify a specific record for editing (or duplication checking or copying) and in the OPAC as the advanced search form. Here ABCD simply offers an editor for the table which defines that search form with 3 columns :



Field name	Fst Id	Prefix
Identifier	1	ID_
Title	2	TI_
Title words	2	TW_
Author	3	AU_
Keywords	4	KW_
Abstract	5	AB_



Script: fst_leer.php

Field selection table (FST) (Display Field Definiton Table (FDT))

ID	I.Technique	Extraction format
1	5	'ID_'v1
2	5	'/TI_/',(v2/)
2	8	'/TW_/'(v2/)
3	5	'/AU_/',(v3^a,|, |v3^b/)
4	5	'/KW_/',(v4/)
5	8	'/AB_/',v5

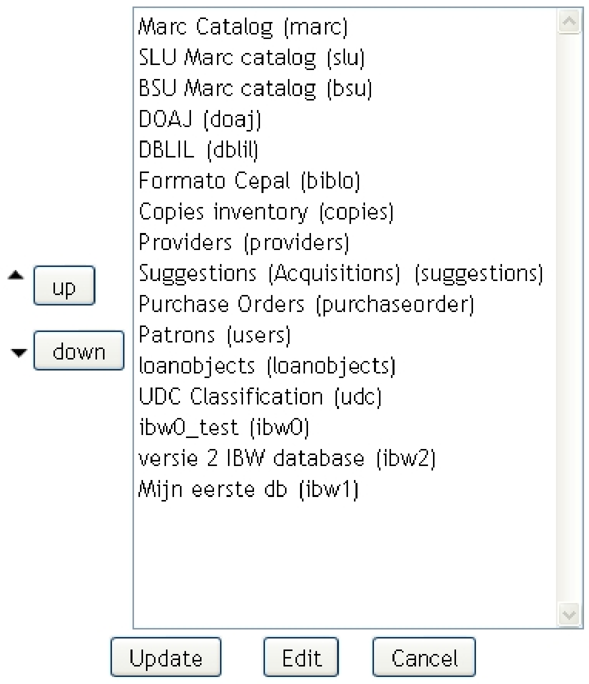
- The first column is the field-name or 'index' as it will appear in the search-form
- The second column gives the identifier used for the given field (or in fact combination of fields) in the FST
- The third and last column keeps the prefix or fixed start-string which is used (if any) in the ISIS Inverted File for this index.

  ABCD will also present, next (to the right) to this table, the existing FST to facilitate the identification of indexes (fields for searching) and the identifiers used.

  Clicking on 'update' saves the table, which in fact is a file 'busqueda.tab', stored in the language subfolder of the 'pfts' folder within the database folder.

2.3.7. Available databases list or table

Here simply a list is given of the databases being defined as 'available' in the ABCD-system. Actions allowed here are only changing the sequence (by moving up or down) of the databases in the list and saving the changed list.

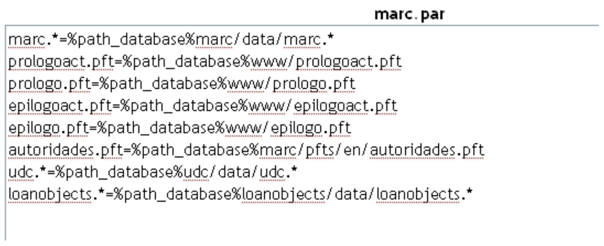


```
Marc Catalog (marc)
SLU Marc catalog (slu)
BSU Marc catalog (bsu)
DOAJ (doaj)
DBLIL (dblil)
Formato Cepal (biblo)
Copies inventory (copies)
Providers (providers)
Suggestions (Acquisitions) (suggestions)
Purchase Orders (purchaseorder)
Patrons (users)
loanobjects (loanobjects)
UDC Classification (udc)
ibw0_test (ibw0)
versie 2 IBW database (ibw2)
Mijn eerste db (ibw1)
```

For adding or deleting databases one has to actually create the new database or delete the one to be taken out of the list. ABCD will take care of the changes in this list automatically. [!!} Databases can be moved up or down by selecting them and using the UP/DOWN buttons as in many of such controls in the ABCD-interface.

2.3.8. [dbn].par

For each ISIS-database in a multi-database application such as ABCD, there is a file needed to tell ISIS where to find the constituting parts of the database-files - which then consequently can reside anywhere in the system. Such files are named after the database-name (therefore indicated here as [dbn] with the .par extension. Again this is a simple ASCII-file which can be edited directly or, as is the case here, from this ABCD-menu.



marc.par

```
marc.*=%path_database%marc/data/marc.*
prologoact.pft=%path_database%www/prologoact.pft
prologo.pft=%path_database%www/prologo.pft
epilogoact.pft=%path_database%www/epilogoact.pft
epilogo.pft=%path_database%www/epilogo.pft
autoridades.pft=%path_database%marc/pfts/en/autoridades.pft
udc.*=%path_database%udc/data/udc.*
loanobjects.*=%path_database%loanobjects/data/loanobjects.*
```

In principle ABCD will take care of this file and make sure that the necessary paths are available. The only special feature here - as compared to the same concept of dbn.par in other ISIS-environments, is the use of 'variables', taken from the operating system's environment variables, which can be dynamically substituted for their actual values. E.g.

%path_database%

is a variable which actually will contain the database-path defined in the config.php main configuration file of ABCD.

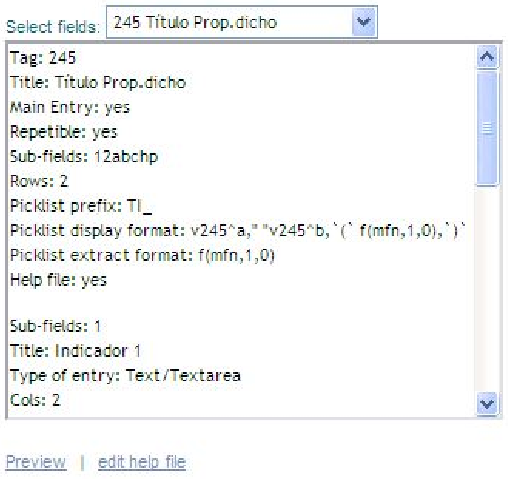
[!!] In the example of the illustration above, the last line is an interesting one as it gives the path to the 'loanobjects'-database for displaying copy-information (if included in one of the MARC-pft's) with the REF-function. In order to find the loanobjects-database for the use of REF->loanobjects, ISIS needs to know the path to this other database and therefore it should be included into the dbn.par.

Note

[!!] For this last feature also to work in the OPAC (iAH), one has to add the path to the first section of the dbn.def file. E.g. the loanobjects-database should be pointed to by a new line there containing : FILE loanobjects.*=%path_database%loanobjects/data/loanobjects.*

2.3.9. Help files on the database-fields

For each field in the database ABCD can provide a help-page, which can be edited from this menu-option. To support the creation of such a page, ABCD will automatically put all known information from the FDT on the given field already available. With the built-in (JavaScript-based) HTML-editor a real nice help-page can then be produced and saved. The 'preview' link of course allows checking the result of your editing effort.



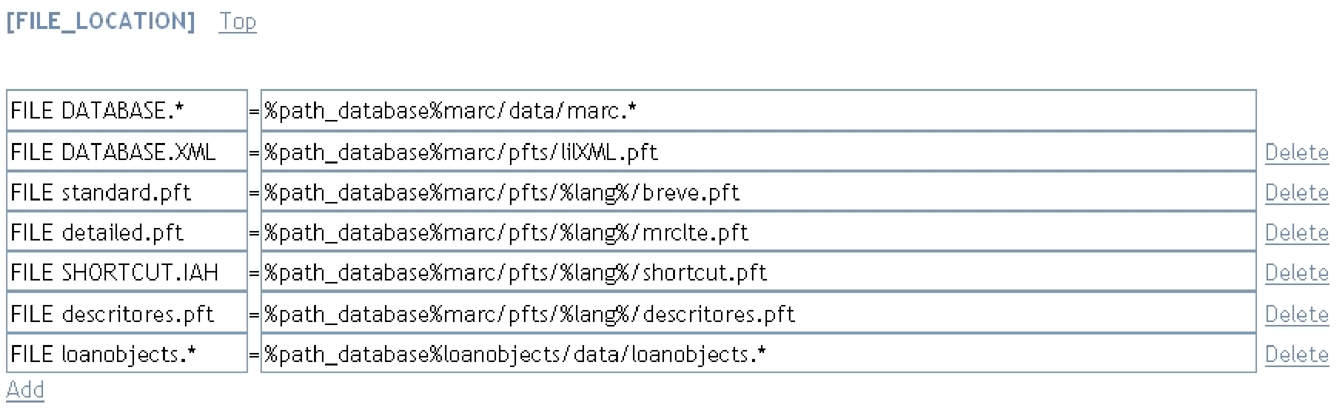
```
Tag: 245
Title: Título Prop.dicho
Main Entry: yes
Repetible: yes
Sub-fields: 12abchp
Rows: 2
Picklist prefix: TI_
Picklist display format: v245^a," "v245^b,`(` f(mfn,1,0),`)`
Picklist extract format: f(mfn,1,0)
Help file: yes

Sub-fields: 1
Title: Indicador 1
Type of entry: Text/Textarea
Cols: 2
```

2.3.10. Configure database in iAH (or OPAC)

[!!] In order to be able to use a newly defined ABCD-database with the advanced iAH OPAC-interface (or in the ABCD Site), a special configuration file is to be edited : *dbn*.def. (where *dbn* has to be substituted for the actual database-name).

This file has the following sections to be edited :

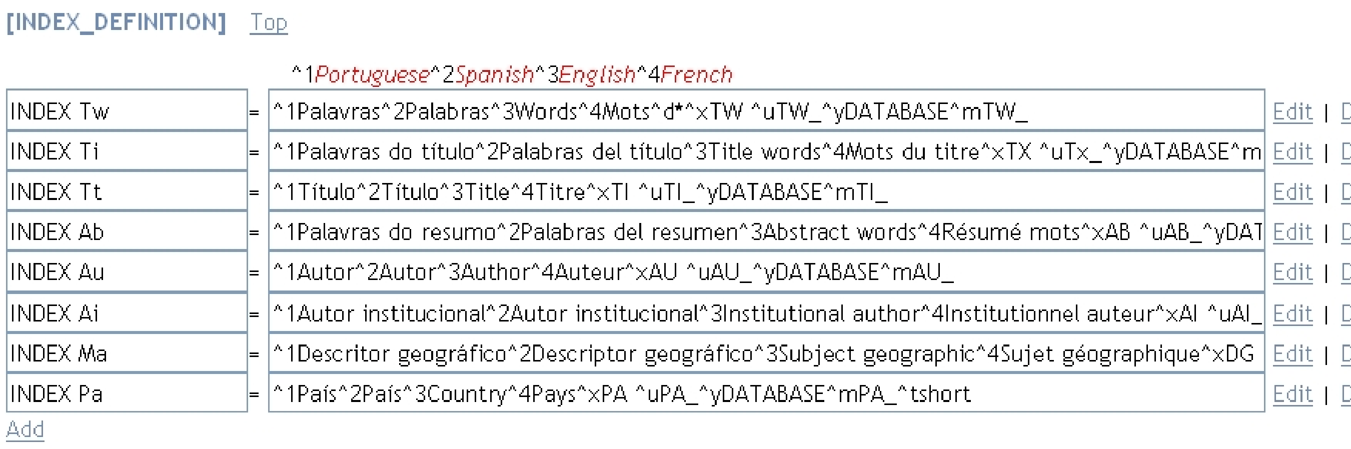
1. The FILES section : here the paths to the files to be accessed need to be given. In this path both the %path_database% and %lang% variables can be used to refer to resp. the actual database and actual language in use.



FILE DATABASE.*	=	%path_database%marc/data/marc.*	
FILE DATABASE.XML	=	%path_database%marc/pfts/lilXML.pft	Delete
FILE standard.pft	=	%path_database%marc/pfts/%lang%/breve.pft	Delete
FILE detailed.pft	=	%path_database%marc/pfts/%lang%/mrclte.pft	Delete
FILE SHORTCUT.IAH	=	%path_database%marc/pfts/%lang%/shortcut.pft	Delete
FILE descritores.pft	=	%path_database%marc/pfts/%lang%/descritores.pft	Delete
FILE loanobjects.*	=	%path_database%loanobjects/data/loanobjects.*	Delete

2. The INDEX-DEFINITION section : here all the information for the active languages (numbered as 1, 2, 3 etc.) has to be entered (in subfields) to allow the interface to recognize the prefixes used for each searchable

field and to name the field accordingly. The first line with prefix 'TW_' is the one used for the 'simple' search interface (Google-like) for searching words from the fields defined (by the prefix in the FST) to be included in this simple search. This is indicated by the presence of the subfield '^d*'.



[INDEX_DEFINITION] Top

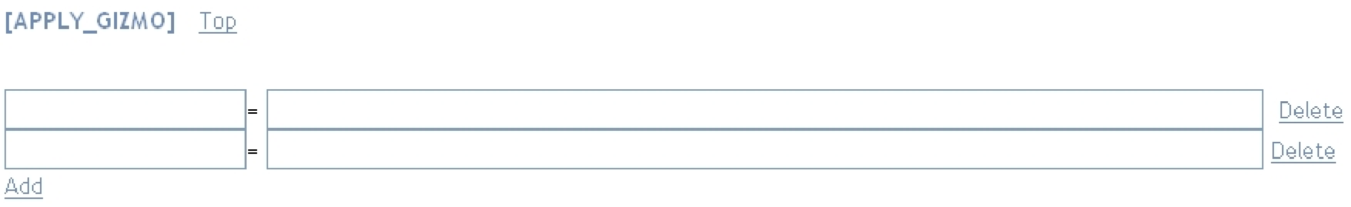
^1*Portuguese*^2*Spanish*^3*English*^4*French*

INDEX Tw	=	^1Palavras^2Palabras^3Words^4Mots^d*^xTW ^uTW_^yDATABASE^mTW_	Edit | D
INDEX Ti	=	^1Palavras do título^2Palabras del título^3Title words^4Mots du titre^xTX ^uTx_^yDATABASE^m	Edit | D
INDEX Tt	=	^1Título^2Título^3Title^4Titre^xTI ^uTI_^yDATABASE^mTI_	Edit | D
INDEX Ab	=	^1Palavras do resumo^2Palabras del resumen^3Abstract words^4Résumé mots^xAB ^uAB_^yDAT	Edit | D
INDEX Au	=	^1Autor^2Autor^3Author^4Auteur^xAU ^uAU_^yDATABASE^mAU_	Edit | D
INDEX Ai	=	^1Autor institucional^2Autor institucional^3Institutional author^4Institutionnel auteur^xAI ^uAI_	Edit | D
INDEX Ma	=	^1Descritor geográfico^2Descriptor geográfico^3Subject geographic^4Sujet géographique^xDG	Edit | D
INDEX Pa	=	^1País^2País^3Country^4Pays^xPA ^uPA_^yDATABASE^mPA_^tshort	Edit | D

Add

3. The GIZMO section : here - if necessary in specific cases - gizmo databases for automatic substitution of strings by other strings (which can be used to change character sets, but also change codes and abbreviations into full values etc...) can be pointed to.

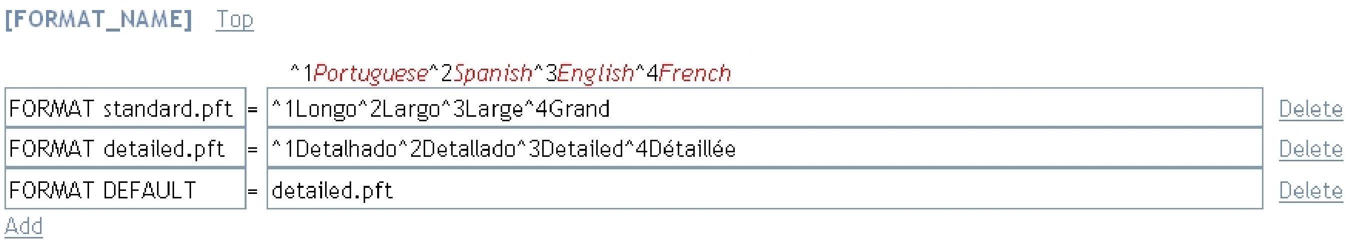
4. The FORMAT section : here the display formats used in the OPAC should be defined, with for each language (in the numbered subfields) the label to be used on the screen. Remember that only formats (files with .PFT extension) can be used which are referred to somehow in the FILES section ! Also the default format can be identified here by simply indicating which format earlier referred to should be used as default display format.



[APPLY_GIZMO] Top

	=		Delete
	=		Delete

Add

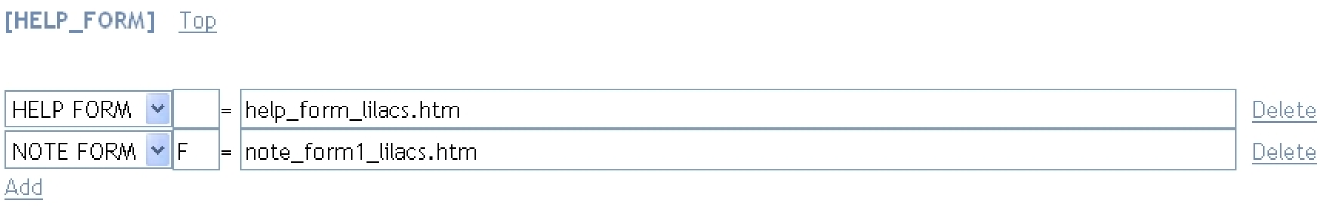


[FORMAT_NAME] Top

^1*Portuguese*^2*Spanish*^3*English*^4*French*

FORMAT standard.pft	=	^1Longo^2Largo^3Large^4Grand	Delete
FORMAT detailed.pft	=	^1Detalhado^2Detallado^3Detailed^4Détaillée	Delete
FORMAT DEFAULT	=	detailed.pft	Delete

Add



[HELP_FORM] Top

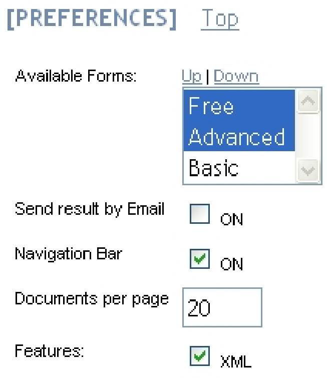
HELP FORM		=	help_form_lilacs.htm	Delete
NOTE FORM	F	=	note_form1_lilacs.htm	Delete

Add

5. The HELP form section : here simply the (HTML-)files containing resp. the help-page and notes-page for the user of this database in the OPAC should be referred to.

6. The PREFERENCES section : here the system manager can indicated which of the three search interfaces (simple, advanced and free) will be available for this database, and some other additional features of the inter-

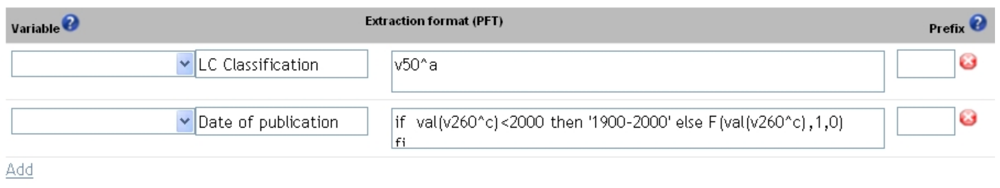
face : whether sending a search result to an e-mail will be possible, whether the results should be listed with navigation buttons, the number of records to be shown in one page and whether XML-export will be offered.


[PREFERENCES] Top

Available Forms:	Up | Down Free Advanced Basic
Send result by Email	☐ ON
Navigation Bar	✓ ON
Documents per page	20
Features:	✓ XML

2.3.11. Statistics : list of variables

This option simply allows quick definition of the variables from the given database with which tables for statistical analysis will be computed. For each criterion or variable (either as row or column for the table) a name and an extraction format has to be given. The extraction format - using the Formatting Language of course - exactly defines how the values in the field should be taken to compute the value in the table. By doing so it is possible e.g. to define ranges of field-values to be combined into one table-criterion. *The file at stake is actually 'stats.cfg' in the (language-specific) def-section of the database-folder.*



Variable		Extraction format (PFT)	Prefix
	LC Classification	v50^a	
	Date of publication	if val(v260^c)<2000 then '1900-2000' else F(val(v260^c),1,0) fi	

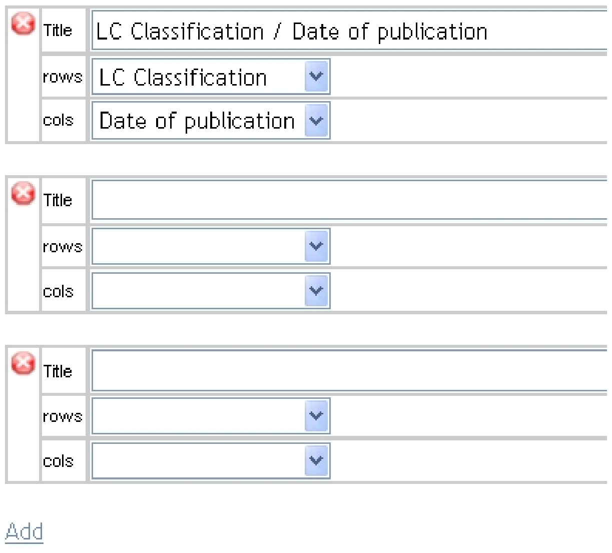
Add

The option to define a prefix is not yet implemented in this version of ABCD. The idea is that the values would be taken from the Inverted File, prefixed with the string defined here. By doing so the values would be computed while 'inverting' the record, not at the stage of producing the statistics table, and therefore allow faster production of the table.

2.3.12. Statistics : list of tables

As with the list of variables above, ABCD also keeps a simple list of available tables, which have been defined previously, for the statistics module. *This file 'tabs.cfg' equally resides in the def-subfolder of the database.* Each line in this file contains three values (separated by the pipe-character) : the name of the table followed by the two criteria used in this table, e.g. :

```
Classification code / Publication date|Classification LC|Publication date
```



Title	LC Classification / Date of publication
rows	LC Classification
cols	Date of publication

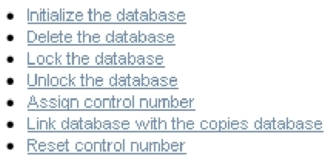
As can be seen from the example the editor in ABCD simplifies editing by providing each of the three values individually but also by providing lists of available row- and column-criteria.

2.4. Reports

In fact creating reports in ABCD means creating ISIS Formatting Language formats (PFT's) with which the reports will create output, because with the F.L. any type of report can be produced and saved for later re-use. We therefore refer to the section on 'Display Format (PFT)' of the 'Update Database Definitions' Central menu option. Exactly the same interface is used here.

2.5. Utilities

In this option ABCD offers some very basic operations on databases :



- Initialize the database
- Delete the database
- Lock the database
- Unlock the database
- Assign control number
- Link database with the copies database
- Reset control number

- Initialize the database means to delete all records in the database but without changing the structures of the database.

- Delete the database of course means fully deleting the whole database with all corresponding files and folders in the ABCD bases/ folder.

- Lock the database means to prevent any other users to make any changes to the records (data-entry), e.g. when a full Inverted File generation would be envisaged.

- Unlock the dabase of course then means to avail the database again to other users.

  Use these options with all necessary care and caution !

- Assign Control Number is the option to automatically put sequential control-numbers in a series of selected records. Only a continuous series of MFN's can be selected here.

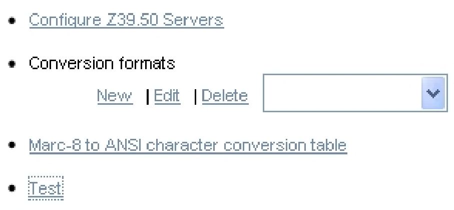
- Link database with the copies database : as explained on the screen itself, here the option to activate the use of the copies database for the actual (bibliographic) database has to be activated.

- Reset Control Number is the function to manually re-set to a specific value the numerical value in the file 'control_number.cn' of the database, in case for some reason the numbering has to be managed manually. E.g. setting this number to 1000 will make ABCD to assign from the next record on control-numbers starting from 1000. This could be useful in conditions where e.g. different cataloging centres are using different ABCD-servers but the resulting databases have to be merged into one catalog with control_numbers not interfering, so covering different ranges. By default however reset will revert to 1 as the basic control_number to start from.

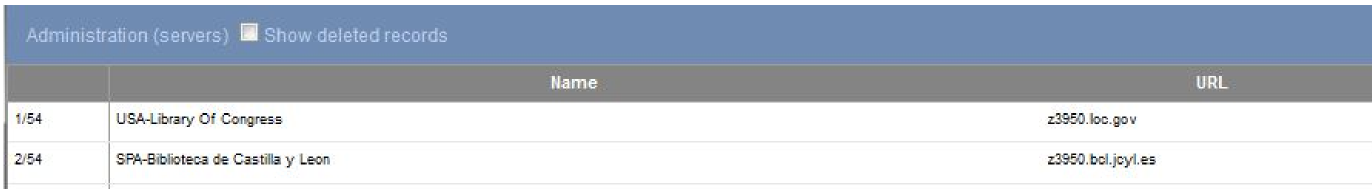
2.6. Z39.50 Configuration

Here the ABCD interface allows setting some parameters to define which servers for Z39.50 shared cataloging services will be offered in the Z39.50 list and some more parameters to ensure proper use of the protocol for the server. Such technical information can be mostly obtained from the service provider, e.g. in the case of the Library of Congress consult the following website : http://www.loc.gov/z3950/lcserver.html.

Configuring Z39.50 has the following parts :

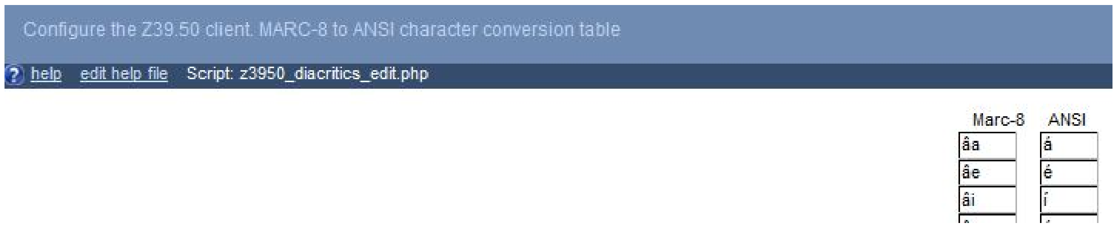

- Configure Z39.50 Servers
- Conversion formats
  New | Edit | Delete
- Marc-8 to ANSI character conversion table
- Test

- Configure Z39.50 servers : in an interface similar to the one of users-administration, the defined servers are to be configured, with the parameters name, URL, Port, Database and UTF-8 (or not), see the illustration under here for Library of Congress. New servers of course can also be added with the 'Create'-icon.


Administration (servers) Show deleted records

	Name	URL
1/54	USA-Library Of Congress	z3950.loc.gov
2/54	SPA-Biblioteca de Castilla y Leon	z3950.bcl.jcyl.es

- Conversion formats : when the incoming records (mostly in MARC21 format) need to be converted to the format used in the destination database, a form can be edited here to specify the conversion from source to destination. The name , subfields and tag of the incoming field will be listed and in the 'Conversion formats' column the ISIS F.L. defines how to extract values for this field in the destination database. ABCD comes with one demo conversion table to convert from MARC to CEPAL formats, with the following example for converting the ISSN MARC field to CEPAL.


Configure the Z39.50 client. MARC-8 to ANSI character conversion table

help edit help file Script: z3950_diacritics_edit.php

Marc-8	ANSI
âa	á
âe	é
âi	í

- MARC-8 to ANSI character conversion table : this is a table which converts characters from the MARC-8 table (e.g. âa) to ANSI format (e.g. á). This table can be edited here if necessary.

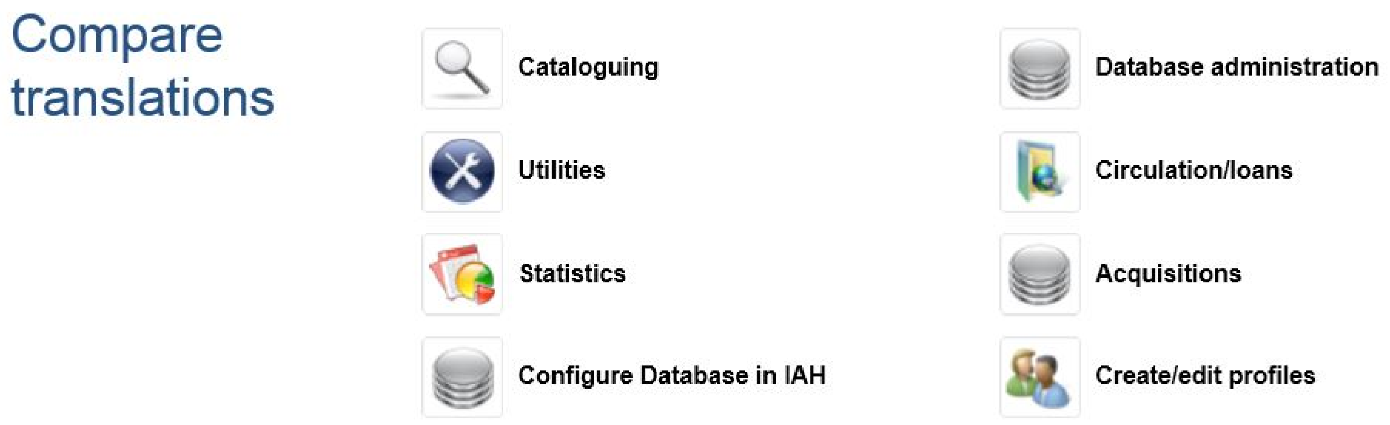
- Finally the Z39.50 can be tested from here, with the same interface as used in the ABCD Data-Entry module (see infra).

2.7. Translate messages and help pages

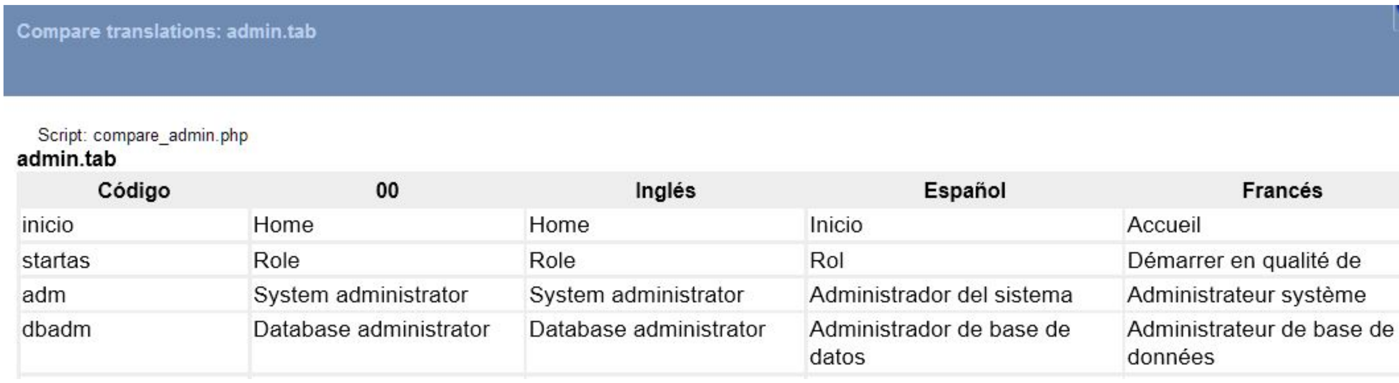
There are two types of language-sensitive content which ABCD uses and which needs to be edited when creating or adapting new or other language versions : short messages and labels on the one hand, full help-pages on the other hand. These can be edited into other languages for each module of ABCD Central.

A handy new feature from v1.2t on is the possibility to see an overview of all messages for different languages next to each other. This can facilitate identification of inconsistencies or the exact meaning of certain messages. For editing the messages subsequently still the messages-editor has to be used.

Figure 2.17. Compare translations



In the partial display of such overview as shown here, the '00' language is the default set, which will be used also when a certain message for a given language is missing.



Compare translations: admin.tab

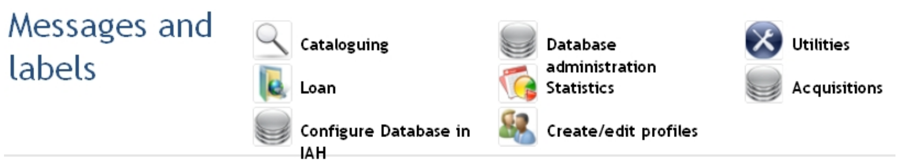
Script: compare_admin.php

admin.tab

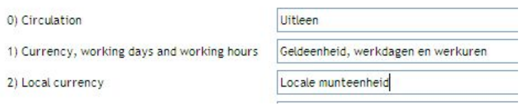
Código	00	Inglés	Español	Francés
inicio	Home	Home	Inicio	Accueil
startas	Role	Role	Rol	Démarrer en qualité de
adm	System administrator	System administrator	Administrador del sistema	Administrateur système
dbadm	Database administrator	Database administrator	Administrador de base de datos	Administrateur de base de données

2.7.1. Translation of short messages and labels

Editing these is facilitated by presenting the default (English) language terms and phrases in a table in wich in the second column the new values should be put by the translator. For each of the main functions such a table is presented :



Here is a sample of some messages translated from English to Dutch for the loans module :

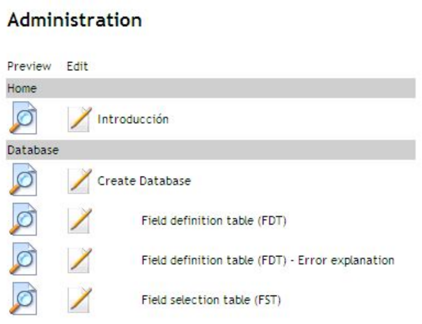


0) Circulation	Uitleen
1) Currency, working days and working hours	Geldeenheid, werkdagen en werkuren
2) Local currency	Locale munteenheid

This screen provides a 'save' icon for storing the table with new translations.

2.7.2. Translation of help pages

Here the approach is different : a list of available help-pages is given



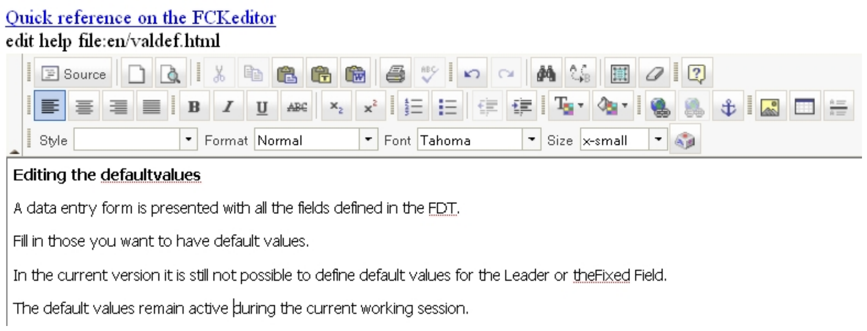
Administration

Preview Edit

Home	
Introducción	

Database
Create Database
Field definition table (FDT)
Field definition table (FDT) - Error explanation
Field selection table (FST)

and for each help-file one can 'preview' or 'edit' the page. In the case of editing the built-in JavaScript-based HTML-editor will be provided :



Quick reference on the FCKeditor
edit help file:en/valdef.html

Style | Format Normal | Font Tahoma | Size x-small

Editing the defaultvalues

A data entry form is presented with all the fields defined in the FDT.

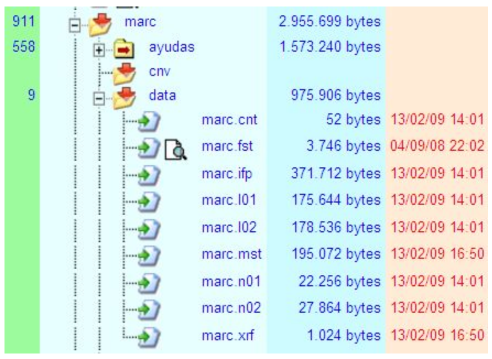
Fill in those you want to have default values.

In the current version it is still not possible to define default values for the Leader or theFixed Field.

The default values remain active during the current working session.

2.8. Explore databases directory

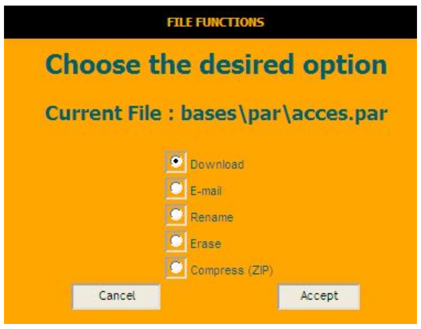
[!!] Exploration of the databases directory can be done (when the dedicated option variable '$dirtree' is put to '1' in CONFIG.PHP !) using a special display in the ABCD-web-page of the folders of the database-folder of the system,



911	marc	2.955.699 bytes	
558	ayudas	1.573.240 bytes	
	cnv		
9	data	975.906 bytes	
	marc.cnt	52 bytes	13/02/09 14:01
	marc.fst	3.746 bytes	04/09/08 22:02
	marc.ifp	371.712 bytes	13/02/09 14:01
	marc.l01	175.644 bytes	13/02/09 14:01
	marc.l02	178.536 bytes	13/02/09 14:01
	marc.mst	195.072 bytes	13/02/09 16:50
	marc.n01	22.256 bytes	13/02/09 14:01
	marc.n02	27.864 bytes	13/02/09 14:01
	marc.xrf	1.024 bytes	13/02/09 16:50

with some possibilities to enter within subfolders and even editing, renaming, zipping etc. some of the text-files in thereon by clicking on the 'details' icon

, given e.g. the following options to apply on the selected file :



FILE FUNCTIONS

Choose the desired option

Current File : bases\par\acces.par

[✓ Download]
E-mail
Rename
Erase
Compress (ZIP)

Cancel Accept

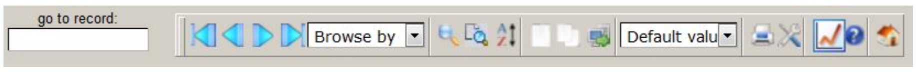
2.9. Statistics

The Statistics module of ABCD also has a dedicated chapter, so here we just refer to this chapter, as this function can also be accessed from this menu but also from the cataloging toolbar and several menu's in the acquisitions and loans modules.

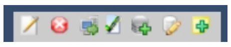
3. Central module : data-entry (cataloging)

In this section we discuss the main techniques of one of the most 'central' functions of ABCD : creating records in a database (in bibliographic applications mostly referred to as 'cataloging').

These frequently used operations for catalogers are grouped into one 'toolbar' on top of the screen - in this way ABCD tries to imitate a 'local desktop software' behavior.



The above shown toolbar is related to the database. Whenever a record is shown, a second smaller toolbar will be shown which contains the buttons related to the current record :



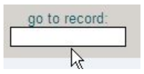
We will discuss all functions of the toolbars here briefly, going from left to right.

3.1. Browsing records

Browsing records, like searching them, is an everyday's cataloger's job since before entering new records usually the cataloger has to check if the record is already in the catalog (which then means only a new copy has to be created from the existing catalog record). The database toolbar provides all necessary tools to do this preparatory work as well as the cataloging work itself.

There are 2 ways for browsing records :

- either by entering the record number (MFN) in the dedicated box

  Here you have to simply enter a number from 1 up to the highest available MFN in the database.

- or by clicking on one of the 'navigation buttons' to go the the first, previous, next, or last records

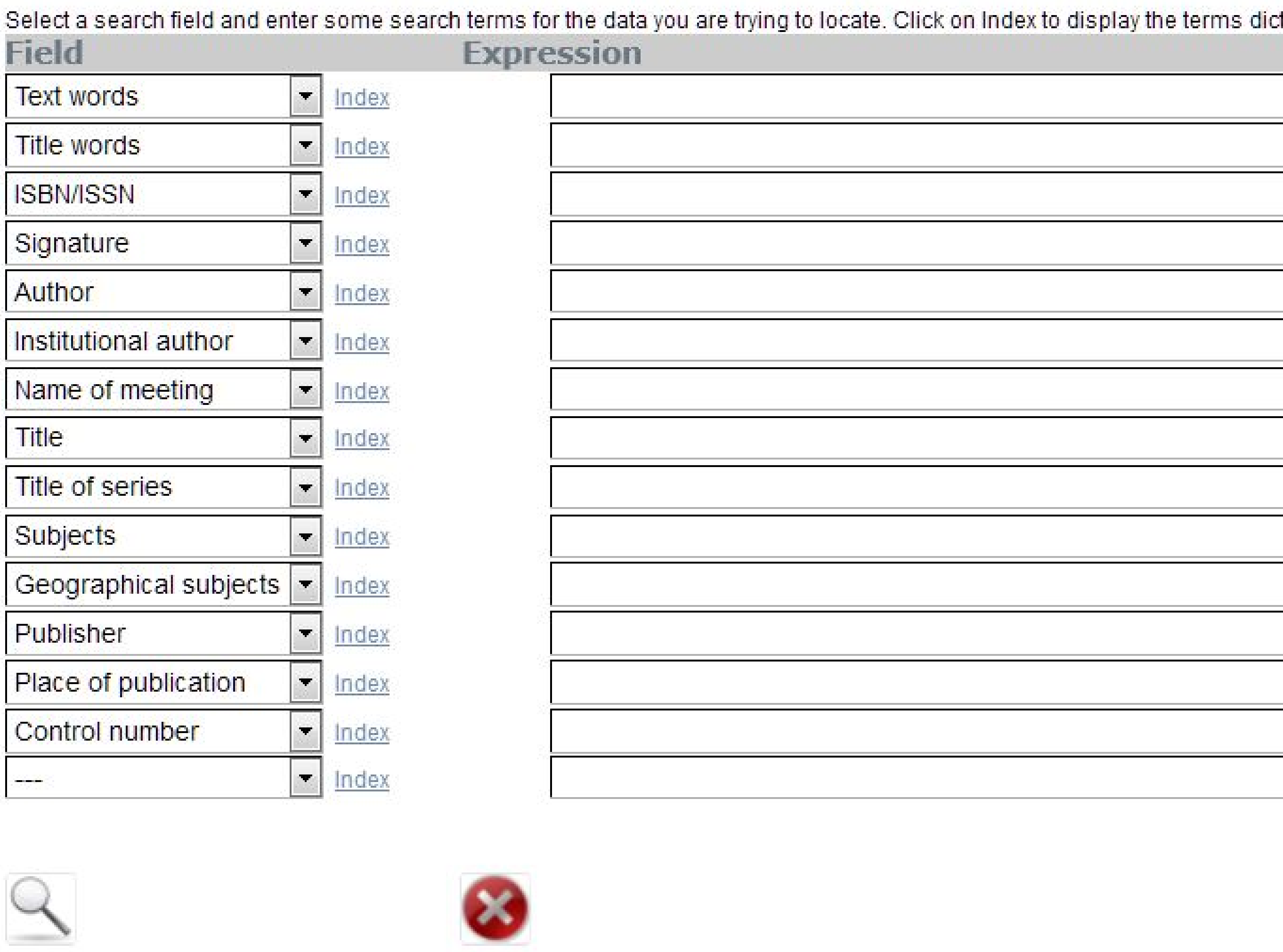
  This navigation will be done either in the full database or in the search result set, according to the selected option in the adjacent menu.

3.2. Searching records

As with browsing, ABCD offers several approaches to identify specific records for catalogers : by either searching or by selecting records through an alphabetic listing (AtoZ)

-

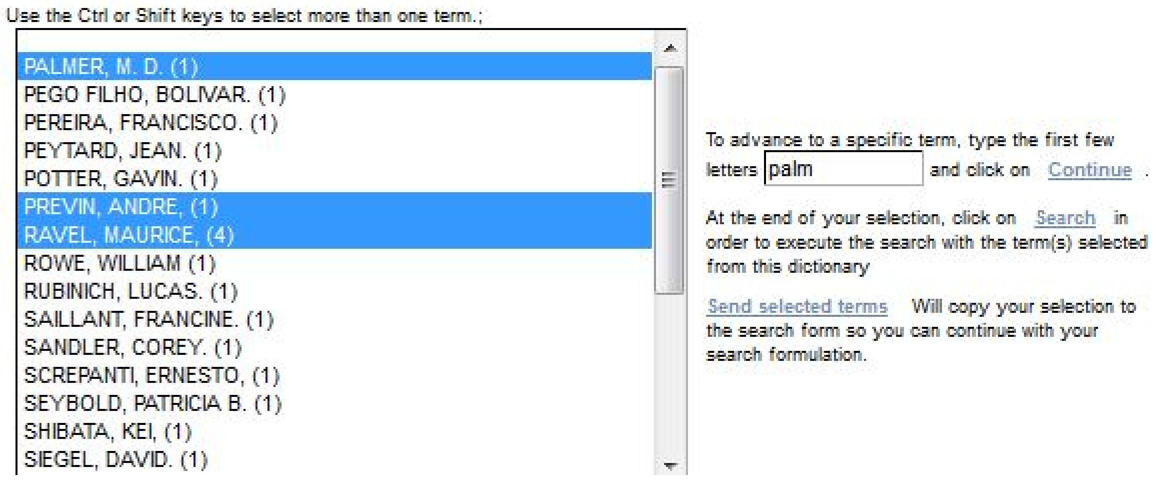
  by performing a search with a powerful search-function built-in into the database administration module, resulting in a real search-form presented (as defined in the 'define search form' function in the main administration menu).



Select a search field and enter some search terms for the data you are trying to locate. Click on Index to display the terms dict

Field		Expression
Text words	Index	
Title words	Index	
ISBN/ISSN	Index	
Signature	Index	
Author	Index	
Institutional author	Index	
Name of meeting	Index	
Title	Index	
Title of series	Index	
Subjects	Index	
Geographical subjects	Index	
Publisher	Index	
Place of publication	Index	
Control number	Index	
---	Index	

Since for each search option (row) a link to consult the index for that option is available, we recommend using this as it will not only give immediately an idea of the number of occurrences in the database (not the same as, but approaching, the number of 'MFNs' retrieved) and allow selecting neigboring entries and strange characters or spellings. Searching this form is done as explained in the 'advanced search interface' section of the OPAC, which indeed uses the same form.



Use the Ctrl or Shift keys to select more than one term.;

PALMER, M. D. (1)
PEGO FILHO, BOLIVAR. (1)
PEREIRA, FRANCISCO. (1)
PEYTARD, JEAN. (1)
POTTER, GAVIN. (1)
PREVIN, ANDRE, (1)
RAVEL, MAURICE, (4)
ROWE, WILLIAM (1)
RUBINICH, LUCAS. (1)
SAILLANT, FRANCINE. (1)
SANDLER, COREY. (1)
SCREPANTI, ERNESTO, (1)
SEYBOLD, PATRICIA B. (1)
SHIBATA, KEI, (1)
SIEGEL, DAVID. (1)

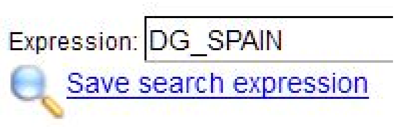
To advance to a specific term, type the first few letters palm and click on Continue .

At the end of your selection, click on Search in order to execute the search with the term(s) selected from this dictionary

Send selected terms Will copy your selection to the search form so you can continue with your search formulation.

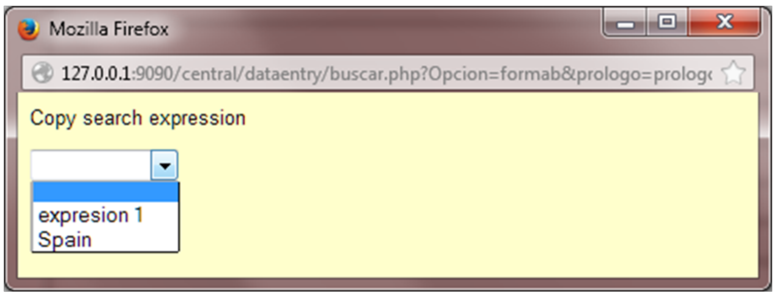
In this illustration we jumped to the section starting with the author 'PALMER' by typing in 'palm' and clicking on 'Continue', then we selected this author, and by clicking Ctrl-click on Previn, André and Shift-click on Ravel, we added these authors into the search set. One can immediately perform the search with the selected entries, of send them to the search form, which allows adding more or elaborating the search with more criteria, e.g. searching other fields with Boolean combinations.

After having searched, by clicking on the 'search' link from the index or the magnifying glass in the search form, the results are displayed but also the option is given to save the actual search expression :



Expression: DG_SPAIN

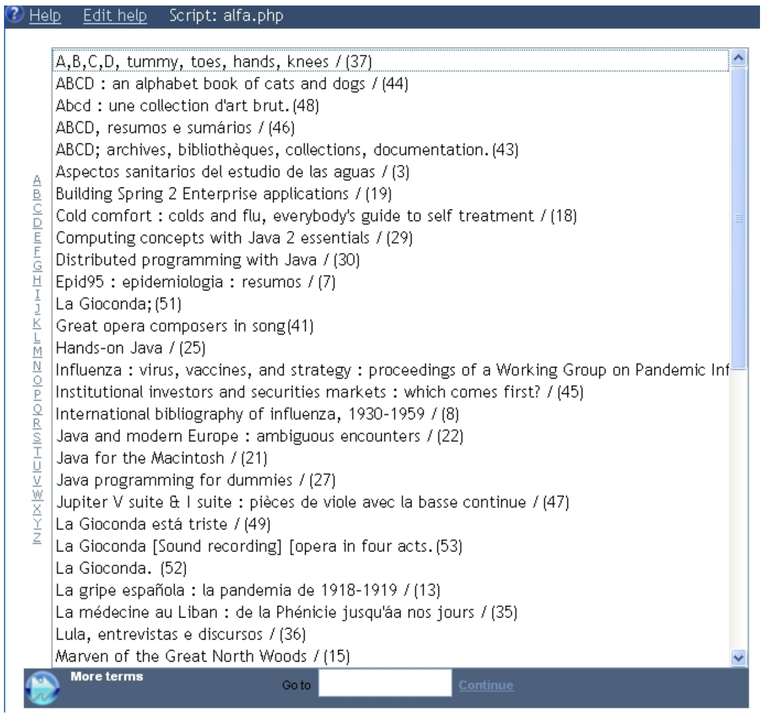
The user will be prompted to assign a 'name' to each search expression for easy retrieval and re-use later on (in the search form). In the search form the magnifying glass at the right side will show - in a small sub-window - the list of saved searches for that database, as shown here :



Mozilla Firefox
127.0.0.1:9090/central/dataentry/buscar.php?Opcion=formab&prologo=prologo
Copy search expression
expresion 1
Spain

- browsing records with the AtoZ selection tool

Clicking on this button will display another, small window in which all records of the database are listed according to the field identified as the 'Identifier' field in the database definition table (3rd column 'I'), mostly the title field e.g. in bibliographic databases. In this list each alphabetic section can be clicked on, and after clicking on the relevant line, the record referenced to by this line will be automatically presented in the main window, ready for e.g. editing. This nice tool extends, as we could put it, ABCD all the way up to Z !



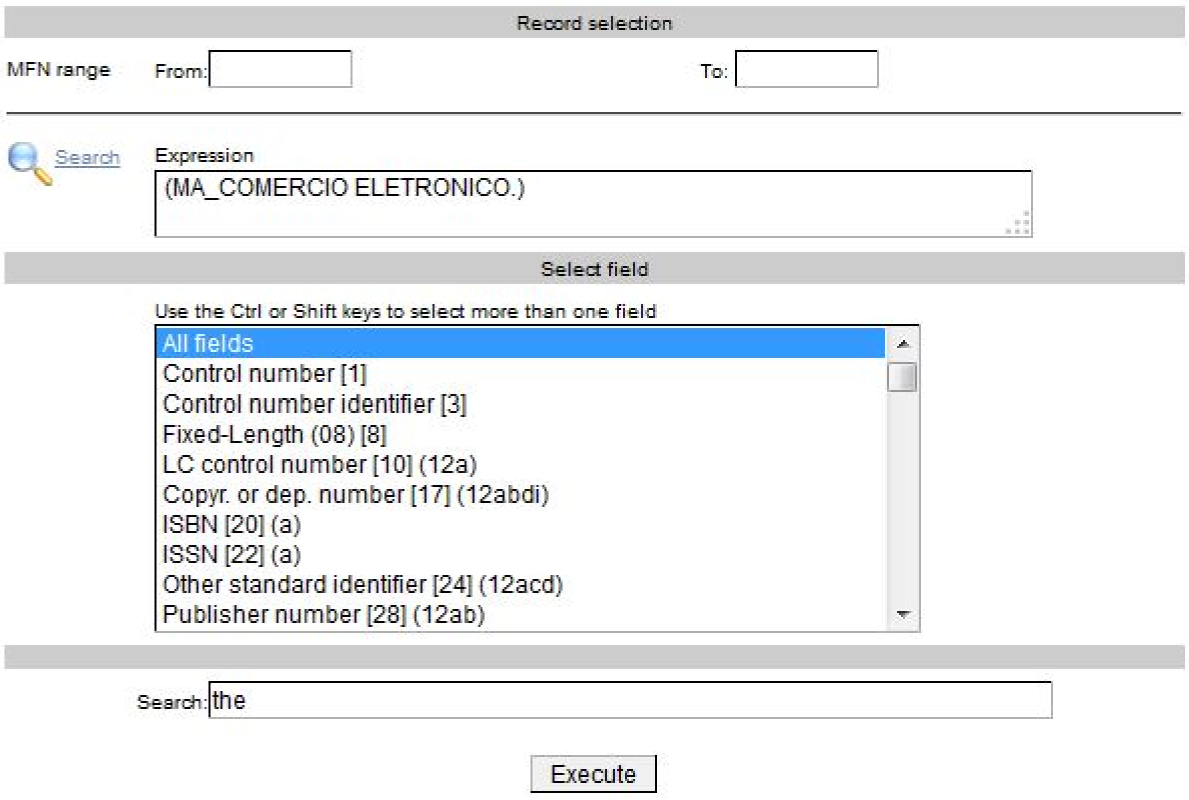
```
A,B,C,D, tummy, toes, hands, knees / (37)
ABCD : an alphabet book of cats and dogs / (44)
Abcd : une collection d'art brut. (48)
ABCD, resumos e sumários / (46)
ABCD; archives, bibliothèques, collections, documentation. (43)
Aspectos sanitarios del estudio de las aguas / (3)
Building Spring 2 Enterprise applications / (19)
Cold comfort : colds and flu, everybody's guide to self treatment / (18)
Computing concepts with Java 2 essentials / (29)
Distributed programming with Java / (30)
Epid95 : epidemiologia : resumos / (7)
La Gioconda; (51)
Great opera composers in song(41)
Hands-on Java / (25)
Influenza : virus, vaccines, and strategy : proceedings of a Working Group on Pandemic Inf
Institutional investors and securities markets : which comes first? / (45)
International bibliography of influenza, 1930-1959 / (8)
Java and modern Europe : ambiguous encounters / (22)
Java for the Macintosh / (21)
Java programming for dummies / (27)
Jupiter V suite & I suite : pièces de viole avec la basse continue / (47)
La Gioconda está triste / (49)
La Gioconda [Sound recording] [opera in four acts. (53)
La Gioconda. (52)
La gripe española : la pandemia de 1918-1919 / (13)
La médecine au Liban : de la Phénicie jusqu'àa nos jours / (35)
Lula, entrevistas e discursos / (36)
Marven of the Great North Woods / (15)
```

- 'free text' searching, clicking on the icon (new as from v1.2t) :

Differently from the standard search which is fully based on ISIS' strong indexing mechanisms, this option simply scans the contents of all records (within a given range) and will select as hits those records in which the search text appears. This option allows searching for anything, also non-indexed parts of records, but is by necessity relatively slow. Slow, because the whole range has to be scanned character by character and only at the end of the whole scan a result will be displayed. For this reason the interface will ask for a range or a previous search result to be identified as the subset to be searched within, and when displaying the results the operator can proceed with a next batch (or range).

The 'search' icon after the range-boxes allows to limit the search, instead of within a given MFN-range, to a given search result, which can be identified by its expression. Clicking on the hyperlink will open the advanced search grid to allow to formulate (not 'execute') such a preliminary search first by 'sending' the selected search expression to the form. E.g. one could use this option to first isolate all publications on a certain theme, then to perform a search on a specific word within that thematic sub-collection of your database.

In th example screenshot here we show a search within the search set on 'electronic business' (comercio eletron-ico) of a database (MARC21), to search for the string 'the' in all fields.


Record selection

MFN range From: To:

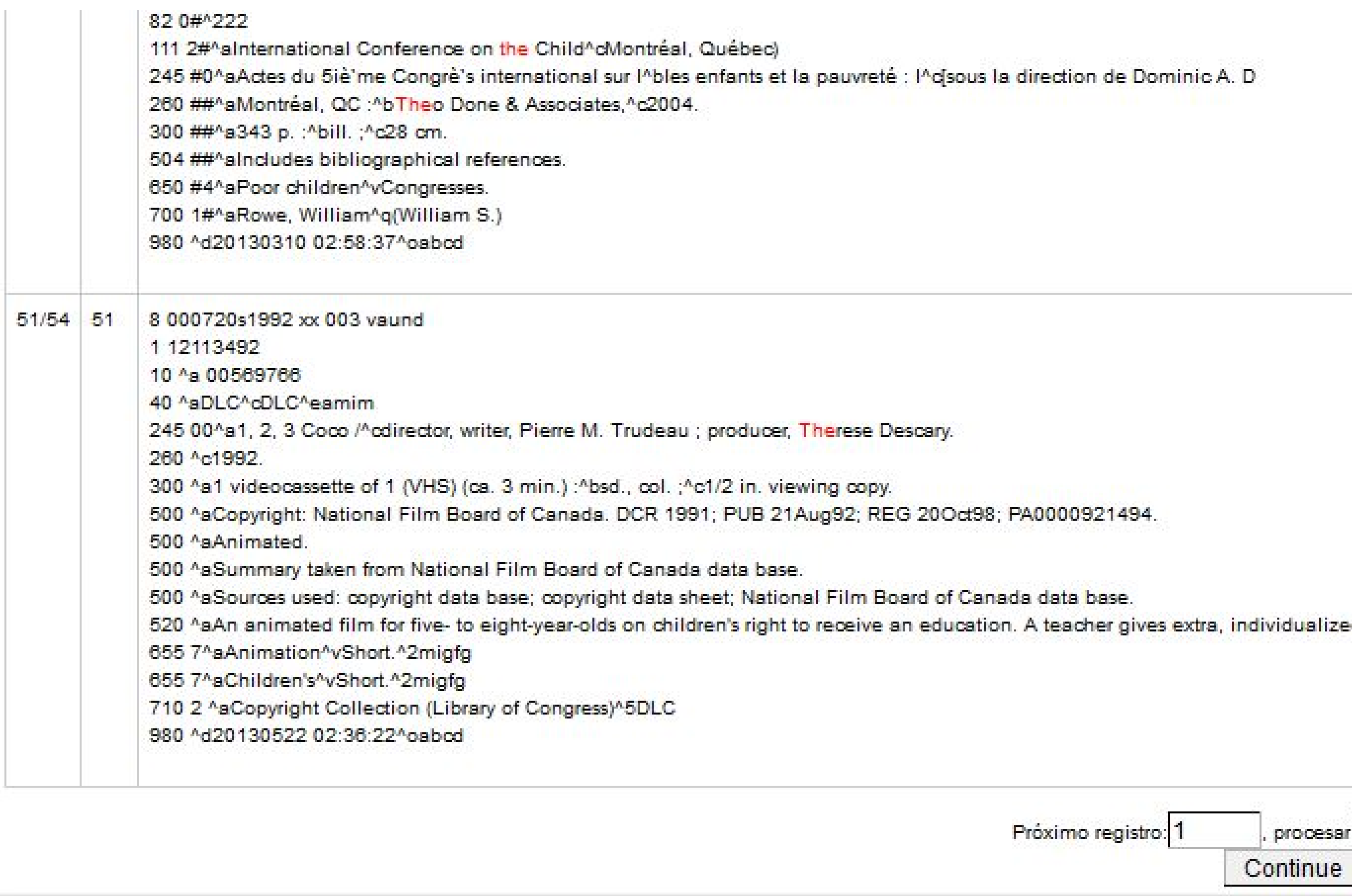
Search Expression
(MA_COMERCIO ELETRONICO.)

Select field

Use the Ctrl or Shift keys to select more than one field
All fields
Control number [1]
Control number identifier [3]
Fixed-Length (08) [8]
LC control number [10] (12a)
Copyr. or dep. number [17] (12abdi)
ISBN [20] (a)
ISSN [22] (a)
Other standard identifier [24] (12acd)
Publisher number [28] (12ab)

Search: the

The resulting records will be clearly shown with their MFN and the search text highlighted (in red), but no further direct action can be applied to them. The power of this search option therefore is only to search anything anywhere and identify the MFN's. Such search also retrieves the search key as substrings from the text in the records, as can be seen from the illustration here : the first time 'the' is the article itself, the next two times in the example result 'the' is a substring of in fact a name starting with it.



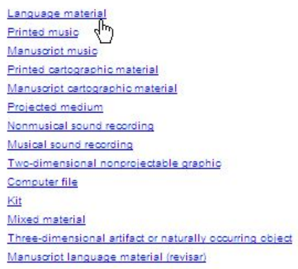
```
              82 0#^222
              111 2#^aInternational Conference on the Child^cMontréal, Québec)
              245 #0^aActes du 5iè'me Congrè's international sur l^bles enfants et la pauvreté : l^c[sous la direction de Dominic A. D
              260 ##^aMontréal, QC :^bTheo Done & Associates,^c2004.
              300 ##^a343 p. :^bill. ;^c28 cm.
              504 ##^aIncludes bibliographical references.
              650 #4^aPoor children^vCongresses.
              700 1#^aRowe, William^q(William S.)
              980 ^d20130310 02:58:37^oabcd

51/54   51    8 000720s1992 xx 003 vaund
              1 12113492
              10 ^a 00569766
              40 ^aDLC^cDLC^eamim
              245 00^a1, 2, 3 Coco /^cdirector, writer, Pierre M. Trudeau ; producer, Therese Descary.
              260 ^c1992.
              300 ^a1 videocassette of 1 (VHS) (ca. 3 min.) :^bsd., col. ;^c1/2 in. viewing copy.
              500 ^aCopyright: National Film Board of Canada. DCR 1991; PUB 21Aug92; REG 20Oct98; PA0000921494.
              500 ^aAnimated.
              500 ^aSummary taken from National Film Board of Canada data base.
              500 ^aSources used: copyright data base; copyright data sheet; National Film Board of Canada data base.
              520 ^aAn animated film for five- to eight-year-olds on children's right to receive an education. A teacher gives extra, individualize
              655 7^aAnimation^vShort.^2migfg
              655 7^aChildren's^vShort.^2migfg
              710 2 ^aCopyright Collection (Library of Congress)^5DLC
              980 ^d20130522 02:36:22^oabcd
```

3.3. Using the editing forms

When clicking on the first icon of the record-toolbar, the record will be presented in an editing form. The same form - but empty - will be used for creation or copy of a new record (from the database toolbar).

Note

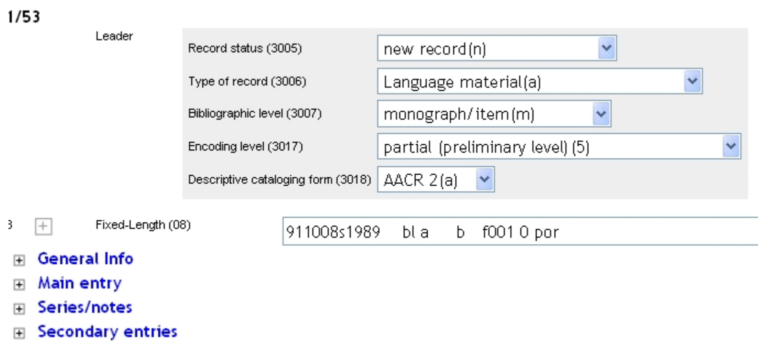
In the case of MARC21 databases, whenever editing a new record, first a selection has to be made from a list of possible record types :



Language material
Printed music
Manuscript music
Printed cartographic material
Manuscript cartographic material
Projected medium
Nonmusical sound recording
Musical sound recording
Two-dimensional nonprojectable graphic
Computer file
Kit
Mixed material
Three-dimensional artifact or naturally occurring object
Manuscript language material (revisar)

Selecting one of these (i.e. clicking on their link) will subsequently invoke the right worksheet with some pre-defined values (e.g. for the 'fixed format' MARC fields).

The MARC-21 edit form, with its long and complicated structure, will be presented in a special way : form sections can be 'collapsed' or 'unfolded' by clicking on the +-link of the according section. The details of the section will be shown or hidden.


1/53

Leader

Record status (3005)	new record(n)
Type of record (3006)	Language material(a)
Bibliographic level (3007)	monograph/item(m)
Encoding level (3017)	partial (preliminary level) (5)
Descriptive cataloging form (3018)	AACR 2(a)

3 Fixed-Length (08) 911008s1989 bl a b f001 0 por

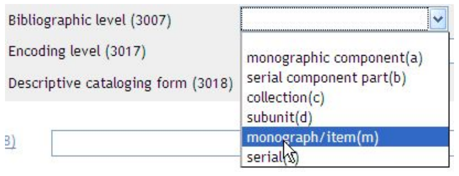
General remark : depending on how the Field Definition Table of ABCD, which in contrast to the one of WinISIS also defines worksheet features, was designed, the worksheet can appear as divided into sections with direct-access links (buttons) and buttons to return to the beginning of the worksheet for faster navigation *within* the worksheet.

Fields for editing can have one of the following formats, each with their own way of dealing with them, briefly explained below as follows :

- a simple editable field, as in the ISBN- field below, where you can only type in a value (e.g. ' ^a8570251270') yourself

20 ISBN ^a8570251270

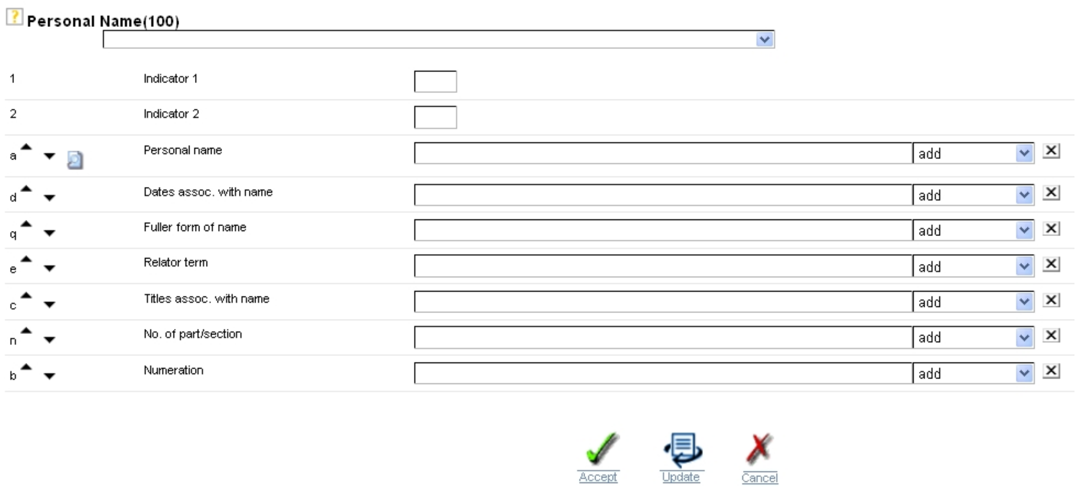
- a field with a menu from which a single option has to be selected, as e.g. 'monograph' as the bibliographic level :



- a substructured field, with a button to give access to a separate window in which all substructures (e.g. subfields or parts of the MARC-fixed header field) can individually entered.

100 Personal Name

Clicking on the icon will invoke a new window in which all substructures are presented individually, e.g. in the for MARC field 'Edition' (250) there are subfields for the edition 'info proper and one for the 'additional information' :


Personal Name(100)

1	Indicator 1		
2	Indicator 2		
a	Personal name		add
d	Dates assoc. with name		add
q	Fuller form of name		add
e	Relator term		add
c	Titles assoc. with name		add
n	No. of part/section		add
b	Numeration		add

In this window each substructure value can be entered into its own edit box, repeats can be added with the 'add' option and the whole 'group' of substructures can be repeated by pressing the button, which will add one series of substructures as a new occurrence of the same field.

By clicking on the button Accept the values will be placed into the field box with the proper substructure delimiters added.

By clicking on the button Update the edited values will be entered in the field box of the main editing window. Needless to say that the 'Cancel' button will just close the 'substructure' window *without* adding any values into the field.

It is possible to change the sequence of the repeats of substructures by using the 'up' and 'down' buttons ▲ ▼ to move up resp. down the actually selected occurrence

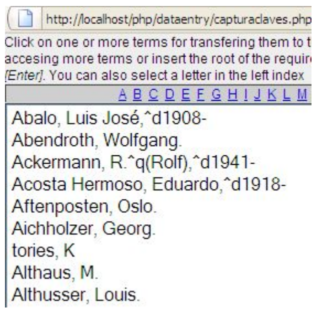
Finally one can also delete one occurrence of a substructured field by clicking on the 'delete' button ☒

Experienced data entry experts will often by-pass this additional support from the ABCD-interface and type in the substructures of the field with their appropriate delimitors directly into the main worksheet editor, which is perfectly possible : the extra window is just an extra support indeed !

- a field with a picklist, indicated by the icon :

In a field with 'authority control' by predefined terms listed in a separate box, clicking on this icon will open the list in a smaller window :

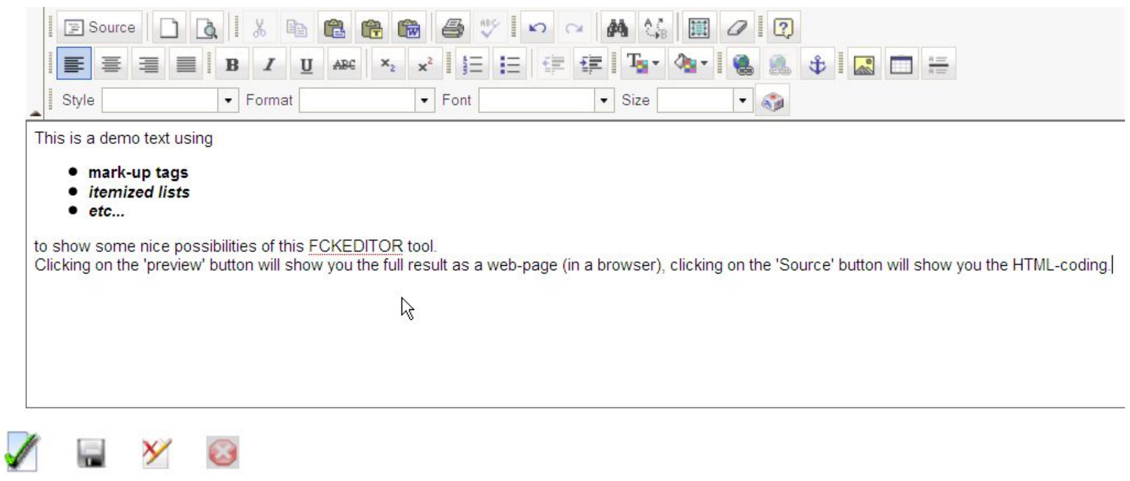
Depending on the definition of this picklist (see the dedicated paragraph in the section on picklist definition of the database definition functions), only one or more items can be selected. Needless to say that the items listed will be precisely defined as well in that Field Definition Table set of colums on picklists, e.g. by defining a common prefix with which the terms have been indexed.


http://localhost/php/dataentry/capturaclaves.php
Click on one or more terms for transfering them to t
accesing more terms or insert the root of the requir
[Enter]. You can also select a letter in the left index
A B C D E F G H I J K L M
Abalo, Luis José,^d1908-
Abendroth, Wolfgang.
Ackermann, R.^q(Rolf),^d1941-
Acosta Hermoso, Eduardo,^d1918-
Aftenposten, Oslo.
Aichholzer, Georg.
tories, K
Althaus, M.
Althusser, Louis.

In this list you can navigate by using the alphabetic control or select one (or more) items from the list by clicking, Shift-clicking or Ctrl-clicking (or dragging the mouse) for multiple selection.

After you are ready with selecting one or more terms, click on 'Continue' (below within the picklist window) to transfer your selection to the main edit form.

- a very special, interesting field is a 'Rich Text Editor' field, which shows up as follows :



This is a demo text using

- **mark-up tags**
- ***itemized lists***
- ***etc...***

to show some nice possibilities of this FCKEDITOR tool.
Clicking on the 'preview' button will show you the full result as a web-page (in a browser), clicking on the 'Source' button will show you the HTML-coding.

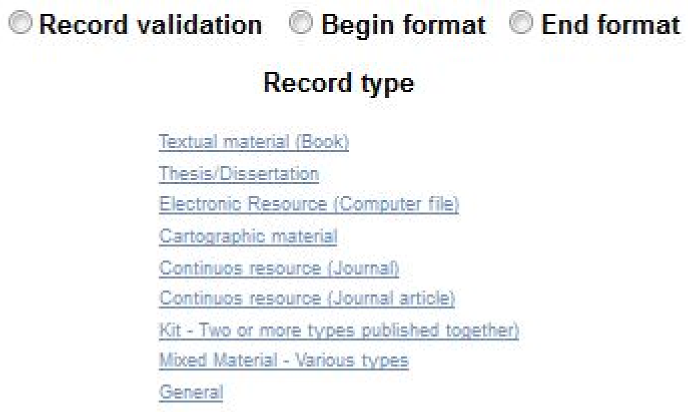
As can be seen, the field is a larger editing box with a series of icons above as in a real text editor with many text-editing features, such as itemized lists, boldface or italics etc. Such a field in ABCD is edited with a special add-on JavaScript tool, i.e. 'fckeditor', which allows to use the icons in order to create the according HTML-tags in the text of the field value. Whenever this value is shown in a WWW-environment (as ABCD itself, or e.g. in J-ISIS), the HTML-tags will be interpreted as such and result in graphical effects, as this is the meaning of HTML-tags.

It is also possible to 'paste' into such a field text obtained from another document (e.g. a Word-document), using a conversion mechanism (to filter out all non-text elements) provided by ABCD/PHP.

3.4. Record and field validation

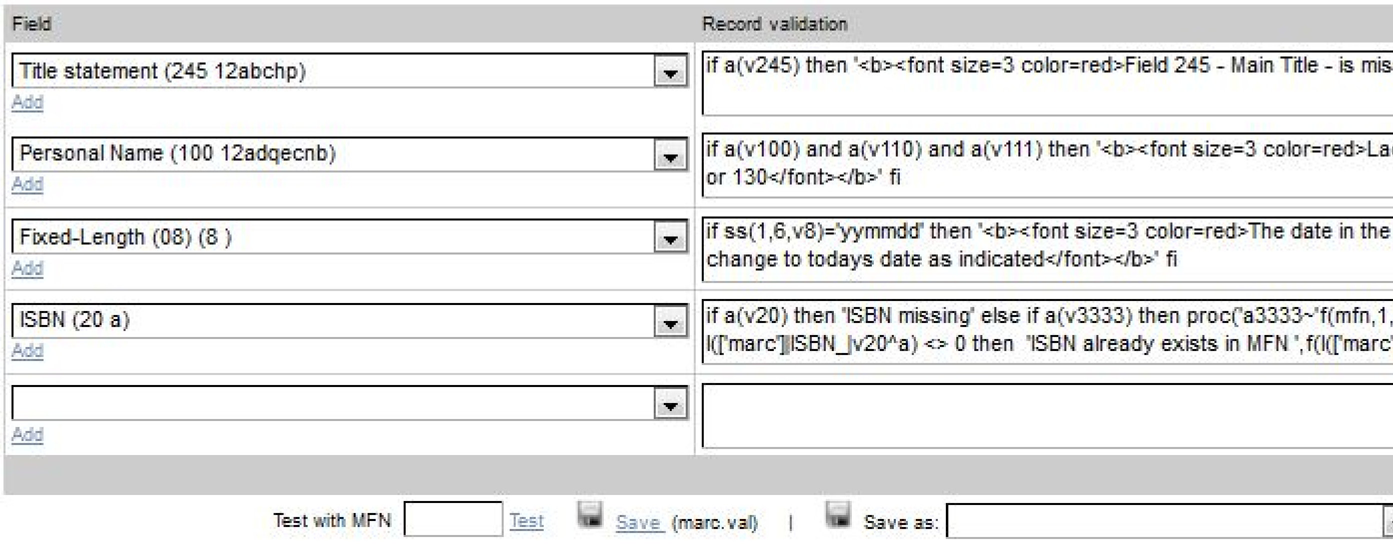
ABCD allows the system manager to design and implement validation statements for quality control at data entry stages. Such statements are written in the ISIS Formatting Language - we would say 'of course'..! See the section on this topic above - as it is an option of the main Central 'Update Database Definitions' menu. In the record-toolbar this function can be executed by clicking the icon ✓ to actually check the current record according to the validation defined. The resulting messages, if any, will be shown in a small separate window.

Validation can be performed at three stages : record-editing, opening a record ('Begin record') and storing of the record ('End record'). For each stage validations are to be defined for a specific field, even if within the PFT-statement itself other fields can be referenced. Moreover, each validation set can be specified either at the 'general' level (= the FDT) or for each worksheet (at the record-type level).


Record validation
Begin format
End format
Record type
Textual material (Book)
Thesis/Dissertation
Electronic Resource (Computer file)
Cartographic material
Continuos resource (Journal)
Continuos resource (Journal article)
Kit - Two or more types published together)
Mixed Material - Various types
General

The first validation in the illustration below is quite simple : if v245 (title in MARC21) is absent ('if a(v245)'), then display in red an error message to warn that the title is missing. It is marked as a 'fatal' error, meaning the record cannot be stored without correction. More or less the same check is done but now on the different author fields, and the error is just a warning, so it can still be stored without authors. The third validation checks if the first 6 characters of v8 are still the default text 'yymmdd', if so the operator is told to change the date according to today's date.

As from v1.2t ABCD includes a more advanced example of field validation which we will explain here. We refer to the part on v20 (ISBN), the last one defined in the illustration.



Field	Record validation
Title statement (245 12abchp)	if a(v245) then '<b><font size=3 color=red>Field 245 - Main Title - is mis
Personal Name (100 12adqecnb)	if a(v100) and a(v110) and a(v111) then '<b><font size=3 color=red>La or 130</font></b>' fi
Fixed-Length (08) (8)	if ss(1,6,v8)='yymmdd' then '<b><font size=3 color=red>The date in the change to todays date as indicated</font></b>' fi
ISBN (20 a)	if a(v20) then 'ISBN missing' else if a(v3333) then proc('a3333~'f(mfn,1, l(['marc']ISBN_|v20^a) <> 0 then 'ISBN already exists in MFN ',f(l(['marc'

The ISBN 'duplicate check' has the following parts :

- if the ISBN is missing, just give a normal warning
- if the internal field 3333 is missing then add the current MFN value into it (to make sure there is a value for v3333)
- if that value for the internal field 3333 is zero, meaning it is a new record as the MFN is not yet assigned, then (and only then) perform the remaining test
- the value of v20^a preceded by the prefix 'ISBN_' is Looked up in the marc-IF, and if already found ('<>0'), then it means the ISBN is already existing and accordingly a message will be produced adding in which MFN the ISBN has been found (by looking it up again and using the resulting MFN).

Note

By clicking on the icon 📝 such longer validation scripts can be edited inside a separate editing window.

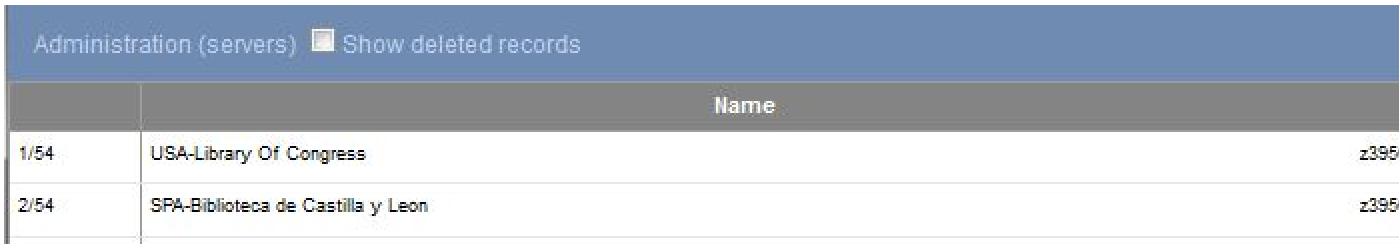
Based on this 'model' it is easy to create similar validations to avoid duplication of e.g. barcodes in the COPIES-database, or avoiding adding a user twice in the USERS-database. Rather than pre-defining all these checks, ABCD prefers to offer the models and possibilities to system managers to implement them where and when desired, notwithstanding the observation that such validation statements can easily be exchanged using other communication means, e.g. the ISIS discussion list or other ABCD-related fora.

3.5. Shared cataloging through Z39.50

Z39.50 is a protocol which allows 'sharing' records (especially in the MARC-format) by first identifying them through a search with one of a series of Z39.50 servers - like Library of Congress in Washington or British Library in London - and then downloading the records into the local system. Here is how it works in ABCD :

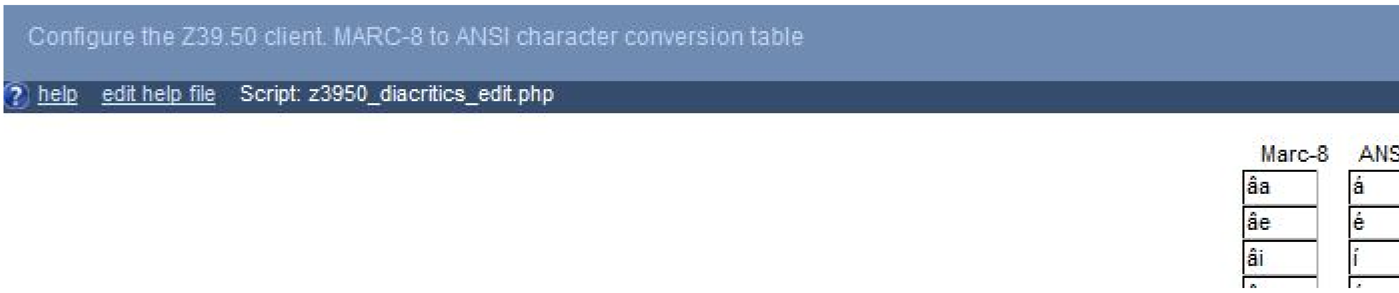
3.5.1. Configuring the Z39.50 of ABCD

Before you can use this function, some configuration has to be set in order to define the Z39.50 hosts you can or want to use and how to reach them. ABCD provides a menu option (main Central menu) for this configuration. The available servers are listed with their main access parameters (login data will be added in a later version) and in the same editing screen servers can be added or deleted :


Administration (servers) Show deleted records

	Name	
1/54	USA-Library Of Congress	z395
2/54	SPA-Biblioteca de Castilla y Leon	z395

In the same Z39.50 client configuration option, also tables can be created/edited to automatically convert incoming characters into others. We show part of the MARC-8 characterset to ANSI conversion table :


Configure the Z39.50 client. MARC-8 to ANSI character conversion table

help edit help file Script: z3950_diacritics_edit.php

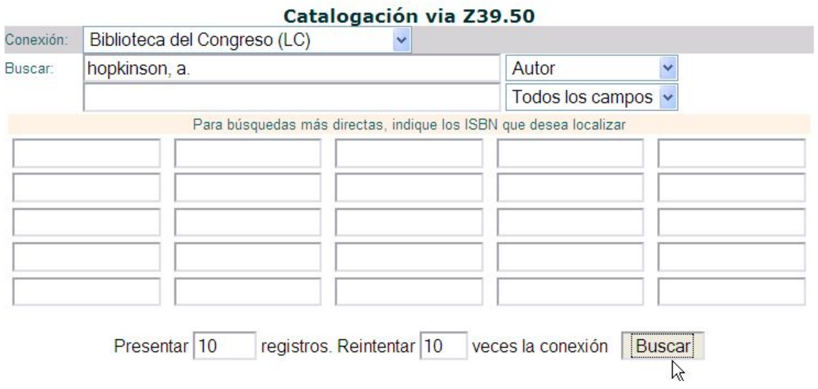
Marc-8	ANS
âa	á
âe	é
âi	í

Finally a 'test'-option allows to test the Z39.50 servers with the given configuration.

3.5.2. Using the Z39.50 tool of ABCD

The following steps have to be taken for actually using the tool :

1. Click on the Z39.50 icon in the cataloging toolbar, invoking a screen allowing the selection of a Z39.50 host and either a search statement or one or more ISBN's to be sent as to identify records for downloading :


Catalogación via Z39.50

Conexión: Biblioteca del Congreso (LC)

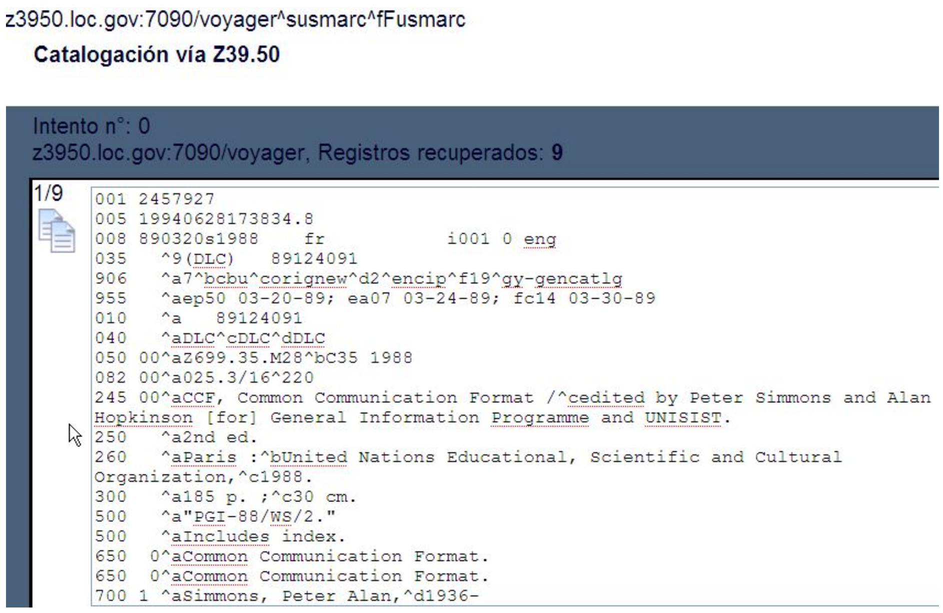
Buscar: hopkinson, a. — Autor

Todos los campos

Para búsquedas más directas, indique los ISBN que desea localizar

Presentar 10 registros. Reintentar 10 veces la conexión

2. After having clicked on the button 'Buscar' (or search), if the submitted request is successful (meaning : the server accepted your request and identified one or more records indeed, complying with your search criteria), a new browser window will present the list of results with a 'copy to database' icon for each individual record :



z3950.loc.gov:7090/voyager^susmarc^fFusmarc

Catalogación vía Z39.50

Intento n°: 0
z3950.loc.gov:7090/voyager, Registros recuperados: **9**

1/9

```
001 2457927
005 19940628173834.8
008 890320s1988    fr           i001 0 eng
035   ^9(DLC)   89124091
906   ^a7^bcbu^corignew^d2^encip^f19^gy-gencatlg
955   ^aep50 03-20-89; ea07 03-24-89; fc14 03-30-89
010   ^a   89124091
040   ^aDLC^cDLC^dDLC
050 00^aZ699.35.M28^bC35 1988
082 00^a025.3/16^220
245 00^aCCF, Common Communication Format /^cedited by Peter Simmons and Alan
Hopkinson [for] General Information Programme and UNISIST.
250   ^a2nd ed.
260   ^aParis :^bUnited Nations Educational, Scientific and Cultural
Organization,^c1988.
300   ^a185 p. ;^c30 cm.
500   ^a"PGI-88/WS/2."
500   ^aIncludes index.
650  0^aCommon Communication Format.
650  0^aCommon Communication Format.
700 1 ^aSimmons, Peter Alan,^d1936-
```

3. When clicking on the 'copy-to-database' icon, the selected record will be transferred into your ABCD cataloging screen as the actual record (a new MFN will have been granted), where it can be edited with the edit button if necessary.

   Needless to say that in most cases such records are high quality MARC-records, so this shared cataloging has many advantages, e.g. saving time (if sufficient bandwidth for the communication with the server is available) and improving quality of your catalogue. If your catalogue does not use the MARC format however you might need to first prepare a conversion mechanism before loading records into your own database. This technique is discussed elsewhere in this document.

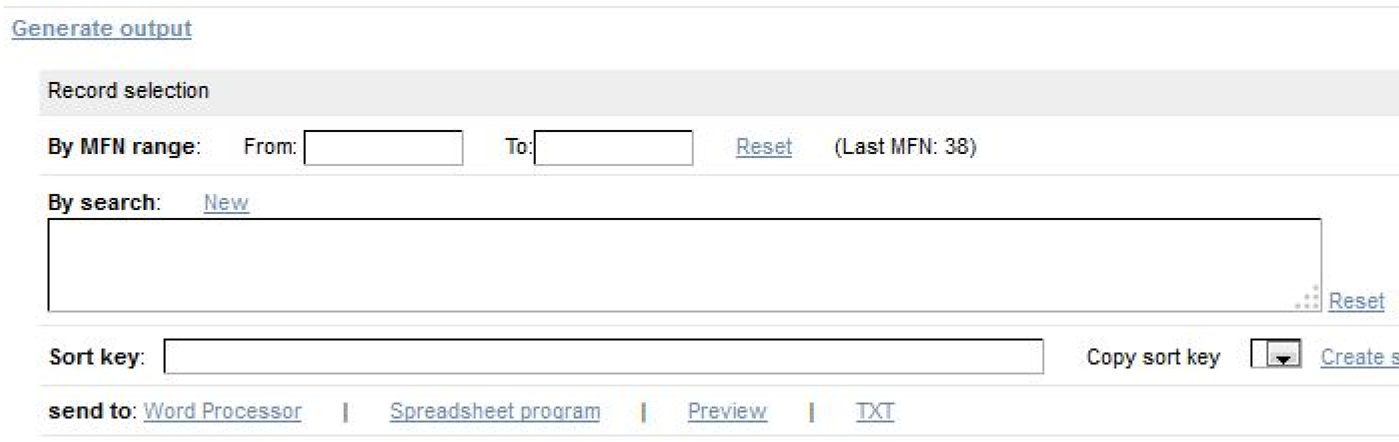
4. When the Z39.50 icon is clicked not from the database-toolbar but from the record-toolbar, the behaviour will be slightly different : select the actual record (preferably by its ISBN) and when available from the selected Z39.50 server, only the missing fields in the actual record will be added, in order to complete the bibliographic record.

3.6. Default values

This is a simple function of the database-toolbar : the edit-form will be presented and if so desired default values can be entered (or deleted) into this form.

3.7. Reports (printing)

As explained earlier, ABCD reports are generated by ISIS Print Formats and therefore the interface for the creation of PFT's is re-used here. In the future ABCD will probably offer more pre-defined frequently-used or typically PFT's, e.g. for listing of overdue loans etc. - which we will discuss when dealing with the Circulation module. The important observation here is that reports are actually print-formats (PFT's) which will be used to produce lists of records, and as such these can be created/used for any database and any use. The actual use of a report is by entering in to the 'generating output' part of the PFT-creation interface :


Generate output

Record selection

By MFN range: From: To: Reset (Last MFN: 38)

By search: New

Reset

Sort key: Copy sort key Create s

send to: Word Processor | Spreadsheet program | Preview | TXT

As can be seen, reports can run either over record-ranges (or whole databases), or over a search result set. Sort keys can be created and used when generating output : in themselves they are 'extraction formats' of (obviously) PFT's again which will be used to sort the output. Finally output can be sent to a word-processor or a spreadsheet in appropriate format, as plain text in a file or simply to the screen itself, called 'preview'.

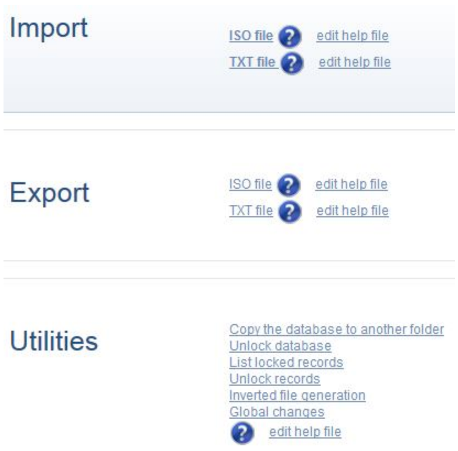
3.8. Utilities

There are two menu's with utilities : specific database maintenance utilities from the cataloger's toolbar and system utilities as an option of the main Central menu.

We discuss them separately here.

3.8.1. Cataloging Utilities

The utilities as the 'toolbox' option from the main toolbar offered in the Data-Entry module are grouped as 'import', 'export' or 'utilities proper' :


Import

ISO file ? edit help file

TXT file ? edit help file

Export

ISO file ? edit help file

TXT file ? edit help file

Utilities

Copy the database to another folder

Unlock database

List locked records

Unlock records

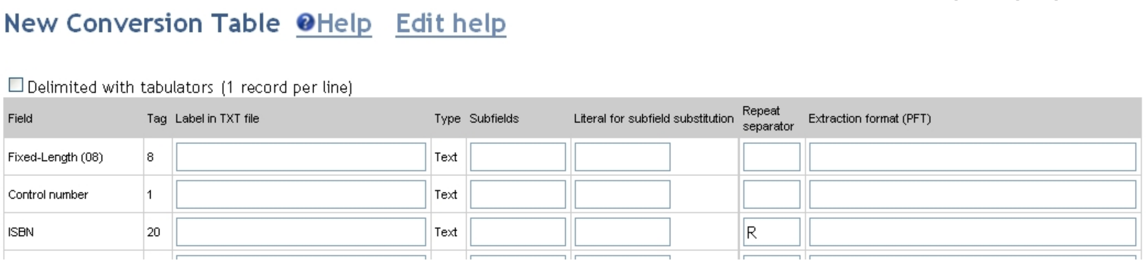
Inverted file generation

Global changes

? edit help file

1. Import

   a. Text Import : text-based structured files can be imported by defining a conversion table, which will take the label for each (sub-)field (or the literals used to separate subfields) in the input-file, the 'repeat-separator' used within one field for different occurrences and the extraction format in ISIS-Formatting Language. :


New Conversion Table ?Help Edit help

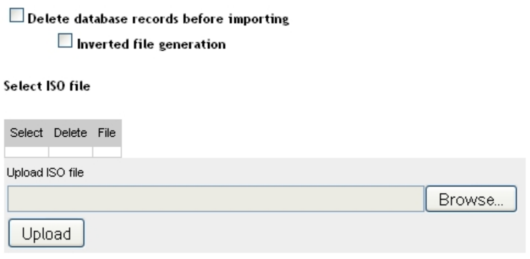
☐ Delimited with tabulators (1 record per line)

Field	Tag	Label in TXT file	Type	Subfields	Literal for subfield substitution	Repeat separator	Extraction format (PFT)
Fixed-Length (08)	8		Text				
Control number	1		Text				
ISBN	20		Text			R	

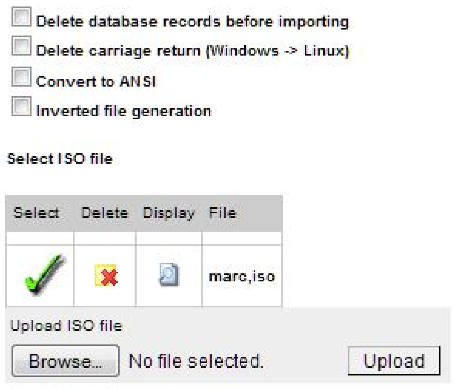
If the text-input file is actually a 'TAB-delimited' file (a 'comma-separated values' file with TAB's in stead of comma's), activate the button and ABCD will simply - without a need to identify each field or column - import each value into a separated sequential field (meaning : the first value or column will go into field 001 etc.).

b. ISO Import

ISO-files can be imported into an ABCD-database in 2 steps : first the ISO-file has to be uploaded by browsing to it :


☐ Delete database records before importing
☐ Inverted file generation
Select ISO file
Select
Delete
File
Upload ISO file
Browse...
Upload

Then the uploaded ISO file will be listed and can be actually imported with the green 'activate' button, after which ABCD will actually import the ISO-records into the actual database (using the CISIS tool for this purpose), meaning that in case of high volumes the Apache webserver might issue a time-out error - then use the command-line tool for this ! :


☐ Delete database records before importing
☐ Delete carriage return (Windows -> Linux)
☐ Convert to ANSI
☐ Inverted file generation
Select ISO file
Select
Delete
Display
File
marc.iso
Upload ISO file
Browse...
No file selected.
Upload

The screenshot shows some extra options : whether or not to delete (empty) the database before importing, change the Linux-end-of-line-character to Windows, convert to ANSI (when the ISO is taken from a DOS-ISIS database) and whether or not to index the database during import - which will then extend the time needed of course.

Note

Indexing during import is not possible when the option 'Convert to ANSI' is activated.

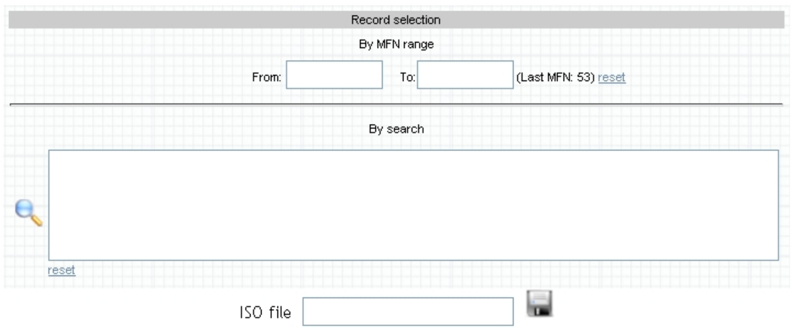
The 'display' option will show the first 100 records in order to check their contents and format.

The actual import will start by clicking on the 'select' icon and confirming the operation to start indeed.

2. Export

a. Export to ISO

This function simply asks for a selection of records (by MFN-range or search-expression) to export and a name of the export ISO-file :

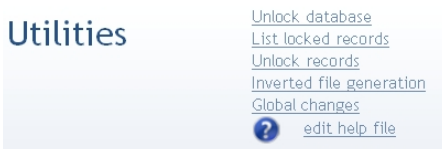


b. Export to TXT

This is the reverse function from the TXT-import, using the same conversion table format, but logically doing the reverse operation on fields. By activating the 'delimited by tabs' option the export will be actually a CSV file which can be imported into spreadsheets etc.

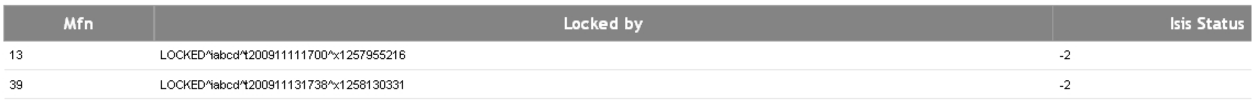
3. Other database utilities

In addition to importing and exporting databases from and to different formats, ABCD also needs to provide some tools for the database-administrator, which we will briefly discuss here :



- unlock the database : in case a database got locked (as part of the multi-user protection) without having the chance of being unlocked, e.g. in the case of a sudden power-cut, the database can be manually unlocked. ABCD will simply report on the status after unlocking.

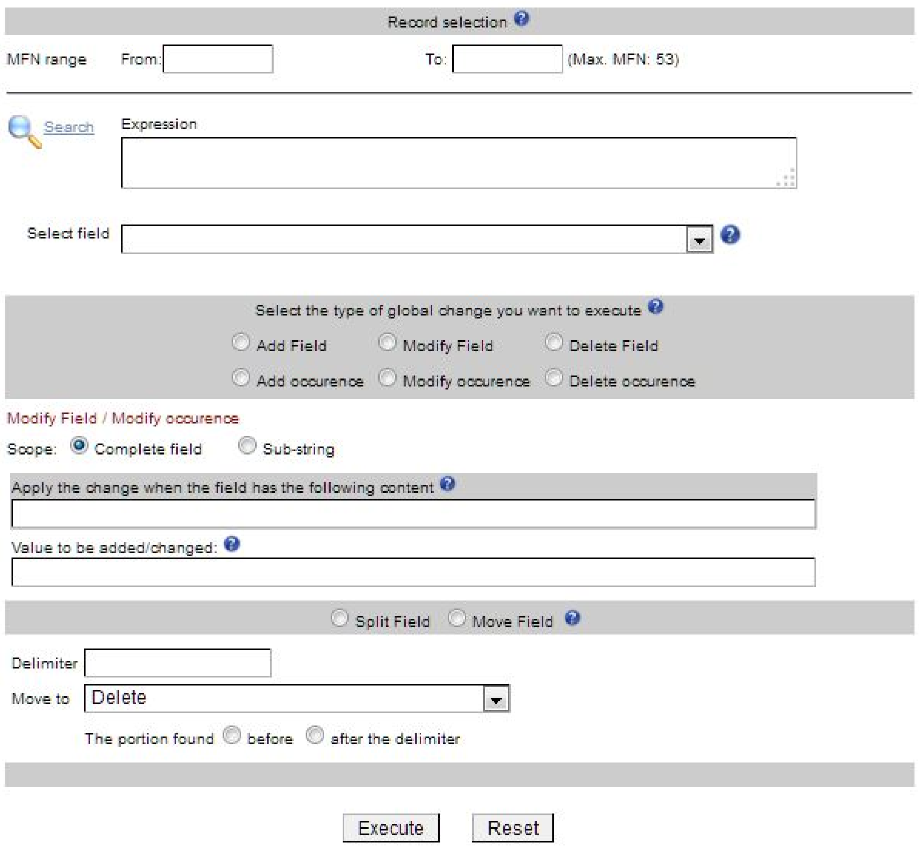
- List locked records : ABCD will list the records which are locked (if any) with their status :



Mfn	Locked by	Isis Status
13	LOCKED^iabcd^t200911111700^x1257955216	-2
39	LOCKED^iabcd^t200911131738^x1258130331	-2

- Inverted File generation : in case structural changes have been made to the FST, the system administrator should re-index the database to apply the new FST-definition onto all the records of the database (not only on the newly edited ones, which will be done automatically). If the set of records is not too large - to be seen in function also of the complexity of the FST and the average size of the records - the full Inverted File generation can be done from within ABCD here. In case of larger databases this should be done by a command-line operation with one of the CISIS tools, e.g. 'mx MARC fst=@marc.fst fullinv=marc now -all tell=1000'.

- Global changes : as in WinISIS (and even provided in the old days of CDS/ISIS for DOS with additional ISIS/Pascal programs) changes can be edited semi-automatically over the full database or a range of records (identified by range or search-expression). The following interface illustrates the possibilities here :



Record selection

MFN range From: To: (Max. MFN: 53)

Search Expression

Select field

Select the type of global change you want to execute

Add Field Modify Field Delete Field

Add occurence Modify occurence Delete occurence

Modify Field / Modify occurence

Scope: [✓ Complete field] Sub-string

Apply the change when the field has the following content

Value to be added/changed:

Split Field Move Field

Delimiter

Move to: Delete

The portion found before after the delimiter

Execute Reset

As can be seen, the first part of the interface allows to define the range of records to operate on by either range or a search-selection; in case of a search one can select search-keys from the dictionary in the standard-search interface of ABCD Central, but not search directly from the dictionary : rather the terms need to be sent to the search form to build the search query which then will be sent to the 'global changes' search-box.

After the definition of the set of records and the specific field to work on, the following options are available :

- Add a field or an occurrence of a repeatable field - a value to be present in the field can be given as a condition to perform the change
- Modify a field or an occurrence of a repeatable field - a value to be present in the field can be given as a condition to perform the change
- Delete a field or an occurrence of a repeatable field - a value to be present in the (occurrence of the) field can be given as a condition to perform the change
- Add an occurrence of a field - a value to be present in the (occurrence of the) field can be given as a condition to perform the change
- Move a field to another : the destination field can be selected from the FDT list of fields
- Split a field into parts : at a defined 'separation character' either the value before or after the delimiter can be moved to another field, which in fact means splitting it

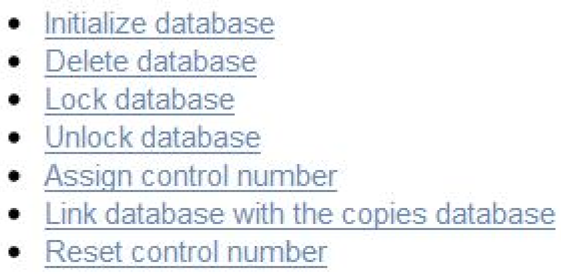
The 'scope' of the operation defines whether when checking for an existing value the whole field or a substring of it is considered.

Once all parameters are defined, the operation can be started; this will run in batches to avoid time-out of the http-protocol and the progress will be reported with the list of all changed MFN's.

- The dedicated HELP-file for these utilities can be edited from this option. The existing text will be given in the HTML-editor and can be edited and/or translated.

3.8.2. System utilities

Somehow confusing might be the other 'utilities' menu but now from the main Central menu, where rather system management is envisaged. let's look at the options there :



- Initialize database
- Delete database
- Lock database
- Unlock database
- Assign control number
- Link database with the copies database
- Reset control number

Needless to underline here that such options will only be available to system administrators (users with the administration profile) as they involve quite dramatic operations like initializing a database (= reset the database to zero records), delete a database, lock or unlock a database. Some extra confirmations will be requested in view of the drasic consequences of these actions. The idea of these options is quite clear in itself and doesn't need much explanation, except for the remark that the (new) option 'unlock database' here is to be used when the database-lock even prevents operators from entering it, so that the unlock option in the toolbar-utilities by definition cannot be used. Locking a database could be necessary when special operations on the database are to be performed, e.g. a full inverted file generation (an option we will add to this menu), loading larger batches of ISO-records (also soon to be added) or any other database operation necessitating 'single user' access by the administrator.

The option 'assign control number' (not to be confused with 'reset control number') allows administrators to automatically create control numbers over a range of records. When clicking on 'Send' here a series of sequential numbers, starting from the given one, will be assigned (= entered into) the range of MFN's. This is a powerful but risky operation, so use with care !

The option 'reset control number' , provided within the 'assign control number' option as in the utilities list itself, simply provides the current CN value in an edit box, so that administrators can change the current 'last number', e.g. to start another range of numbers for a specific cataloging centre.

Finally the option 'Link database with the copies database' actually activates the database (by ticking a checkbox, which tells ABCD putting the character 'Y' as a third column in the file 'bases.dat') for the copies/loans system, meaning that for this database the loanobjects will be created from the copies-database, as distinguished from the (new) option to run the circulation based on identifiers within the catalog records.

For copies management, the objects database must have a unique control number for each record. This field needs to be defined in the FDT with Type: Autoincrement, and indexed in the FST with the prefix: CN_. If there is no Autoincrement input type defined for the database ánd an entry exists in the FST creating terms with the prefix 'CN_', the system will protest whenever trying to create a physical copy ('exemplar') record in the COPIES-database directly from the catalog. More details will be given in the sections later on concerning copy-cataloging and circulation configuration.

3.9. Statistics

This remaining database-toolbar option is the one on Statistics. This option is explained in its own section but can be accessed also from the cataloger's toolbar.

4. Central module : statistics in ABCD

Fully following the basic ABCD-philosophy of being open and flexible (the price for this to be paid being the necessity of quite some 'do-it-yourself' flavor), statistics in ABCD is a function which is based on a series of actions which can be prepared and performed on ANY database and ANY series of values in there. Mostly librarians want to prove their 'performance measurement' qualities by showing statistics on the circulation - as the most 'visible' use of their library, but remember in ABCD statistics can be produced about any database, e.g. the profiles of users (in the USERS-database), shifting collection policy over time in the catalog or even, much less popular : cataloging output by catalogers...

This philosophy of openness results in the following steps to be performed for doing statistics in ABCD :

- define fields as 'variables' to be counted/processed; such fields can be processed first by any PFT statement (the always present ISIS-philosophy !) before yielding values on which the statistical analysis will run
- create with these pre-defined variables either single-dimension or multi-dimension (cross-)tables
- open one of the pre-defined table-definitions
- apply the table on a series of records (MFN's) to get the numerical results
- (optional) present the numerical data in a graphical way
- (optional) output the numerical data to external software for further processing (spreadsheet, document or printer).

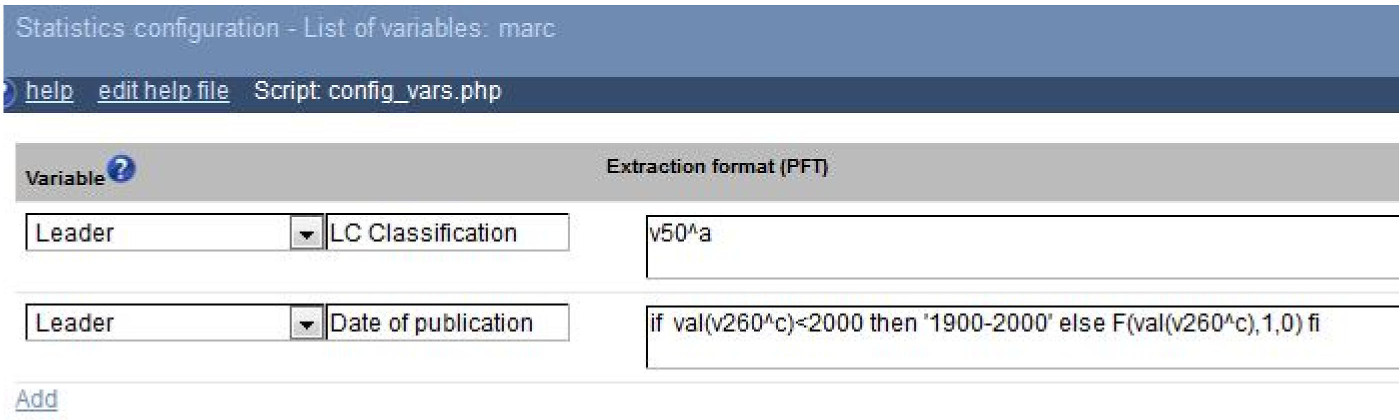
The first 2 steps are presented as 'Statistics Configuration' in the Statistics menu and also appear (at the end) in the list of 'Database Definition' options. We will first deal with this preparatory part here, then briefly explain how to actually run statistics in ABCD.

4.1. Statistics configuration

Configuring statistics in ABCD means : precisely identifying the variables and combining them into tables.

4.1.1. Identifying the variables

From either the 'Update database definitions' menu (last but one option) or the last option in the Statistics menu itself, the following screen is shown :


Statistics configuration - List of variables: marc

help edit help file Script: config_vars.php

Variable		Extraction format (PFT)
Leader	LC Classification	v50^a
Leader	Date of publication	if val(v260^c)<2000 then '1900-2000' else F(val(v260^c),1,0) fi

Add

Left-most is the list of fields available for the given database (in this case 'MARC'), followed by the name of the selected field. The interesting part is the third column in the middle : the extraction format. As with the same concept of 'extraction format' for FST's, the administrator can suffice here to simply refer to a field value, like in the first example : v50^a, or process the values with a real PFT. In the second example the years of publication (v260^c) are regrouped in either one category '1900-2000' for all values of which the numerical equivalent (val(v260^c) is less than 2000 (<2000), or the actual publication year if more recent than 1999, by 'F'ormatting this value with zero decimals [F(val(v260^c),1,0)]. Another example - which could actually be applied to the same 'date' field - is to convert the month as a numerical value to the month's name; an example of such PFT is given in the section discussing FST formats.

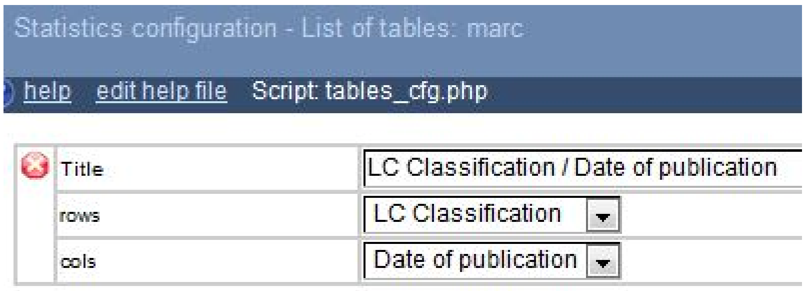
With the 'Add' link more such variables can be identified for this database. In principle all fields which can be more or less 'categorized' are usable, but remember an important law in statistics : not only one can lie with them easily but most of all : it should make sense ! In the example one can see the meaning of checking over time (date of publication) how the subjects of your catalog have evolved (LC Classification), but e.g. cross-tabling a title-field with publication year would not only be impossible (as titles can hardly be categorized) but also not make sense.

If the given field is indexed by a prefix, the prefix can be supplied in the last column, and the row as such can be removed if wanted. The question mark will invoke the FST to check which prefixed have been used in your database.

Any variables 'saved' (with the SAVE button) will subsequently appear in the list of 'existing variables' for later use, i.e. when creating tables with them.

4.1.2. Creating the tables

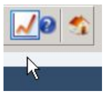
Once variables exist, tables using them either in single or in two-dimensional ('cross-tables') modus can be simply defined and saved, to make them appear in the list of 'existing tables' when running the statistics. The 'title' of a table is the name of it as given by yourself to appear in the list of tables. In the list of available column fields, there is an empty line which can be selected to keep it empty and create a one-dimensional table.


Statistics configuration - List of tables: marc

Title	LC Classification / Date of publication
rows	LC Classification
cols	Date of publication

4.2. Running statistics

The Statistics module can be invoked from either the main Administration menu or from the cataloging toolbar's option :



The main Statistics screen offers three functions to actually run them :

1. Use an existing table
2. Create a new table
3. Generate output

   a. Use an existing table

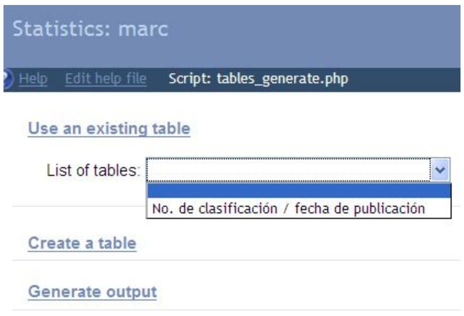
      Existing tables will be listed in the options list by their row/column criteria. In the ABCD demo version one table has been pre-defined : a 'classification code by publication date' table.

      Note

      The list of available tables is taken from a text-file 'tabs.cfg' in the /bases/{dbname]/def/[language]/ folder. This file contains, for each pre-defined table, three values separated by the '|' character :

      - the name as displayed in the menu of tables (which for clarity could be a mentioning of both criteria)
      - the field for the rows
      - the field for the columns

        e.g. Classification number / Date of publication|Classification LC|Publication Date



After selection of the table one simply has to continue by using the 'Generate output' option, see below.

b. Create a new table

As explained above, a table has to be defined by identifying which values (as contained in an ISIS database field) will be used in the horizontal (rows) direction and which ones in the vertical (columns) direction. ABCD will display a list of available criteria (or fields) to select from for both rows and columns. If only one dimension (row) is given, the table will not be a 'cross-tabling' construct but simply analyze frequencies over the variable given for that table.

Note

The list with available fields or criteria is taken from the text-file 'stat.cfg' in the folder /bases/[dbname]/def/[language, which contains on each line one criterion, specified as a field name and a PFT to exactly define how the values should be extracted from the field. The MARC database demo example is :

```
Classification LC|v50^a

Date of publication|if val(v260^c)<2000 then '1900-2000' else
F(val(v260^c),1,0) fi
```

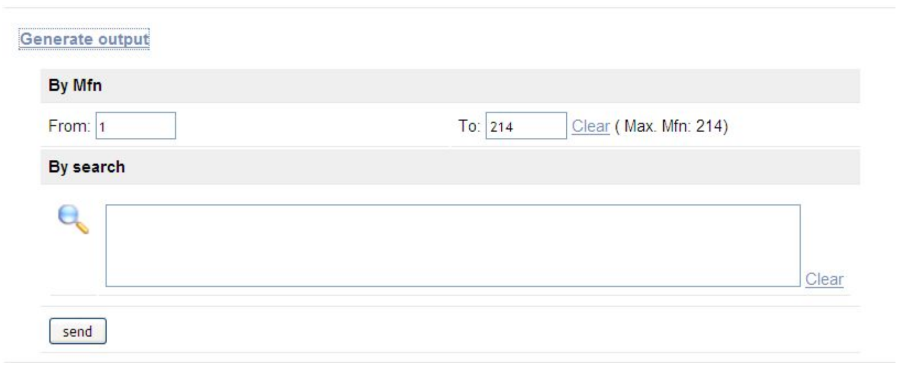
in which the second line illustrates how it can be more than just a simple field value statement by using conditions etc.

Once both the 'Rows' and 'Column' criteria have been defined, the table can be used to generate output as explained below.

c. Generate output

Generation of output in ABCD involves 2 stages : creating the table with the values and - if so desired - creating graphical output charts or exporting the table to other external document formats (e.g. worksheet for spreadsheet software which offers mostly more advanced/detailed graphical output possibilities).

Before creating the output one has to define the range of MFN's (ISIS records) to be used or a 'query' statement to apply the statistics only on a certain sub-set of the database. In the 'search' (or query) box one can use the ISIS Query Language with any accepted statement.


Generate output

By Mfn

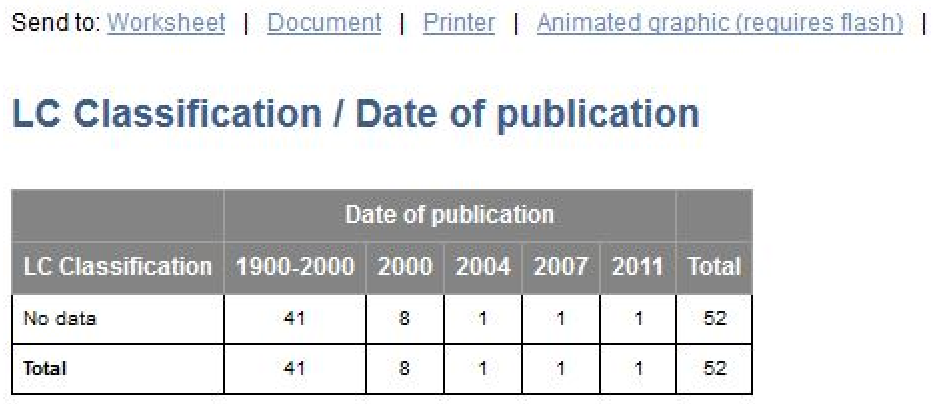
From: 1 To: 214 Clear (Max. Mfn: 214)

By search

Clear

send

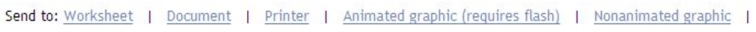
In the first stage it is a matter to actually create the table with all the values in the cells combining both row- and column criteria (e.g. the documents published at a certain publication date belonging to a given LC classification code). Be careful to not calculate tables with individual extended range values (e.g. all years or all LC-codes), but rather use the Formatting Language to combine values into classes, as these make probably much more sense in your statistical report. The example above gives some hints on how to obtain such classes by using a condition on the value of a date, so the user can derive more classes from this basic example.


Send to: Worksheet | Document | Printer | Animated graphic (requires flash) |

LC Classification / Date of publication

	Date of publication					
LC Classification	1900-2000	2000	2004	2007	2011	Total
No data	41	8	1	1	1	52
Total	41	8	1	1	1	52

The output options given by ABCD are the following :

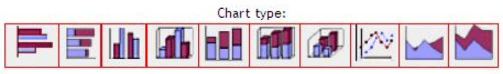

Send to: Worksheet | Document | Printer | Animated graphic (requires flash) | Nonanimated graphic |

- Output to a worksheet will transfer the table to a Spreadsheet for further processing
- Output to a document will transfer the table to a Word Processor document for further processing - this could be practical to include results into your annual report document etc..
- Output to printer allows direct printing of the table using the printer(s) available to your Operating System
- There are 2 graphic outputs provided by ABCD itself :

  i. Animated graphic : this is a fancy way of displaying the graphs, mostly suited for attention-tracking presentations (but requiring Flash software to be installed on the computer)

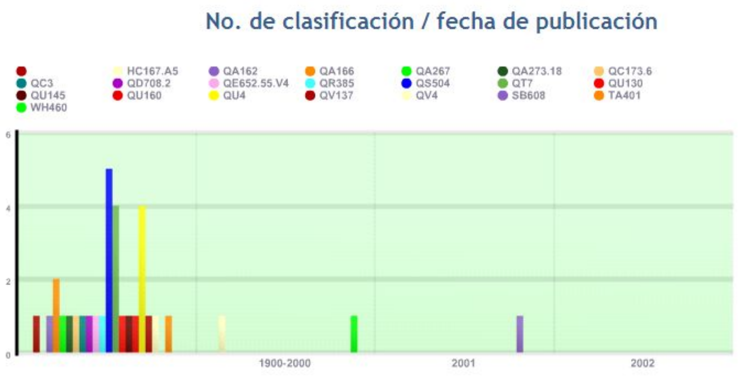
  ii. Non-animated graphic : displaying the graphs but without the fancy animation.

  In both cases several typical graphical output styles (horizontal or vertical bars, lines, three-dimensional bars etc...) are available and should be selected.



If you are not acquainted with such styles, why not just try them and decide for yourself which one presents the message of the statistics in the most clear way ? Remember that conveying a message is the crucial thing here with statistics, not impressing an audience or reader with a jungle of too-much data - as is often the case with statistical tools. Often the simpler the more convincing !

An example output of the graphical output looks like this :



5. Central module : acquisitions management

This module deals with the administration of newly acquired objects and a pre-cataloging function. The pre-cataloged objects can, once acquired, be stored as objects for the Loans module. This is what ABCD 'integration' in between (sub-)modules is about.

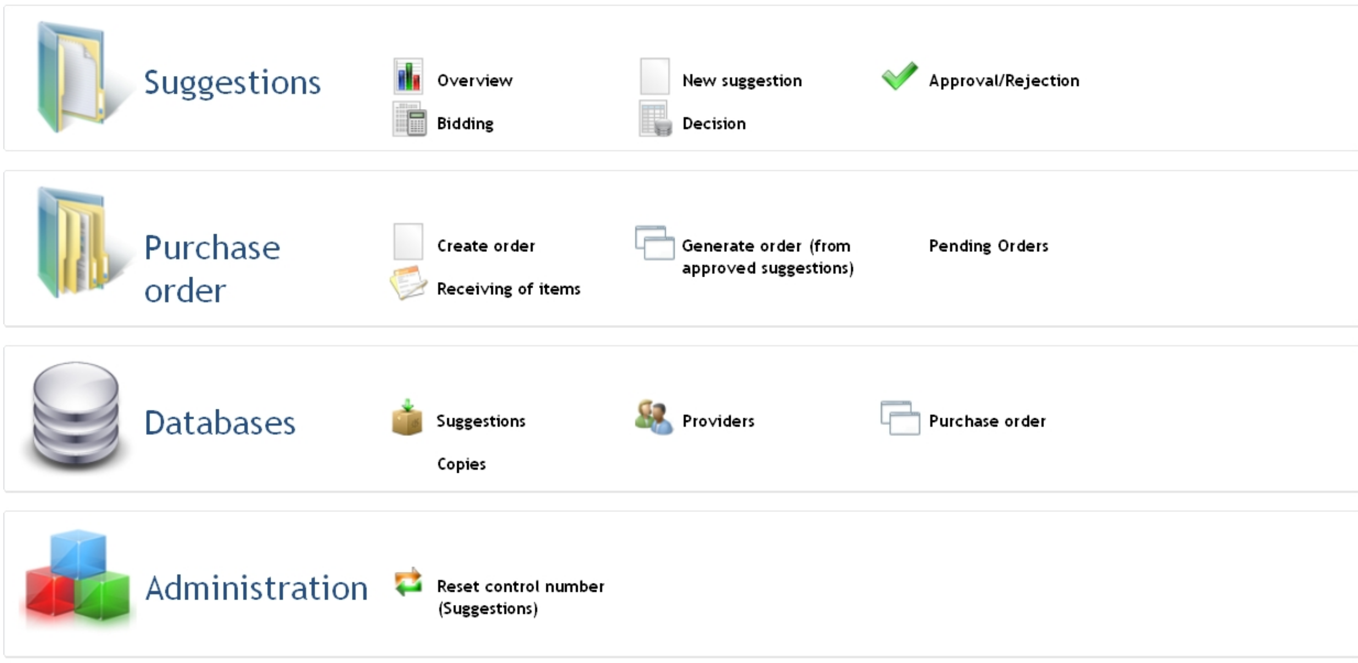
The acquisition module has the following main logical functions :

1. Suggestions : the starting process of ordering/obtaining objects

2. Purchase orders : the actual procurement process of objects

3. Databases : management of the 4 acquisitions-related databases (suggestions, providers, orders and copies)

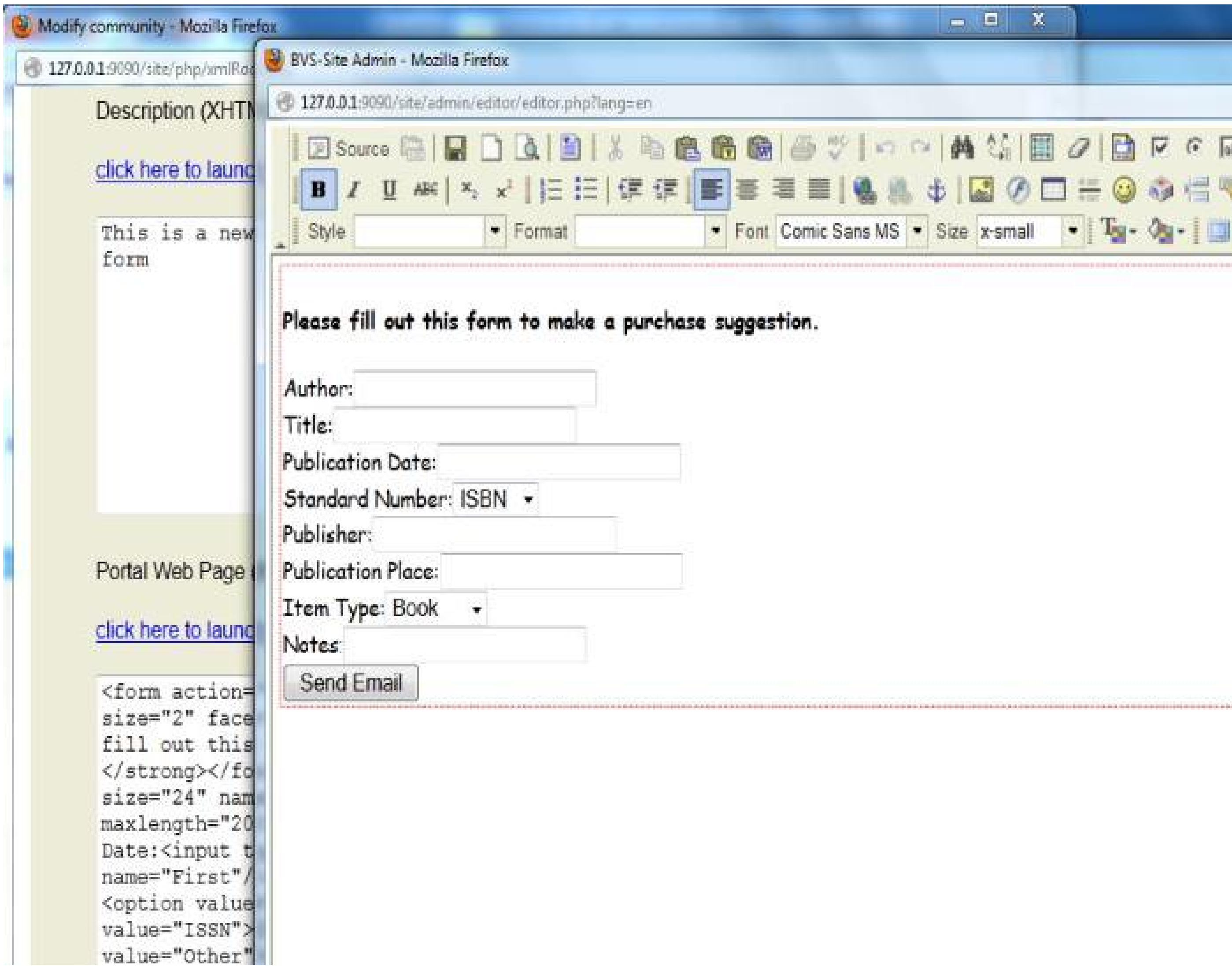
4. Administration : configuration, statistics and reports, weeding (discarding objects).



Let's discuss each of these in more detail.

5.1. Suggestions

The logical 'workflow' of acquisitions starts with suggestions (by library users, colleagues...) to purchase a book, followed logically by a purchase decision (approval or rejection), a bidding process and a decision on where to buy the book. Where the suggestions come from, at this time, has not yet been included into ABCD, but it is conceivable that a form to submit suggestions exists either on a webpage (e.g. the ABCD Site) or an e-mail can be sent to the librarian, either to be converted automatically into a suggestion record or to be manually edited into such a record by a librarian. The illustration here shows the Site-Admin interface where such advanced HTML contents can be created as part of the ABCD Site.


Modify community - Mozilla Firefox
127.0.0.1:9090/site/php/xmlRo
Description (XHTM
click here to launc
Portal Web Page
click here to launc
BVS-Site Admin - Mozilla Firefox
127.0.0.1:9090/site/admin/editor/editor.php?lang=en
Source
Style
Format
Font
Size

Font: Comic Sans MS
Size: x-small

```
This is a new
form
```

```
<form action=
size="2" face
fill out this
</strong></fo
size="24" nam
maxlength="20
Date:<input t
name="First"/
<option value
value="ISSN">
value="Other"
```

Please fill out this form to make a purchase suggestion.

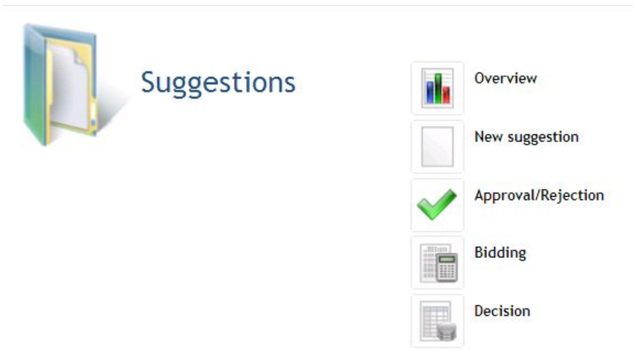
Author:	
Title:	
Publication Date:	
Standard Number:	ISBN
Publisher:	
Publication Place:	
Item Type:	Book
Notes:	

Send Email

The 'submit button' of this HTML code, labeled 'Send Email' here, triggers the action 'mailto:' with pre-filled e-mail address for the library's acquisition administration, and results in a structured e-mail in a text-format ('tagged text') which can be processed either manually or even converted into an ISIS-record, even by ABCD techniques.

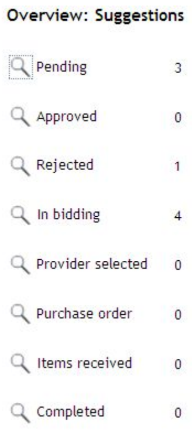
A record on suggested books, whether or not actually purchased, needs to be kept for future use (e.g. when the same book is suggested again).

This 'logical' flow is more or less reflected in the ABCD-acquisitions module.



5.1.1. Overview

Here the librarian can consult the status of the actual acquisition system's activities, with an option to get a listing (popping up in a separate window) of the activity.



Overview: Suggestions

Pending	3
Approved	0
Rejected	1
In bidding	4
Provider selected	0
Purchase order	0
Items received	0
Completed	0

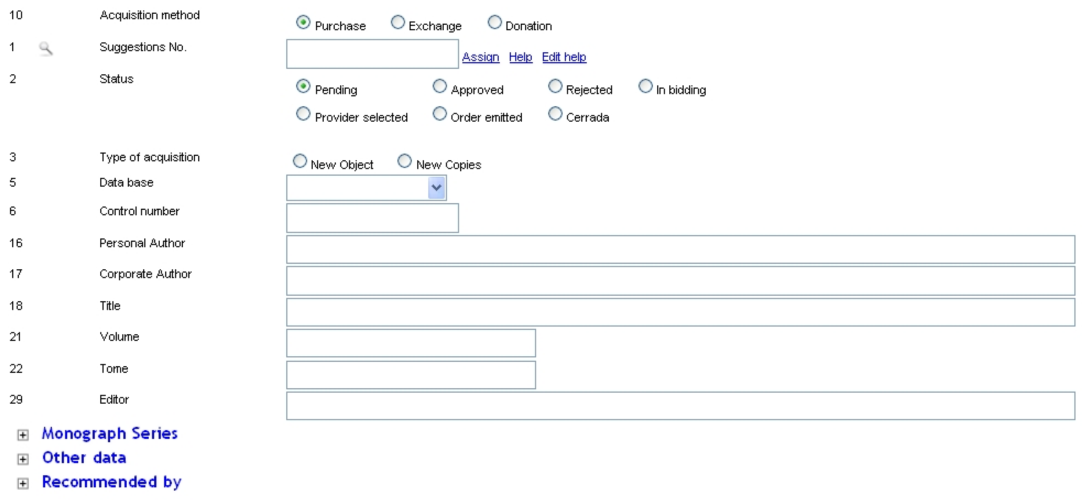
No changes or editing functions are provided here, so this is only for consultation of the acquisitions system.

5.1.2. New suggestions

Here a worksheet is provided for the acquisitions manager to actually enter a new suggestion for acquiring an object, with some bibliographical fields but also the status and a 'recommended by' field. Under here is shown part of the data-entry worksheet.

Note

For the 'date of suggestion' field a calendar pop-in function is used.

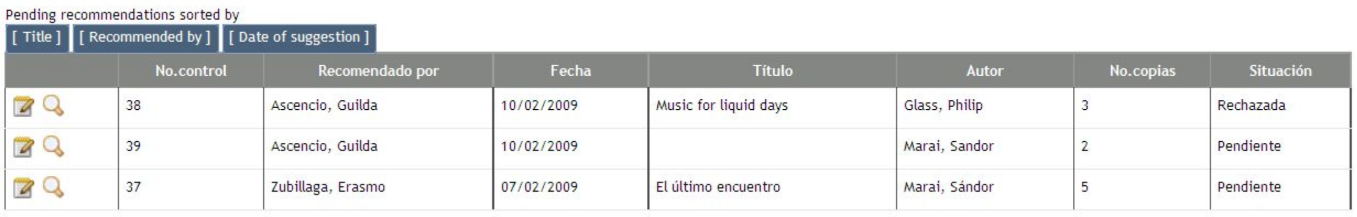


10	Acquisition method	[✓ Purchase] Exchange Donation
1	Suggestions No.	Assign Help Edit help
2	Status	[✓ Pending] Approved Rejected In bidding Provider selected Order emitted Cerrada
3	Type of acquisition	New Object New Copies
5	Data base	
6	Control number	
16	Personal Author	
17	Corporate Author	
18	Title	
21	Volume	
22	Tome	
29	Editor	

Monograph Series
Other data
Recommended by

5.1.3. Approval/Rejection

ABCD will give a listing of the requested activities with their status, sortable by either Title, Recommended by or Date of recommendation.

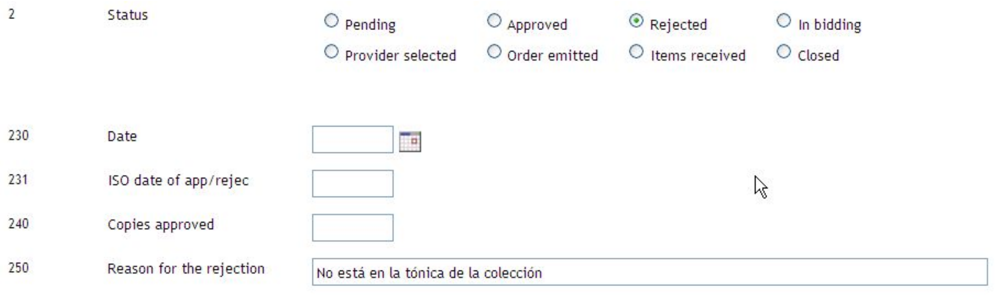


Pending recommendations sorted by

[Title] [Recommended by] [Date of suggestion]

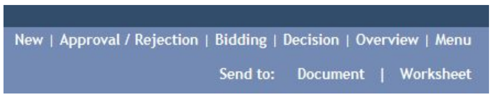
	No.control	Recomendado por	Fecha	Título	Autor	No.copias	Situación
	38	Ascencio, Guilda	10/02/2009	Music for liquid days	Glass, Philip	3	Rechazada
	39	Ascencio, Guilda	10/02/2009		Marai, Sandor	2	Pendiente
	37	Zubillaga, Erasmo	07/02/2009	El último encuentro	Marai, Sándor	5	Pendiente

Each of the items can be opened for editing, where the actual approval or rejection can be given. Under is an example of a rejected item :



2	Status	Pending / Approved / [✓ Rejected] / In bidding / Provider selected / Order emitted / Items received / Closed
230	Date	
231	ISO date of app/rejec	
240	Copies approved	
250	Reason for the rejection	No está en la tónica de la colección

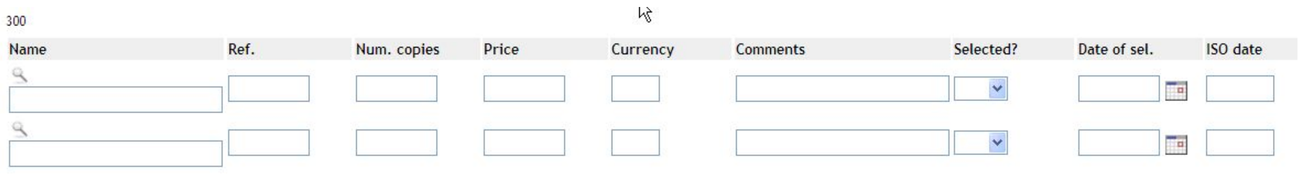
At this point, after having either cancelled or saved (updated) the transaction, direct access to the main Acquisition menu's options is also offered :


New | Approval / Rejection | Bidding | Decision | Overview | Menu
Send to: Document | Worksheet

Also options are provided to send the listing to either a document (.DOC format) or a Spreadsheet (.XLS).

5.1.4. Bidding

At this stage ABCD offers, starting from a sortable listing of the actual transactions, a worksheet where the bidding process can be kept track off : for each participant in the bidding some essential data (name of bidder, price offered etc.) can be stored with a box to mark acceptation or not :



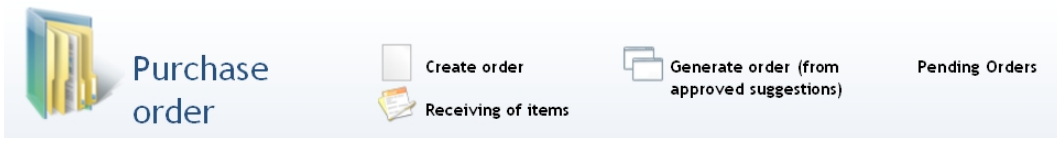
300

Name	Ref.	Num. copies	Price	Currency	Comments	Selected?	Date of sel.	ISO date

5.1.5. Decisions

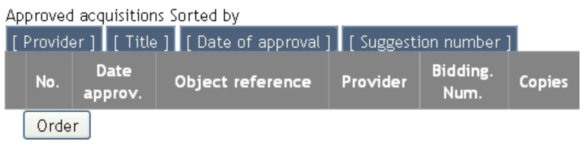
Finally the recommendations administration is completed by a listing of the approved offers in a format (worksheet) similar to the above. Again this listing can be exported to either a document or a spreadsheet software.

5.2. Purchase orders



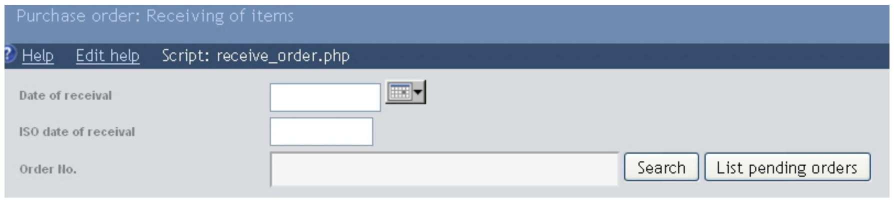
[!!] The new acquisitions first need to be categorized as either purchase, donation or exchange. According to the selected option the subsequent editing form will contain slightly different elements, but these don't need a lot of explanation here.

ABCD facilitates the ordering of objects by providing a form to create a new order, by listing the pending orders with editing possibility and by providing a listing or search possibility in the list. The received items, with their price and number of items acquired, can be filled in. Whenever a selection has to be made from a list of candidates, these can be sorted by different criteria which are listed above the list itself.


Approved acquisitions Sorted by
[Provider] [Title] [Date of approval] [Suggestion number]

No.	Date approv.	Object reference	Provider	Bidding. Num.	Copies

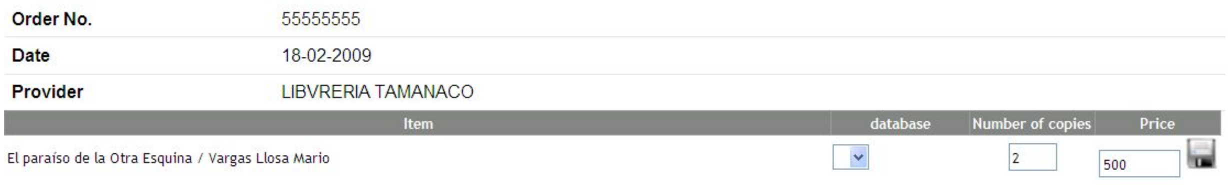
Acquisitions which have been received can be entered into the inventory with their date and order-number in a small form :


Purchase order: Receiving of items

Script: receive_order.php

Date of receival	
ISO date of receival	
Order No.	

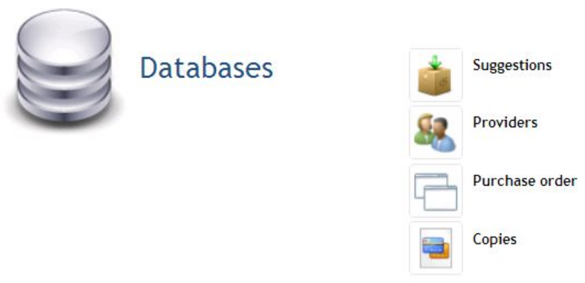
Listing the received orders is the last option in here :



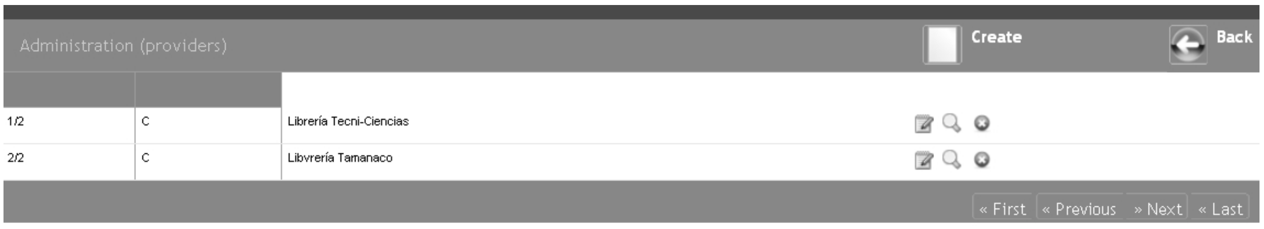
Order No.	55555555
Date	18-02-2009
Provider	LIBVRERIA TAMANACO

Item	database	Number of copies	Price
El paraíso de la Otra Esquina / Vargas Llosa Mario		2	500

5.3. Databases



The ABCD Acquisitions module maintains 4 databases, which can be accessed directly here, each time with a search function for fast retrieval of a specific record. Each of the retrieved records can then be edited with its dedicated worksheet and saved if indeed changes have been made. As an example we show under here the providers database.

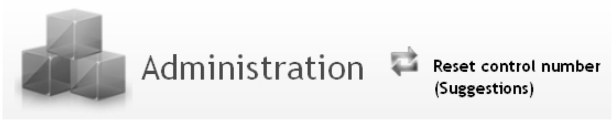

Administration (providers)

1/2	C	Librería Tecni-Ciencias
2/2	C	Libvrería Tamanaco

The 'Create' icon allows to create a new provider record.

5.4. Administration of acquisitions module

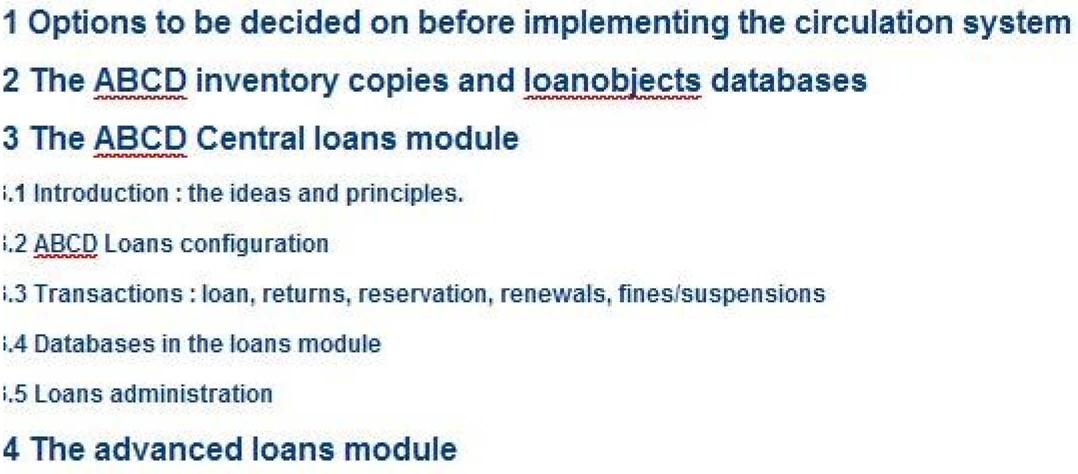
In this last section of the Acquisitions module some functions are offered to manage the acquisitions system with reports, statistics, configuration of some parameters and also weeding the acquisitions system by deleting unwanted items. In the initial version only one function is provided : reset the control number (which is the number of the last entered item or copy control number, to which 1 will be added when creating a new one ('auto-increment' field). Actually this number is kept in a file 'control_number.cn' in the folder \ABCD\www\bases\copies\data\, where simply the last assigned number is stored. Whenever copies are created, this file is write-protected to avoid duplication of the same number by concurrent users - so other users have to wait until this file is given free for writing.



6. Central module : loans/circulation

This section is about the ABCD basic loans module as it comes pre-configured with the system. Still, always following the ABCD philosophy of flexibility, the loans system can work with any bibliographic and objects database.

The overall structure of this chapter is shown in the screenshot here :



We start with a brief discussion of the decisions to be taken before using the ABCD Circulation module(s), then discuss two special databases which are used next to the catalog and users databases, and then deal with the details of the implementation and the use of the daily transactions themselves.

In addition, ABCD offers an 'advanced loans' module (EmpWeb) which can cope with much more complicated situations in e.g. multi-branch organisations with different loans policies etc. This separate module is only very basically discussed in the last part of this chapter, while we need to refer to different documents dealing with it in more detail.

The transactions in this Advanced Loans module will be stored in SQL-tables, as are the users-data, either locally on your own server or on another SQL-server, but it can also access user-data (through WebServices) stored in ISIS-databases proper to your own ABCD.. This allows more flexible high-performance applications of the loans module; e.g. one can use a users-database produced/maintained elsewhere in your institution (e.g. registrar's office) while still having the possibility to quickly create users on site in your own ABCD-users database. In addition EmpWeb or ABCD Advanced Loans offers a reservation function and a function 'MySite' where end-users can check and keep track of availability of loan-objects in their personal (password-protected) account of ABCD-Loans. The standard circulation module of ABCD Central now (v1.2t) also offers reservations but will offer 'MySite' only from v2.0.

This advanced module however requires installation of additional software (Java/Jetty and an SQL database) and is therefore offered as an optional extra module : without it all other modules will still work fully.

We suggest to check the following criteria in order to decide on whether you will need the advanced module or not :

- multiple servers in your system ?

- multiple users/copies databases in your system ?
- high volume of transactions ?
- any data source needs ODBC drivers ? (ODBC drivers are software to couple mostly relational databases or SQL-tables to software - since EmpWeb uses Java we are actually talking about JDBC drivers here, but these are available for almost any SQL-database).

If you have a 'yes' with serious importance for your system, you will be better off with the advanced loans module. Most smaller libraries however, certainly where there is a lack of expertise on using Java and SQL-databases, will be better of with this ABCD Central Loans module as it is fully based on the ISIS-database technology and doesn't need any additional software to be installed. Since now also reservations and soon MyLibrary are also implemented in the Central Loans module, little has to be sacrificed.

6.1. Options to be decided on before implementing the circulation system

In view of the flexibility of ABCD system managers have to take some decisions before the implementation of the circulation system can actually be performed :

1. Central Loans or Advanced Loans (EmpWeb) : this is probably the most important but at the same time tricky decision to be made, since everything else following will be quite different according to the choice made here. The two systems are now functionally more or less equivalent (Central Loans having added reservations in v1.2t and OPAC/MyLibrary access from v2.0 on) but only share the loanobjects database and - as an option - the users database. EmpWeb uses Java with an SQL database and web-services to link to ISIS or possibly other sources of information. So the criteria to be used for this decision are :

   - is there expected a high volume of daily transactions on several concurrently running terminals ?
   - is there a potential need for use of several different policies and the possibility to add new rules to the loans system ?
   - do you need to spread the system over different servers ?
   - are users data available form other sources, most probably SQL-based ?
   - is there sufficient knowledge available on using an SQL-based database, e.g. MySQL
   - does the server run Java and either Tomcat or Jetty ? (note : this is a lot less risky than running local Java applets in your browser, where Java had some security problems of late)

     If most of the answers tend to be positive, you might need to consider the use of EmpWeb. We think this is typically the case in larger libraries, e.g. university libraries.

     Looking at this decision from the other side, there are some reasons why you might need to use the Central Loans instead :

   - Do you need to keep copies-information in the catalog-database ?
   - Is it rather impossible to put Java/Tomcat/Jetty and MySQL on the server ?
   - Is only knowledge on ISIS and ABCD available in the house ?
   - Do you need to combine several catalog databases into your circulation system ? If yes : EmpWeb can also do that but does not proved an easy implementation interface for this...
   - Are expected transaction numbers low, i.e. not hundreds per hour ?

     Differences which existed before but no longer now from v1.2t on are :

- processing more than one item at a time (now Central Loans also allows multiple item selection for issuing, returning..)
- reservations were not but now are included
- soon (as from v2.0) also reservations and renewals can be accessed from the OPAC by end-users.

We expect that most smaller libraries (like school-libraries, small local public libraries) will opt for the Central Loans, esp. if no experts are available - as is often the case - to manage MySQL, Java, another server (Jetty) on top of Apache, and only some ISIS-knowledge and ABCD-experience are available.

If you decided for EmpWeb, check the dedicated manuals and documents (they exist but are not yet very well co-ordinated and elaborated). If you decided for Central Loans, go on reading here.

2. Dedicated loanobjects database or taking copy information from catalog

ABCD normally uses two dedicated extra (to the catalog) databases to manage objects as opposed to 'titles' : COPIES for management of the stock and inventory, and LOANOBJECTS to manage the circulation system. Strictly speaking we can leave out of the equation the COPIES database here as for the circulation system it is only an intermediate step to create loanobjects, but in reality the circulation system only needs some basic information to run :

- the unique identifier of each object, e.g. barcodes, where several copies of the same title have DIFFERENT identifiers.
- the name of the catalog database from where title etc. can be 'borrowed' from
- the control_number of the title of the object into the catalog
- location (library, position within a library like section, sub-library, shelf, room...) of the object
- (optionally : volume and part of multi-part objects)

As can be concluded from this limited list of loans-parameters, one could wonder why create a dedicated database for this ? ABCD thinks that there are several good reasons to keep such data in a dedicated database LOANOBJECTS :

- the nature of these data are a bit more 'dynamic' whereas the catalog is meant to be a rather stable database which represents a huge man/hour effort and should be maintained carefully as a real 'treasure'
- when there would be regularly many copies (e.g. more than 5) for one same title, the copy-related field(s) would start slowing down the catalog database (which increases unnecessarily in size, logically deleted records etc. : ballast !

But adding a LOANOBJECTS database introduces additional complexity, so when your library is rather small, does not have frequent new acquisitions of existing titles or movements accross the institute etc., one could prefer to keep the copy-related information inside the catalog.

Note

Even if we have seen library systems where not only the copy information but even the loans-transactions are kept with the title-records in the catalog, for the same 'simplicity' reasons, we think that this is out of the question as dynamics and nature of the data are totally different from a catalog database.

If you decide to keep the copy-information inside the catalog, you can do so with Central Loans (not with EmpWeb) but will have - only at the initial configuration stages - to add some extra information to explain to the system exactly where in your catalog these data can be found : 'extraction formats' which are technically speaking ISIS PFT's.

If you don't see a reason to do so : welcome to the default ABCD circulation set-up where you can create copies and loanobjects directly from the catalog cards to immediately include them into your circulation system.

In reality your system will end up being of one of the following three types in this respect :

- you put all copy information into the LOANOBJECTS database (no loans.dat file will be created and used)
- you don't use the LOANOBJECTS database at all, just one (or more) catalog(s) with the copy info somewhere into the bibliographic records
- you have a mixed system : you use the LOANOBJECTS database (e.g. for the main books catalog) but also some 'special' databases having the copy info inside (e.g. for AVM materials)

In practice this means that whenever you deal with a loans-transaction, you have to select the 'source' database, i.e. the database from which the copy info is to be taken, and when this is not the LOANOBJECTS database, you will have needed to add some extra configuration data to explain to the system from exactly where the copy info can be taken.

3. Which catalog(s) to use in the circulation system ?

   ABCD allows to use more than one catalog in the loans system (both in Central Loans and EmpWeb, in this last case however configuration has to be added manually inside XML-parameter files). In most cases however the system will run with only one single catalog. There are however no other consequences of using multiple catalogs than simply making sure all catalog databases have got their circulation configuration (see below) and their name added to each loan-object record.

4. Which users categories to use ?

   ABCD allows local creation of ANY list of user-categories, as much detailed (or not...) as you wish. Whereas it is possible to change this list later on, remember that all existing users will need to be re-categorized in case of a change, so better to think carefully at the start on this issue. Our experience is that many librarians initially prefer to start with very detailed lists of user-categories (e.g. 1st year student, 2nd year student etc...) but later on regret as the policy-matrix becomes too complicated and they return to lesser categories.

5. Which objects categories to use ?

   Exactly the same reasoning as in the previous point 4 apply : maybe less is more ! As a rule of thumb one could say : only introduce a new category if there will be real consequences for the loans-policy, e.g. longer/shorter loan periods, different fines etc.

6. The main issue : which policy or policies to use ?

   After all previous decisions have been made, ABCD - as any other library system - will allow to define rules for the circulation system : how many books each category of users can take on loan for how long, with which penalties in case of late return or non-pickup of a reservation etc.. A set of such rules is a policy, so in principle one needs to create one policy per combination of users/object categories. This is why these categories best are kept to a minimum : they multiply !

   A possibly important decision here is also whether there will be loans by hour in addition to the normal loans-per-day : e.g. intra-library consultation of precious materials could be put on loan for just 2 hours etc. with of course quite different parameters applying. Both Central Loans and EmpWeb allow this loan-by-hour option.

   Policies are intrinsically non-technical but purely managerial decisions, reflecting the 'style' (e.g. strict/sober/efficient vs. flexible/human/subjective or anything in between in fact).

   ABCD does not imply or promote any style or policy, only provides quite a lot of parameters with which the policy can be built.

7. Calendars, fines and currency

Finally you will have to decide on which days count for your circulation system when calculating duration of a loan or overdue return, the unit per such day to be applied in the fine or suspension measure and the monetary unit for these to be used. As for the calendar issue : you will simply have to identify into calendars per month (Central Loans) or per year (EmpWeb) the non-operational days, e.g. weekends, holidays. Fine calculation or calculation of days of suspension need to be rectilinear to the number of time-units : it is e.g. not possible - unless extra coding is done, in EmpWeb possible using the Groovy language, in Central it has to be put as JavaScript or PHP inside the scripts - to calculate fines in an exponential way (two days delay means 4 units instead of 2) or other mathematical formulas, no matter how sophisticated they can be from a managerial point of view.

After you have prepared your circulation system's philosophy based on the above discussed decisions to make, we are almost ready to start actually implementing the system, but first let's look into some more details at some useful points regarding the circulation system setup.

6.2. The ABCD inventory copies and loanobjects databases

ABCD uses two different databases to deal with information based on the physical objects in the library : the COPIES and LOANOBJECTS. Both databases have different purposes and scopes and therefore the data are kept separate. A good understanding of their different purposes and structures will help in understanding ABCD. Let's discuss each of them here.

1. The ABCD COPIES database.

   This database has the function of keeping track of all administrative data on objects in the library collections. These data in principle need to be kept for inventory and book-keeping reasons, whereas data related to loan-objects can be deleted when the loan-object is removed from the collection. Actually the record created at the end of an Acquisitions procedure (from suggestion to received item or anywhere in between) will be stored in this copies database. The copies field structure is as follows (field tags are followed by the field name) :

1|Control number	68|Acquisition type
10|Database	70|Provider/Donor/Institution
20|Call number	80|Date of arrival
30|Inventory number	85|ISO Date of arrival
200|Status	90|Price
35|Main Library	100|Purchase order
40|Branch library	110|Suggestion number
60|Volume/Part	300|Conditions
63|Copy Number	400|In exchange of

   The field 30 (Inventory Number) is created by the auto-increment mechanism of ABCD : the next number to be assigned is stored in a small file 'control_number.cn' in the data-folder of the database (in this case COPIES), as each record has to be identified by a unique value which can be used for linking from the other databases (as 'primary key').

   Essentially only the first 7 fields in the COPIES worksheet have to be filled in, the remaining fields could be left empty if you are not planning to use this inventory database fully, but only for the purpose of creating loan-objects from it for your circulation system.

   Differently from the LOANOBJECTS-database the COPIES-database has one record per object : each 'bar-code' (or inventory number) has its own record in the database, whereas in LOANOBJECTS each title has one record with all existing copies of that title are put into one repeatable field. These records are normally created by the 'Add to copies' function of the cataloging module.or the Acquisitions module.

This database has to be indexed with 2 mandatory keys :

```
1 0 "CN_"V10,"_"V1,

200 0 "STATUS_"V200^a
```

The first key here assures that the identifier of each copy can be quickly referred to by the prefix 'CN_' followed by the bibliographic database name V10 (because several catalogs can be combined into one copies-database) and the identifier in that catalog itself (V1). For an entry in the copies-database to be allowed to enter into the loanobjects database it needs such an entry in the Inverted File (and assumed the status-field is at least at level 2, which means 'verified and stamped', that is why the second FST-entry indexes on the status of each copy).

In the cataloging module copy records will be shown in a separate smaller window when clicking on the 'Add copy' icon of the record-toolbar and then on 'show copies'.

2. The ABCD LOANOBJECTS database

This database has different goals from the copies-database : it's purpose is only to provide the necessary data about objects which can be used in the loans-system. Its contents are quite limited :

FIELD	SUBFIELDS
1| Control Number	-
10 Catalog name	-
959 Object info	^i Identifier (barcode etc.)
	^l library
	^b location (shelf, special collection etc.)
	^o object type (book, AVM...)
	^v and ^t volume and part info

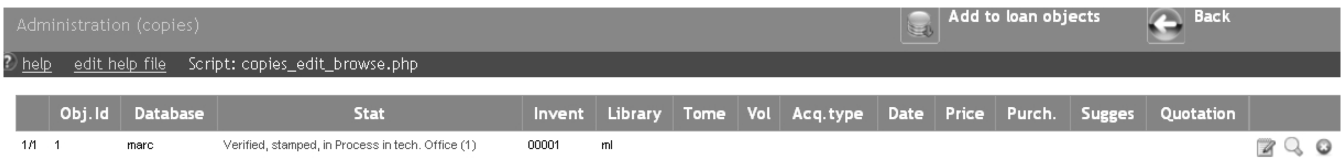
As can be seen from this very short list of fields (in fact only 3), each loan-object is identified by its control_number V1 (automatically filled if created with the 'add to loanobjects' option of the cataloging module), the bibliographic database to be linked to (V10) and the actual details on each copy as a repeated subfielded field V959 (this gives, in ISIS-technology, the fastest performance in case of high number of copies - we have tested with over 350 copies). The subfields allow to specify the unique identifier (e.g. barcode) of each copy, the main library and location within that library (e.g. shelving information), the loan object type (books e.g. will have different loan-parameters from video's) and in case of a multi-volume work, therefore optionally, the volume and part - identifiers.

The loanobjects database has to be indexed with at least the following instruction :

```
1 0 "CN_"v10,"_"V1
```

because (as in the copies-database) the string 'CN_' followed by the catalog-name and catalog-identifier allows each copy to be identified. If this entry cannot be found in the loan-objects database for a given book, it will not be possible to use the object in the loans system and an error will be issued on the screen.

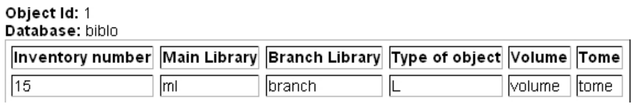
Loanobjects normally are created from the cataloging module after having created the according inventory copy with the icons in the record-toolbar of the cataloging module :



Script: copies_edit_browse.php

	Obj.Id	Database	Stat	Invent	Library	Tome	Vol	Acq.type	Date	Price	Purch.	Sugges	Quotation
1/1	1	marc	Verified, stamped, in Process in tech. Office (1)	00001	ml								

After having clicked on 'Add to loan objects' the record will have been created in the loanobjects database. If this database is to be edited directly as an ABCD-database, remember to include it in the list of databases (bases.dat, which can be accessed directly from the Operating System but also through the corresponding 'Update database definitions' menu in ABCD Central) and to the profile of users which need access to this database directly. There the loanobject records will be presented for browsing and editing in a table format (as explained in the section on the FDT-definition for 'group'-fields).


Object Id: 1
Database: biblo

Inventory number	Main Library	Branch Library	Type of object	Volume	Tome
15	ml	branch	L	volume	tome

6.3. The ABCD Central loans module

6.3.1. Introduction : the ideas and principles.

This loans module is called 'basic' because it is fully integrated with the other ABCD Central modules, using the same underlying technology: ISIS-databases, ISIS Script and PHP. Looking at its functionality however one could hardly call this 'basic' : as from v2.0 all functions, including reservations and 'MyLibrary' (OPAC access) are available and two fundamentally different approaches are offered : with the copy-information still in the catalog (new since v1.2t), often practiced in small libraries for simplicity reasons, or with copy-information in the inventory and LOANOBJECTS dedicated databases. As explained above, one can even mix both systems, with e.g. all books in the LOANOBJECTS and all other objects with copy information into their description records (the 'catalog').

Let's first explain this double approach (which, by the way, is NOT available when using the Advanced Loans Module EmpWeb, which only allows the second approach unless manually changes are applied in the XML-files containing all the configuration details mapping fields and databases). In essence, what one needs in order to be able to circulate a book in addition to having the book cataloged with its title and bibliographical data, is a unique identifier for each physical copy of the title. Titles indeed cannot be put on loan, as they are not physical units but - according to the FRBR principles - intellectual units at a higher level closer to the top-level of 'work'. Today mostly such unique identifiers are applied as 'barcodes' : labels sticked onto the physical units with a number printed on them, this number being also represented with the bar-code system for quick reading by bar-code optical readers - but they still remain essentially 'numbers', which could also be entered by a keyboard e.g. In addition to this essential identifier, most libraries want to attach some more elements of information to each physical copy of a title : the library where it is going to reside physically and the more detailed location inside that library, e.g. collection code, classification category or shelf number, and if present data on the volume and part-numbers of a multi-part title. These data are needed only to facilitate the practical work of processing books in the circulation system.

In ABCD Central we add one more necessary field : the name of the catalog database, simply because in one loans-system several catalogs can be combined, and in order to 'join' bibliographic information - which is not repeated in the COPIES and LOANOBJECTS databases - the system needs to know which catalog database to link to.

Since these necessary identifiers for a physical loans object are rather limited, in smaller libraries they could be simply kept as part of the description in the catalog (see the discussion above in section 1 of this chapter). The ISIS-repeatable field feature makes it even quite handy to do so : combined with the subfields idea one could suffice with only one field in which these data are kept, one occurrence at a time for each physical copy available, in the sequence of subfields. Only when - in reality or potentially - not more than just a few copies (let's say less than 10) would need to be accounted for, this approach is practical : forget about adding 300 copies of the same title into one single catalog record. Even if it would still be possible regarding storage capacity of one record (ISIS can do large records), it would 'spoil' the catalog with data of a different nature, e.g. much more dynamic than the stable bibliographic fields with e.g. copies being added, removed, moved to another location etc... once in a while. This is also the reason why we do not recommend putting, as is also done quite often in smaller systems, the loans-data themselves into the catalog : date of issue, code of the loan-taker, date of return etc. as these are BY DEFINITION very dynamic data and therefore do not belong in a catalog. In ABCD, regardless of the approach chosen, the loans data (the 'transactoins') are always kept in a dedicated database (named 'TRANS'). Only for the few copy-related data for a small number of copies the catalog could be used.

The other model, using a dedicated LOANOBJECTS database, is used - since the early days - by the standard ABCD loans module to allow also applications with many copies in larger libraries. This module takes as its departing point the 'objects' created by the acquisitions module into the separate database 'copies', or created directly from the catalog at the cataloging stage, from which subsequently 'loanobjects' are created into yet another dedicated database, at least for those objects which are meant to become part of the loans system. Not all e.g. precious objects need to go into the loans system and that is exactly the reason why ABCD prefers to keep, in addition to the catalog database, not just one but two extra databases : one for stock-keeping (COPIES) and one for circulation management (LOANOBJECTS).

In reality, as we will see later on, the difference is only relevant when configuring the circulation system : the ISIS 'formats' (PFT's) needed to extract the necessary identifiers (e.g. barcode, object-category, location) will be either applied to the LOANOBJECTS database or to the special field(s) in the catalog database (e.g. MARC). In the case of the LOANOBJECTS database ABCD knows which fields to look for, in the case of a catalog - which can be in fact any - additionally the (sub-)fields to be used have to be identified. That is the main difference, so the further configuration actions are not different, e.g. defining loan policies, users-database fields etc. If at least one catalog database is used in addition to the LOANOBJECTS database, one has to select the 'source' database for each transaction from the databases list defined for this purpose.

Once the objects are correctly identified, the system is ready to apply rules on all kinds of 'transactions' on them : issuing to a user, returning, reservation, loans renewal. Rules for all types of transactions can be defined and will be applied according to the object category in combination with the user category. Categories for objects and users can be defined 'ad libitum' with specified number of objects allowed, hours/days (taking into account a calendar specific for the library), fines and renewal conditions for each combination object/user. These actual loans transactions are all stored, regardless of the chosen approach, into a dedicated database 'transactions', which itself again has a rather simple structure (and therefore SQL-like tables are used for them in EmpWeb) : essentially 3 types of data have to be stored : 1) type/data on the transaction (issuing, returning and renewal) 2) the data about the object involved and 3) the data about the user involved. The only more complicated field here is v100, which has subfields to store bibliographic data taken from the catalog database : author, title etc. as wanted to display the transactions in a more user-friendly way.

Now that the principles and ideas have been exposed, we can turn to the more practical and in-depth discussion of the different aspects of the loans system, following the logics of the main Circulation module home-page.

The main menu of this loans module has three sections :

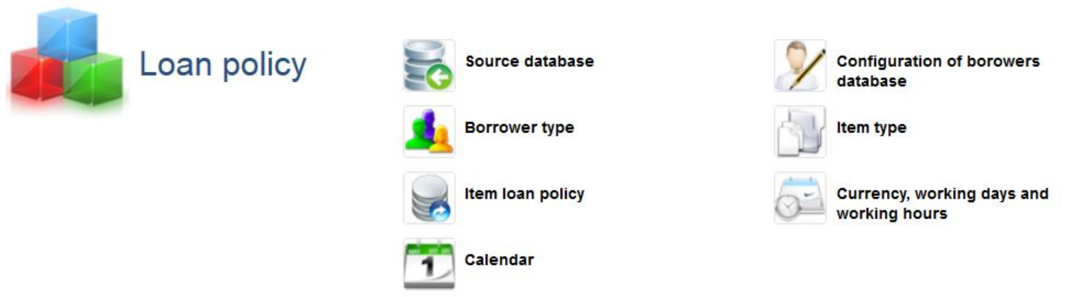
1. Transactions : here the real every-day loans transactions (loaning a book to a user, returning it, reservations etc.) are dealt with.

2. Databases : here the databases on which the loans system is based can be accessed and managed : the borrowers or users, the transactions, the reservations and the fines.

3. Configuration of the loans system : here the 'rules' can be defined for combinations of object types with user categories, and calenders, currency etc. can be defined.

6.3.2. ABCD Loans configuration

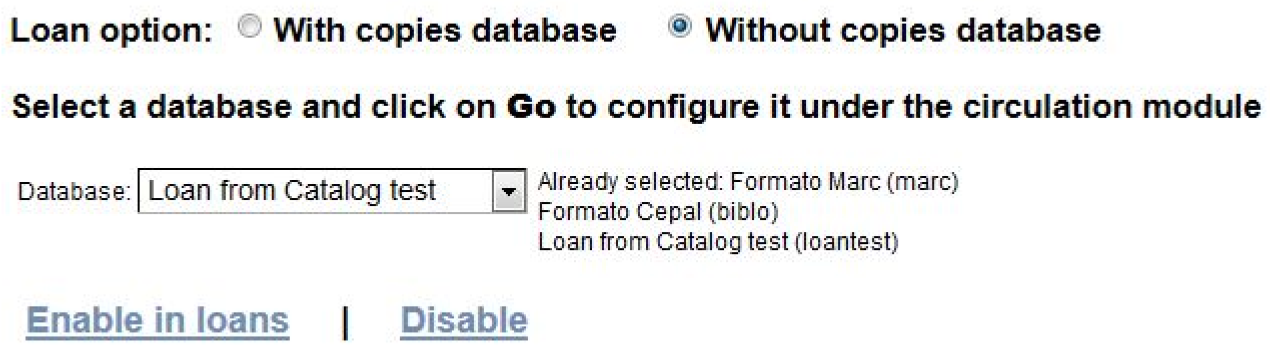
The configuration of the loans is the first thing to do before trying to operate the loans system. Such configuration entails several parts which we discuss subsequently one by one :


Loan policy
Source database
Configuration of borowers database
Borrower type
Item type
Item loan policy
Currency, working days and working hours
Calendar

6.3.2.1. Source database configuration

The loans configuration in ABCD allows to define which source bibliographic databases (catalogs) to link the loans system to - it can be any database indeed ! - and to define the parameters which will constitute the 'policy' on each object/user combination to be applied.

Since any database can be linked into the loans system as 'source', there is a need to explain to the loans system how values from these databases will be used in the loans system. This is the point where the interface and actions to be taken slightly differ according to whether one uses a dedicated loanobjects database or takes these data from the catalog database, so we will discuss both situations here separately. When configuring the system one needs to clearly identify for each 'source database' which option is used, at the top of the dedicated screen following a click on the 'Source database' option of the Loan Policy menu; we first show the example 'Without copies database' (meaning : direct use of the catalog for copies identification) :



Loan option: With copies database [✓ Without copies database]

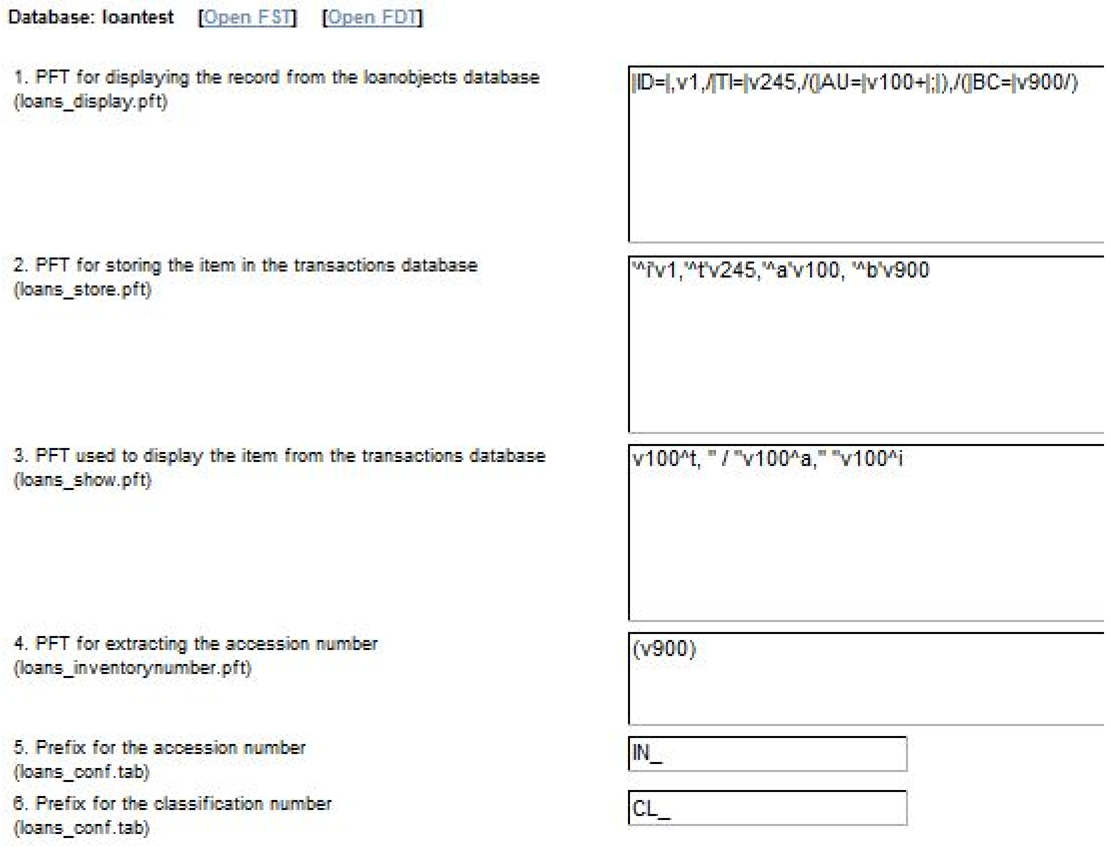
Select a database and click on **Go** to configure it under the circulation module

Database: Loan from Catalog test

Already selected: Formato Marc (marc)
Formato Cepal (biblo)
Loan from Catalog test (loantest)

By clicking on the link 'Enable in loans' the configuration worksheet is opened, which gathers the formats to create the correct identifiers for building the loanobject to go to the transactions database.

First we show the configuration worksheet for a source database in the option 'Without copies database', meaning the database to take loan-objects data from is the catalog itself. This also means that when the dialog mentions 'the loanobjects database' in fact the database is meant which contains the copy information, which can in this case also be the catalog itself.



Database: loantest

1. PFT for displaying the record from the loanobjects database (loans_display.pft)

```
|ID=|,v1,/|TI=|v245,/(|AU=|v100+|;|),/(|BC=|v900/)
```

2. PFT for storing the item in the transactions database (loans_store.pft)

```
'^i'v1,'^t'v245,'^a'v100, '^b'v900
```

3. PFT used to display the item from the transactions database (loans_show.pft)

```
v100^t, " / "v100^a," "v100^i
```

4. PFT for extracting the accession number (loans_inventorynumber.pft)

```
(v900)
```

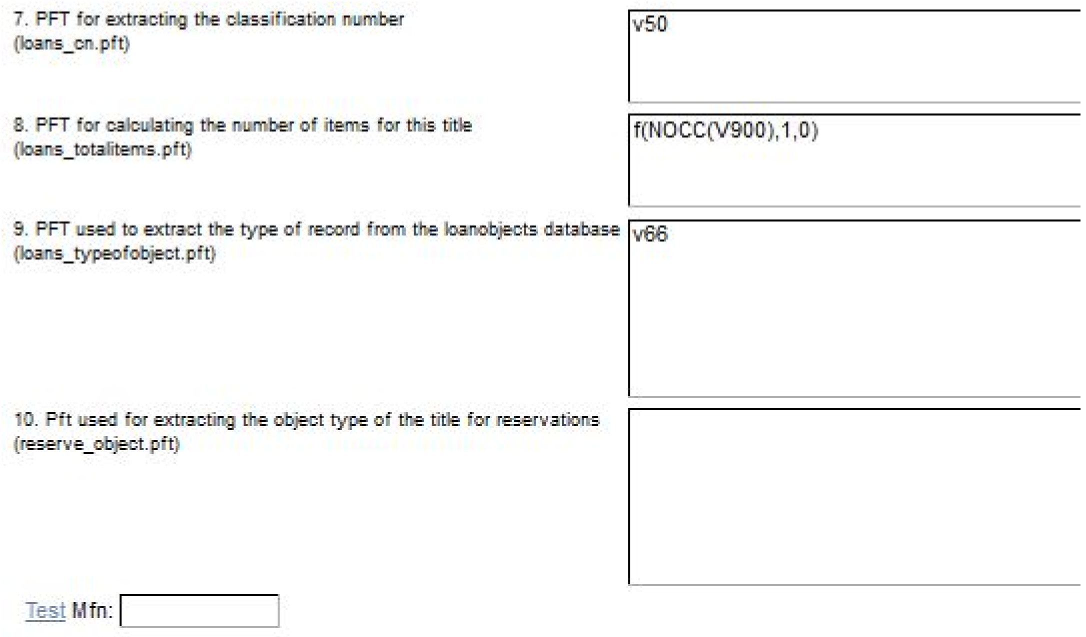
5. Prefix for the accession number (loans_conf.tab)

IN_

6. Prefix for the classification number (loans_conf.tab)

CL_

Figure 2.18. (continued)


7. PFT for extracting the classification number (loans_cn.pft)

```
v50
```

8. PFT for calculating the number of items for this title (loans_totalitems.pft)

```
f(NOCC(V900),1,0)
```

9. PFT used to extract the type of record from the loanobjects database (loans_typeofobject.pft)

```
v66
```

10. Pft used for extracting the object type of the title for reservations (reserve_object.pft)

Let's briefly explain each box of this worksheet. Each box results in a PFT stored into a 'loans' subfolder of the related database folder itself.

- PFT for displaying the record from the items database (loans_display.pft) : the PFT to apply onto the bibliographic catalog database for showing the object. In a MARC-like catalog this could be something like in the example given, taking the title from v245 and the author from v100 and adding some typical prefixes.

- PFT for storing the item in the Transactions database (loans_store.pft) : this PFT defines how the v100 of the transactions database will identify the title of the object involved. Since this field is a subfielded field (but which subfields and subfield-identifiers is free to chose), one has to create the subfield identifiers followed by their content, e.g. '^a'v100 will first create the subfield ^a and then put the value of v100 in it.

- PFT used to display the item from the Transactions database (loans_show.pft) : this is the opposite use of the previous ; once you have stored the data in v100 of the transactions database (see item above), how will the system display data from these records when dealing with v100 ? E.g. the statement 'v100^a' will add the author's name to the display.

- PFT for extracting the accession number (loans_inventorynumber.pft) : since the barcode or identifier has to be taken from a field which is not predefined (as it is in the LOANOBJECTS database), the administrator needs to explain exactly where to take that identifier from in the bibliographic record, wiht a PFT, e.g. 'v900^i' would tell the system the subfield i of v900 in the catalog is actually the identifier of each copy.

- Prefix for the accession number (loans_conf.tab) : since ABCD will lookup the barcode or accession number from the Inverted File, the prefix used to index it should be given here. Actually the pre-defined prefix should be 'IN_'.

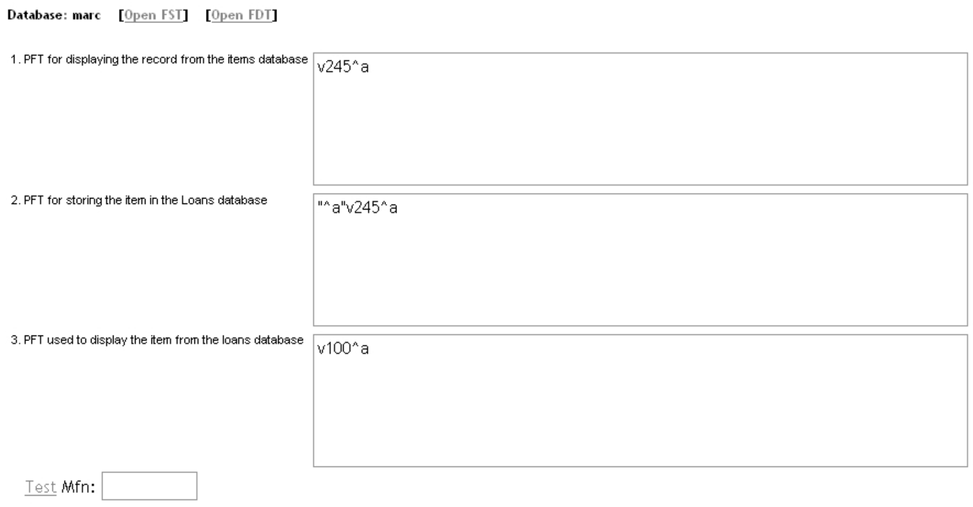
- Prefix for the control_number or classification number (loans_conf.tab) : since in many catalog databases the classification number serves as the most unique identifier of a book, this field can be used for retrieving the title if no control_number is used, otherwise use the Control_number field. Here the prefix used to index the classification number is to be given optionally, e.g. 'CL_' or in the case of the Control_number field we use the mandatory 'CN_' prefix.

- In combination with the previous item : the PFT to extract the classification number from the catalog record, e.g. 'v50^a'; if you would want to combine several subfields of such a classification field to one single identifier just concatenate them separated with dots : e.g. ' v50^a,|.|v50^b,|.|v50^c,|.|v50^d '

- PFT for calculating the number of items the title has (loans_totalitems.pft) : since e.g. for the reservation function the system needs to know how many copies are available for the loans-system, a PFT is needed which results in that number. In the LOANOBJECTS database this is the number of occurrences of v959, so it can be pre-defined in the ABCD-scripts, but if this has to be deduced from the catalog of in principle any structure, the PFT will explain how to do this for your exact situation. Suppose the v900 is used for the purpose of identifying copies, it could be the following format, which simply calculates the number of occurrences of that field and formats it as a string : f(NOCC(V900),1,0)
- PFT used to extract the type of record from the items database (loans_typeofobject.pft) : again this information is at a fixed position in LOANOBJECTS (v959^o) but since this is not the case in your catalog, it has to be explained to your circulation system in order to select the correct policy for that object type. Just put the (sub-)field which contains this information, e.g. v900^o. Contrary to what would be expected no need to indicate that this is a repeated field. Remember that since this is again a real PFT, it can also use logical rooting to deduce the object type from different source-fields if necessary, e.g. if the type has to be deduced from the bibliographic level field e.g. V4, one could use a statement like : if s(V4):'M' then 'B' else 'R' fi, which means : if the record is a 'M'onograph-level record, then put 'B'(ook) as the type, otherwise put 'R'eserved or something like this.
- [Prefix for extracting the object type for reservations (reserve_object.pft) : not used anymore, will be removed]

The worksheet for defining the Source Database parameters when the LOANOBJECTS database is used, is simply a subset of the previous worksheet, since several parameters are fixed in the structure of that LOANOBJECTS database.

We can best illustrate this by giving the example of the CEPAL-database of the ABCD-model application :



Database: marc [Open FST] [Open FDT]

1. PFT for displaying the record from the items database	v245^a
2. PFT for storing the item in the Loans database	"^a"v245^a
3. PFT used to display the item from the loans database	v100^a

Test Mfn:

This form shows the information needed for the loans system, e.g. which prefix is used in the index for the accession number or which print formats (PFT's) to use to produce the data in the loans screens.

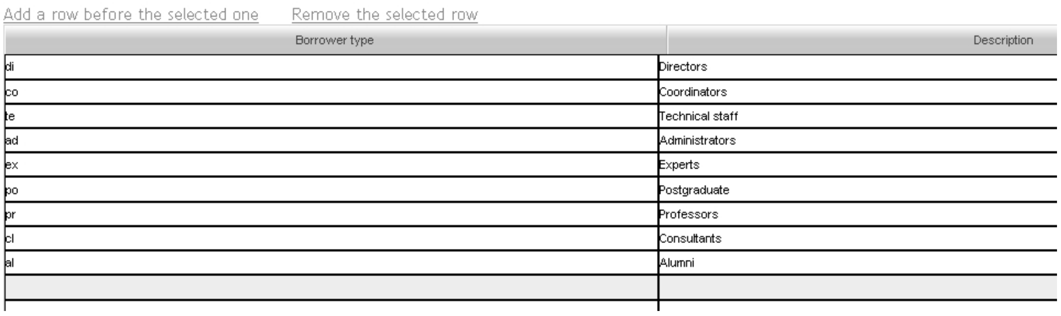
The power of the Formatting Language can be applied here, of course. For example, instead of the rather dumb example above here as the 'PFT to be used to extract the type of record', one could define a different type (with consequently different loan parameters for the type) according to some conditions, e.g. the date, the month etc. So an object which is a normal loans object could be changed into 'special material' during the exams period etc. The ISIS Formatting Language provides most necessary functions (e.g. Date() with substring extraction) for this purpose.

6.3.2.2. Borrower types and users data

'Users' here are in fact library patrons who use the circulation system. They need to belong to a specific category which defines the policy (e.g. how many books can the user take ?) and as with the 'source database' it has to be defined from where the identifying data can be taken, since ABCD allows this to be any database with any structure.

First let's define the possible user-types.

After clicking on the corresponding entry of that menu, using a table ABCD will then present the defined borrowers types and allows to add any number of more such types :

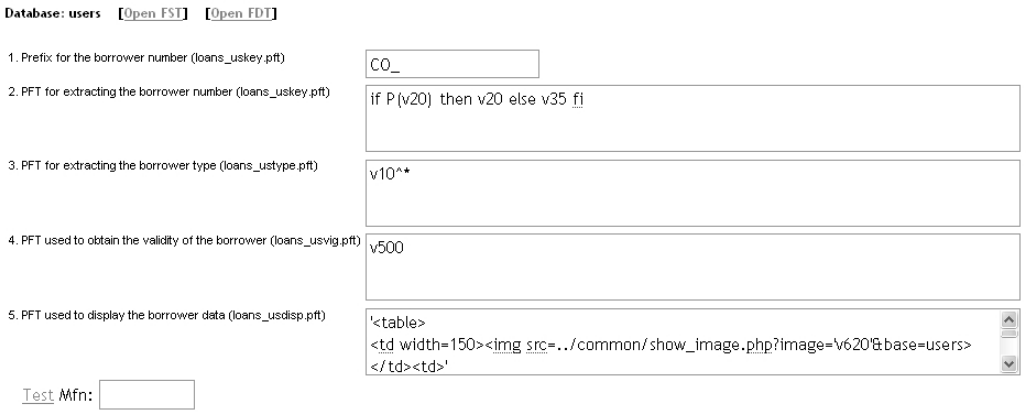


Add a row before the selected one   Remove the selected row

Borrower type	Description
di	Directors
co	Coordinators
te	Technical staff
ad	Administrators
ex	Experts
po	Postgraduate
pr	Professors
cl	Consultants
al	Alumni

Here we only show part of the table, but in fact the interface will always offer a few more empty lines to add more types, and lines with a type can also be added in between existing rows. The left column, as always, represents the internal code, the right one the name as displayed in the interface (the menu's).

Once the possible types have been defined, the system needs to know how the actual type of the borrower can be identified. The worksheet for this is similar to the one for Source Database data :



Database: users [Open FST] [Open FDT]

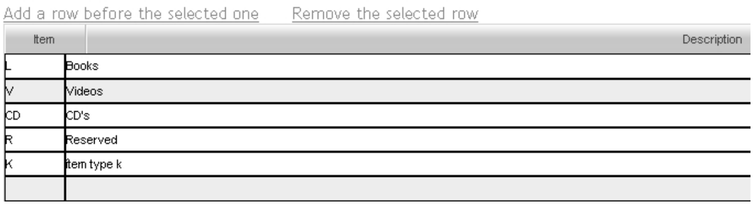
1. Prefix for the borrower number (loans_uskey.pft)	CO_
2. PFT for extracting the borrower number (loans_uskey.pft)	if P(v20) then v20 else v35 fi
3. PFT for extracting the borrower type (loans_ustype.pft)	v10^*
4. PFT used to obtain the validity of the borrower (loans_usvig.pft)	v500
5. PFT used to display the borrower data (loans_usdisp.pft)	'<table> <td width=150><img src=../common/show_image.php?image='v620'&base=users> </td><td>'

Test Mfn:

Instead of simply taking the 'value of Field 10' (v10) to define the borrower type, one could put a more sophisticated Formatting Language statement here in order to make the status of the borrower even dynamically dependent on other conditions (again : date, but also other conditions can be defined).

6.3.2.3. Objects types definition

This is the same as for the patrons types, but since only one object-element is needed to define the loan policy, we need only to create/edit the corresponding table, which will result in the file 'typeofusers.tab' in the correct language subfolder of the 'DEF' subfolder of the 'CIRCULATION' database :



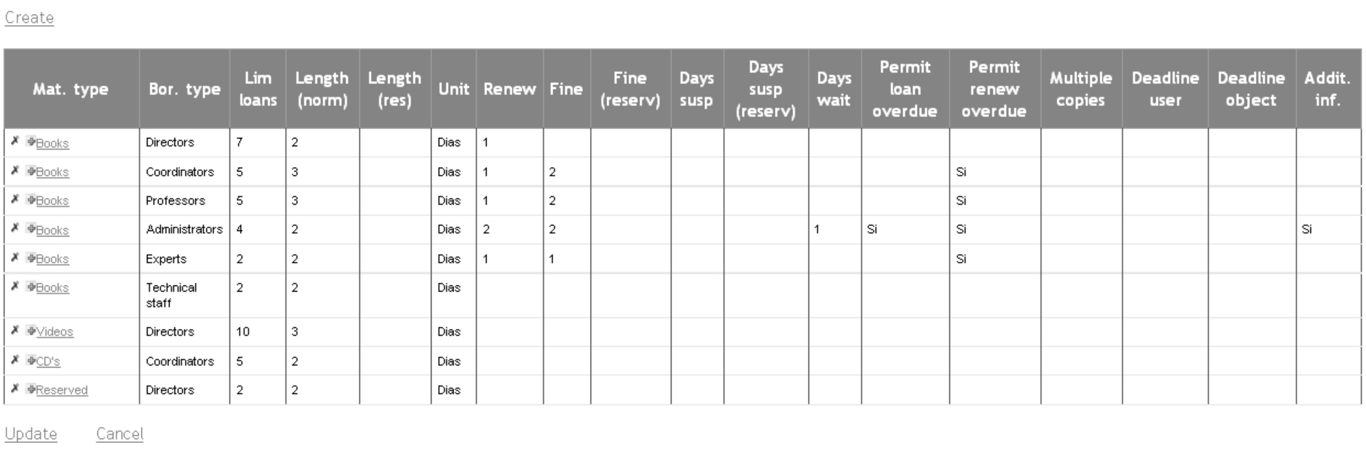
Add a row before the selected one   Remove the selected row

Item	Description
L	Books
V	Videos
CD	CD's
R	Reserved
K	Item type k

Don't forget to re-save your work when having made any additions or editions.

6.3.2.4. The loan policy definition

With both data on 1) patron and 2) object combined, the system now can select one of the defined sets of rules or 'policies', which are to be entered into the system using the following matrix :



Create

Mat. type	Bor. type	Lim loans	Length (norm)	Length (res)	Unit	Renew	Fine	Fine (reserv)	Days susp	Days susp (reserv)	Days wait	Permit loan overdue	Permit renew overdue	Multiple copies	Deadline user	Deadline object	Addit. inf.
✗ Books	Directors	7	2		Dias	1											
✗ Books	Coordinators	5	3		Dias	1	2						Si				
✗ Books	Professors	5	3		Dias	1	2						Si				
✗ Books	Administrators	4	2		Dias	2	2				1	Si	Si				Si
✗ Books	Experts	2	2		Dias	1	1						Si				
✗ Books	Technical staff	2	2		Dias												
✗ Videos	Directors	10	3		Dias												
✗ CD's	Coordinators	5	2		Dias												
✗ Reserved	Directors	2	2		Dias												

Update Cancel

The first column is presented as a hyperlink, which opens the related row into a separate window for easier editing of all paramaters involved. Most parameters are self-explanatory. Note that there are different values for parameters for items with or without running reservations and both fines and suspension days can be used as 'punishments' for late returns. With already quite a number of such parameters this circulation system is not simply 'basic' indeed.

As can be seen, lots of parameters are stored and used in the decision-making process of each transaction (e.g. is this user allowed to loan this type of material, how many of them, for how long, what is the fine for late return etc.). Units can be either days or hours and the calculation of 'number of days elapsed' will be based on a calendar function (see below).

Some remarks on the otherwise obvious parameters are given here :

- if all cases when no maximum is desired (number of loans, expiry date...) : simply leave the value for the parameter blank
- the overall 'maximum number of allowed loans' overrides the sum of the numbers allowed per object type, if this sum is higher than the overall maximum
- once a user has reserved a specific object, no more reservations on the same object can be taken by that user

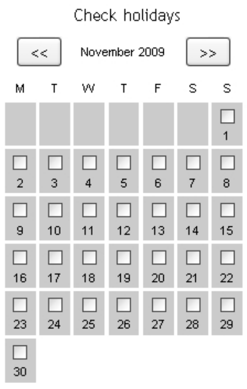
Note

It is important to note here the link to 'Update' the table : not only when one has edited the row in the sub-window this window has to be stored into the matrix, but at the end also the matrix has to be stored into the system itself !

6.3.2.5. Other circulation configuration : calendar, currency

Configuration of the loans system continues with two more options :

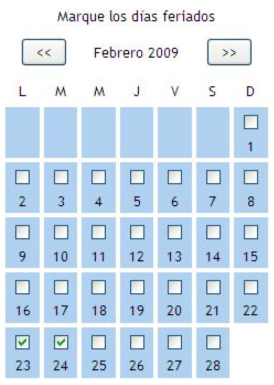
- definition of the currency , fine unit, date format and working days/hours :



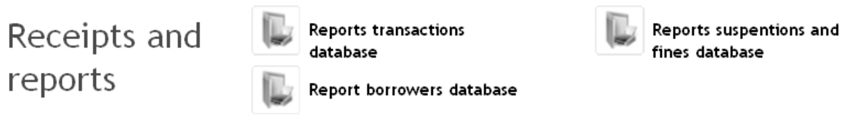
Caution

In order for the loans system to work well, don't leave this 'working days' calendar empty ! If no working hours or days at all are defined the creation of a loans records will fail as no return date can be calculated.

- definition of the holidays (non-working days) in the calendar , where simply the holidays need to be indicated on each month's map :

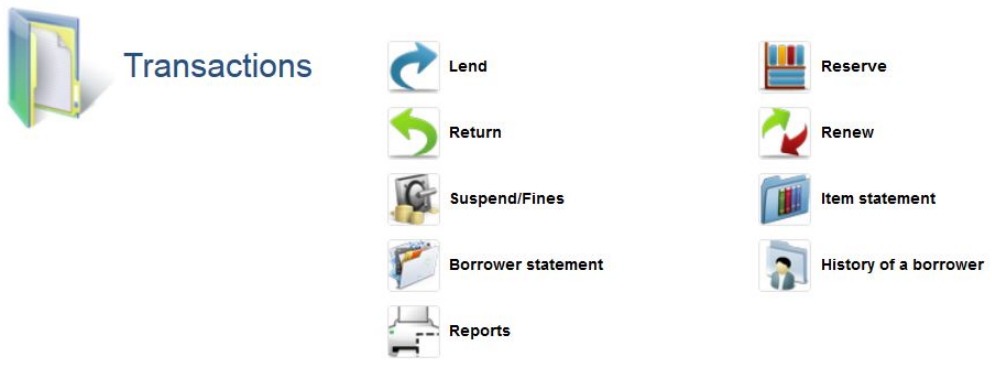


- generation of Loan reports : for each database in Loans (transactions, borrowers and suspensions) a set of PFT's (as these are in fact the reports in ABCD) can be generated using the same interface as used for other PFT's (e.g. in the Database Administration module). As the procedure is fully identical we won't repeat the steps here, please refer to the dedicated section in the Database Administration module chapter.


Receipts and reports
Reports transactions database
Reports suspentions and fines database
Report borrowers database

6.3.3. Transactions : loan, returns, reservation, renewals, fines/suspensions

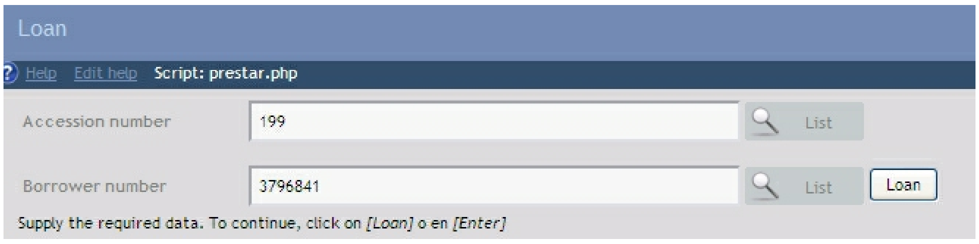
Most of the transactions themselves are rather easy to understand. Efficiency is the key here : mostly the system simply needs one or two bar-codes to be scanned in, and pressing a button 'go' to store the transaction. A list of possible transactions is shown in the transactions menu of ABCD :



Let's discuss each of them now.

6.3.3.1. Issuing a loan on an object

This page offers identification boxes (in which either the barcode or identifier can be input directly or selected from a list) for both the object and the user, the two crucial elements of any loans transaction.

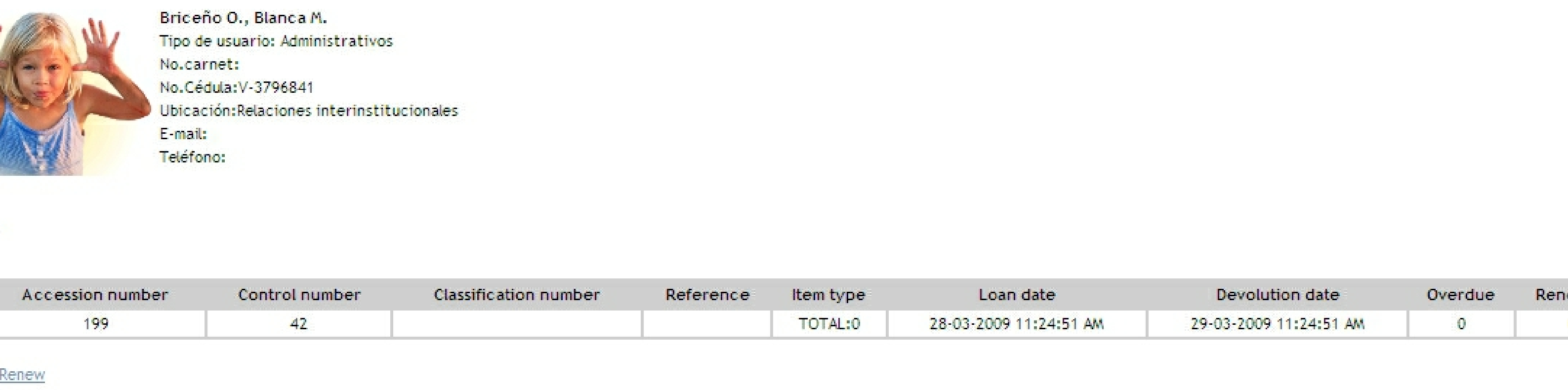


Script: prestar.php

Accession number	199
Borrower number	3796841

Supply the required data. To continue, click on *[Loan]* o en *[Enter]*

After having identified both elements, a simple click on the 'loan'-button will actually create the transaction (into the 'loans' database). The user information is shown and all loans transactions related to the users will be listed in a table, where one or more transactions can be selected for immediate editing (e.g. returning or renewing the loan).



Briceño O., Blanca M.
Tipo de usuario: Administrativos
No.carnet:
No.Cédula:V-3796841
Ubicación:Relaciones interinstitucionales
E-mail:
Teléfono:

Accession number	Control number	Classification number	Reference	Item type	Loan date	Devolution date	Overdue	Ren
199	42			TOTAL:0	28-03-2009 11:24:51 AM	29-03-2009 11:24:51 AM	0	

6.3.3.2. Reservation of an object

The reservation function is new in ABCD Central as from v1.2t. Reservation parameters have been added to the policy of each user/object categories combination and if the rules to be observed by the system have been defined correctly, reservation by the librarian will go as follows.

Note

Reservation by end-users from the iAH and MyLibrary screens are discussed separately.

The reservation function of ABCD Central is based on a 'normal' ISIS-database 'reserve' with the following fields defined :

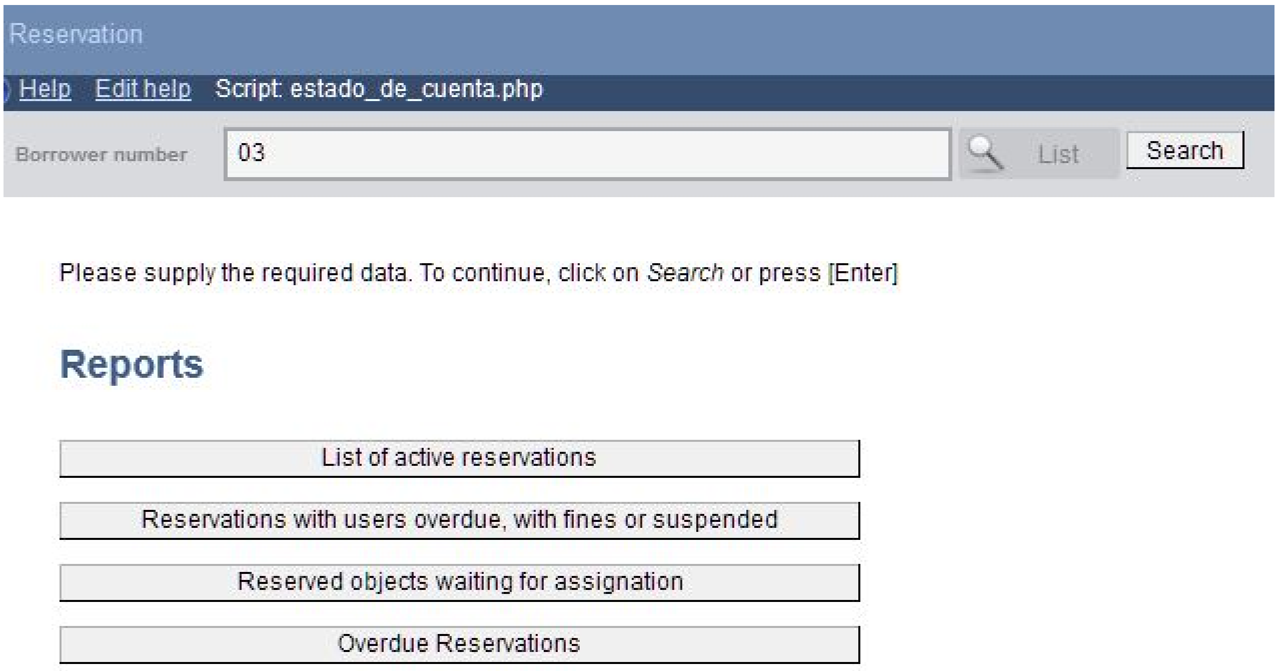
- v1 : status of the reservation (0 : active reservation queuing, 1 : canceled by administrator, 2 : reservation canceled because waiting time elapsed, 3 : reservation assigned and 4 : reservation converted into loan with date noted in date-field)
- v10 : user ID
- v12 : user category, taken from the list in %path_database%circulation/def/es/typeofusers.tab
- v15 : name of the catalog database in which the reserved item has its bibliographic data
- v20 : Control Number in the catalog
- v30 : date of creation of reservation
- v31 : time of creation of reservation
- v32 : operator who creates the reservation
- v40 : expiry date of the reservation, calculated by the system taking into account the policy-parameters and calendar

- v50 : classification number of the title
- v60 : date when the reservation is assigned (when a copy of the reserved title is returned and becomes available)
- v130 : date of cancellation
- v131 : time of cancellation
- v132 : operator of cancellation
- v200 : date of moving reservation to loan

The files used in the reservation system - which can be edited but with caution ! - are :

- tbreserve.pft : general format of the reservation records in a table format (don't change the first 4 columns)
- tit_reserve.tab : the 'headings' of the columns used in tbreserve.pft (adjust according to changes made in the columns 5..n of tbreserve.pft)
- reserve_01.pft : list of reservations in queue (v1=0), list of assigned reservations (v1=3), list of cancelled (or expired) reservations
- tit_reserve_01.tab : the 'headings' of the columns used in reserve_01.pft

When clicking on the reservations transaction, the screen following will first require to identify the user, after which in the user record the icon for 'reserve' can be clicked.


Reservation

Help Edit help Script: estado_de_cuenta.php

Borrower number: 03

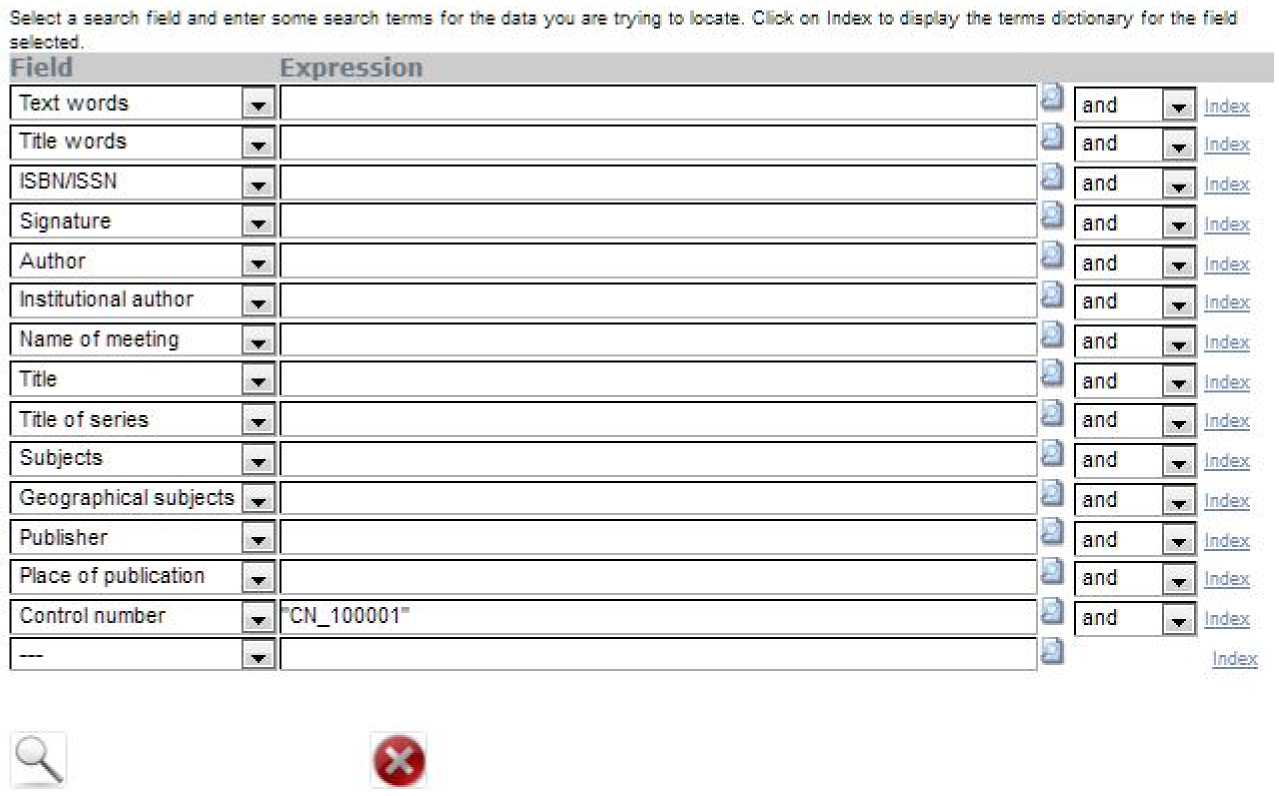
Please supply the required data. To continue, click on *Search* or press [Enter]

Reports

List of active reservations
Reservations with users overdue, with fines or suspended
Reserved objects waiting for assignation
Overdue Reservations

This screen also offers direct options to view some reports, which we believe are clear in themselves and will not be discussed now.

When 'searching' a user has identified the user involved, the administrator for the reservation has to select a database form the list of databases in which loanobjects can reside. As a result of selecting the database, the standard 'search form' for that database will be shown, with which a specific object can be identified using any of the search options provided by ABCD Central for that database. The difference is that no real search can be executed but only the selected search statement, once put in the grid either directly or from the index-consultation, will identify the single object for reservation and having clicked on the 'search'-icon (the magnifying glass), after which for that user the object can be reserved if the rules of the system allow. Obviously the selection made in the search-grid in this case should always point to one single catalog entry, from which the first available copy will be automatically selected for reservation.


Select a search field and enter some search terms for the data you are trying to locate. Click on Index to display the terms dictionary for the field selected.

Field	Expression		
Text words		and	Index
Title words		and	Index
ISBN/ISSN		and	Index
Signature		and	Index
Author		and	Index
Institutional author		and	Index
Name of meeting		and	Index
Title		and	Index
Title of series		and	Index
Subjects		and	Index
Geographical subjects		and	Index
Publisher		and	Index
Place of publication		and	Index
Control number	"CN_100001"	and	Index
---			Index

The best identification is to selected from the control_number, with which the system will check if there are still copies for that 'loanobject' available. If so, this will be communicated and no further reservation action can be taken as the object can be issued on loan by normal procedures.

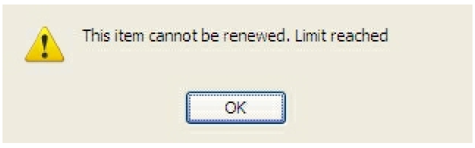
If no more loanable copies are available (i.e. the number of copies minus the sum of the number of loaned copies and the number of copies-required-as-a-minimum in the reading room), the system will check the parameters for the transaction (user not expired, no overdue loans if that is set as a rule etc.) and if a reservation is allowed, the system will put the next available copy under reservation for a given number of days, during which the user has to come to pick up the copy. From then on the transaction will be changed from status 'reserved' to 'on loan' and normal loan rules and procedures apply.

6.3.3.3. Returning an object

Here only the object returned needs to be identified and with a simple click the transaction record in the loans database will note the fact that the object has been returned. The loaned object can also be returned from the table in the borrowers' statement, then the transaction will be removed from the table. [!!]

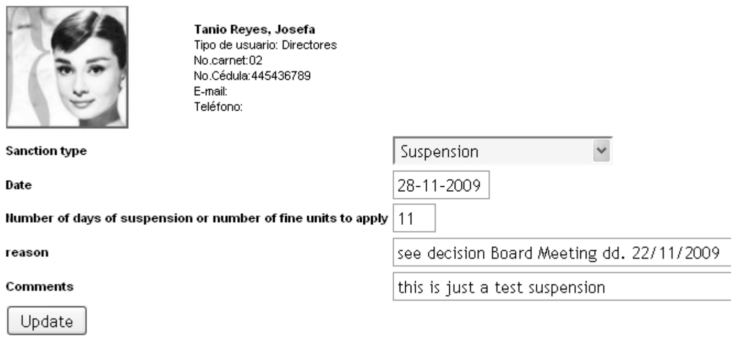
6.3.3.4. Renewing a loan

This is a simple continuation of a running loan, but dependent again on specified rules as on whether the object has not been reserved by someone else and the user requesting the renewal has no pending fines etc. When consulting the list, only the objects on loan will be listed. In case all conditions for renewal are fulfilled, the transaction will be granted and listed, if not a warning or error message will be shown, e.g.


This item cannot be renewed. Limit reached

6.3.3.5. Fines and suspension of users

Fines and suspension of users are offered in the same ABCD-page :



Tanio Reyes, Josefa
Tipo de usuario: Directores
No.carnet:02
No.Cédula:445436789
E-mail:
Teléfono:

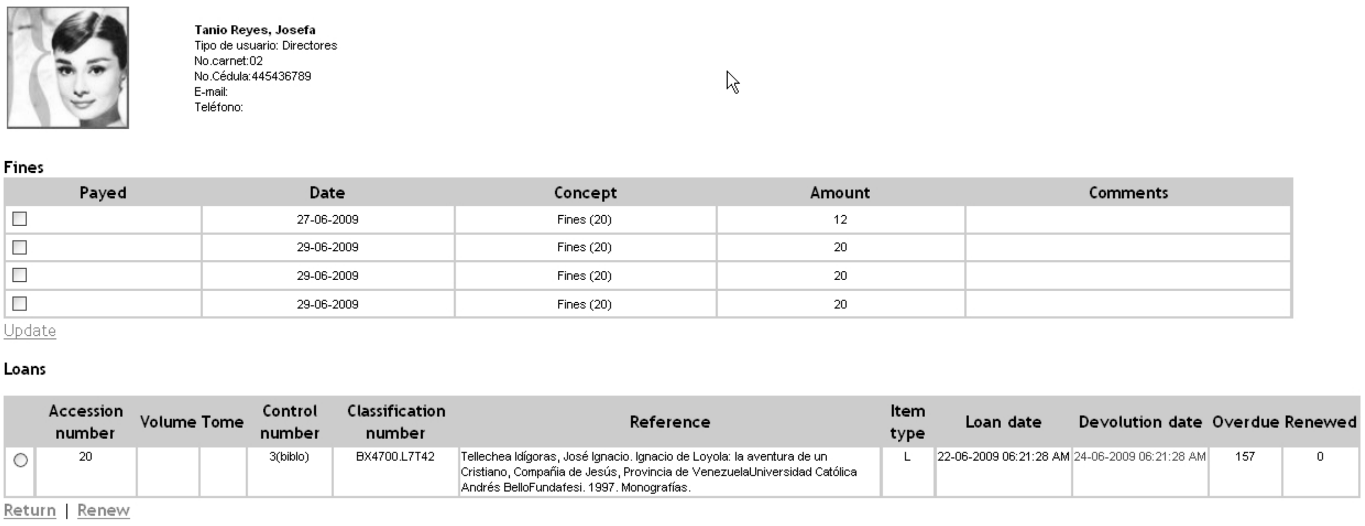
Sanction type	Suspension
Date	28-11-2009
Number of days of suspension or number of fine units to apply	11
reason	see decision Board Meeting dd. 22/11/2009
Comments	this is just a test suspension

For the selected user the following fields can be filled in : the type of sanction (fine or suspension), the date, the number of days or the amount of the fine, the reason (motivation) and any comments.

6.3.3.6. The borrower statement

From this screen all information on a borrower or user is displayed, giving also direct access to other functions where only an access from the object was given, e.g. to allow renewal from the borrower's identification instead of the object.

[!!] Interesting to note that borrowers not only can be selected from the list of borrowers but also from the inventory-number of the objects loaned by the borrower.



Tanio Reyes, Josefa
Tipo de usuario: Directores
No.carnet:02
No.Cédula:445436789
E-mail:
Teléfono:

Fines

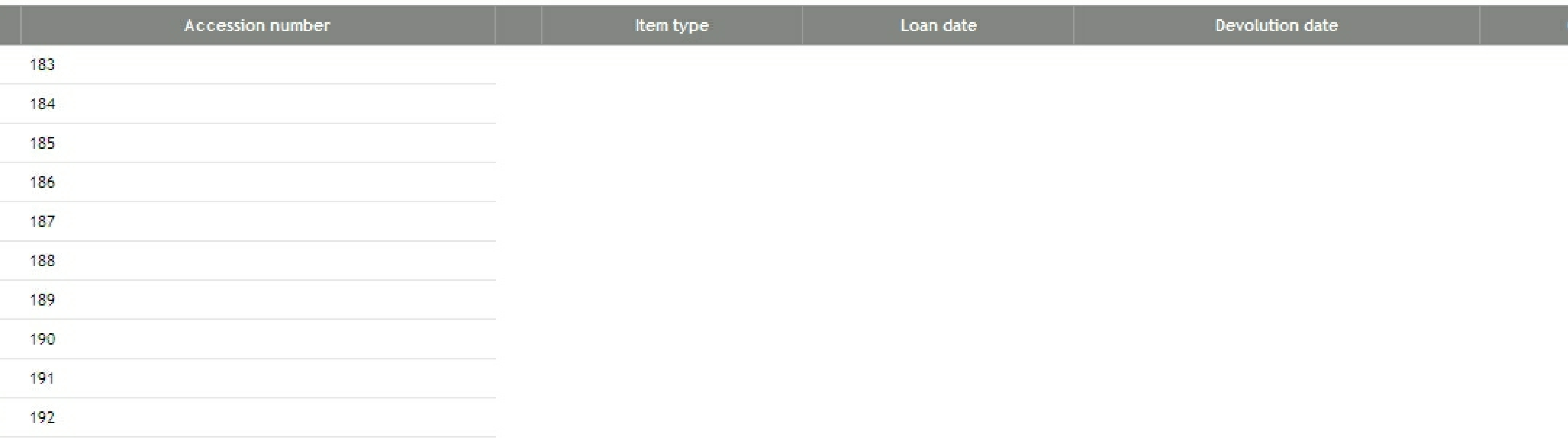
Payed	Date	Concept	Amount	Comments
☐	27-06-2009	Fines (20)	12	
☐	29-06-2009	Fines (20)	20	
☐	29-06-2009	Fines (20)	20	
☐	29-06-2009	Fines (20)	20	

Loans

	Accession number	Volume Tome		Control number	Classification number	Reference	Item type	Loan date	Devolution date	Overdue	Renewed
○	20			3(biblo)	BX4700.L7T42	Tellechea Idígoras, José Ignacio. Ignacio de Loyola: la aventura de un Cristiano, Compañia de Jesús, Provincia de VenezuelaUniversidad Católica Andrés BelloFundafesi. 1997. Monografías.	L	22-06-2009 06:21:28 AM	24-06-2009 06:21:28 AM	157	0

6.3.3.7. State of an item

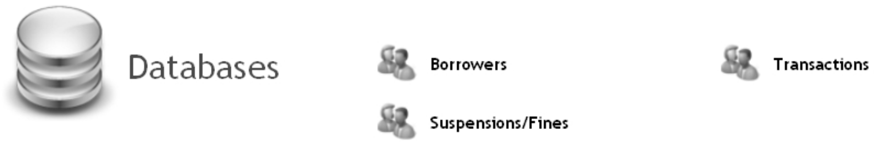
As with the borrower's statement, an overview (history) of all loans of this given item or object can be retrieved here.



Accession number	Item type	Loan date	Devolution date
183			
184			
185			
186			
187			
188			
189			
190			
191			
192			

6.3.4. Databases in the loans module

The following databases are used in the Central Loans module :



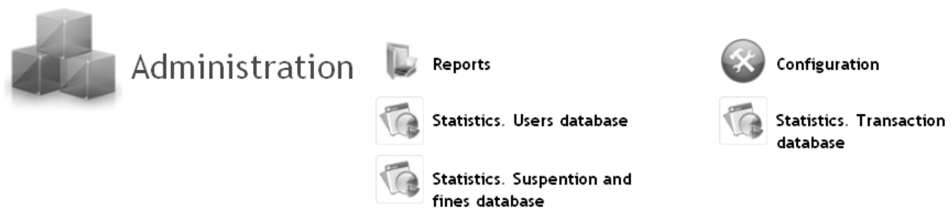
For each of these databases, whose names explain their purpose, an interface is presented which lists the records with a search function and edit or delete buttons.

The transactions database records the actual events in the loans system. ABCD opts to keep this database as 'mean and lean' (i.e. compact) as possible, without duplicating data e.g. from the bibliographic databases. This database will be rather 'dynamic' with many movements, and since ISIS is not at its best in such environment (for which simple tables with fixed structures would be more suited), we need to keep its structure as compact as possible, using the REF-function of ISIS to 'loan' data from other databases, e.g. the bibliographic data.

So this will allow the librarian to directly interact with the records of the borrowers (e.g. to create new library users), the transacions (e.g. to check a loans record), reservations and suspensions or fines. In this last database the librarian could interfere with existing due fines and suspensions of users in case of a necessity to do so by-passing the rules - take care !

6.3.5. Loans administration

This third and last section of the loans module not only offers the loans configuration option (discussed above here) but also gives access to the Statistics module and the 'Reports' option (which are also discussed in the Database Management part of this manual as they use the same principles and techniques). This module will create all types of output documents, e.g. overdue alerts, confirmations of loans etc.


Administration
Reports
Configuration
Statistics. Users database
Statistics. Transaction database
Statistics. Suspention and fines database

As for the reports of the circulation module, there are some observations to take note of :

- in each language-folder for the databases 'trans', suspml and reserve (the three databases involved with transactions) a file 'outputs.lst' keeps track of the available reports, one line for each report specifying with '|'separated values :
  - the report code/name

- the actual file name (PFT)
- the column headers (separated by '#')
- (optional) sorting criterion (e.g. : `ref([`users`]l([`users`]`CO_`v20),v30)`will sort on user-name),
- search expression (with the following pre-defined ones : '*TR_P$*' lists all active loans, '*TR_X$*' lists all returns, '*RE_M and ST_0*' lists all active fines, '*RE_M and ST_1*' lists all cancelled reservations, '*RE_M and ST_2*' lists all paid fines, '*RE_S and ST_0'* lists all active suspensions and '*RE_S and ST_1*' lists all cancelled suspensions)
- the report name
- a logical field to denote the need to request additional selection criteria, e.g. only records with today as return date or with specific user category etc.

- the file 'status.tab' (in the language-depending subfolder of the 'DEF'-subfolder of the SUSPML-database) lists all applicable sanctions
- the option 'reports' in the main Loans menu, section 'Transactions' allows to access and edit these reports. The interface allows definition of all elements listed just above here. The best way to deal with this is to study the existing examples and from there build your own - if needed. Remember that it is always a good idea to share good results (in this case : report formats) in a FOSS-community !

6.3.6. An example : implementing an AVM loans system

In order to illustrate all the above, let's try to set up a real application in the ABCD Central Loans environment : a system for circulation of AVM. All of the steps have been described above, but let's go through them one by one again with this actual 'case study'. We will follow the scenario where we want AVM to have their copy information inside the very small AVM database itself, so no LOANOBJECTS.

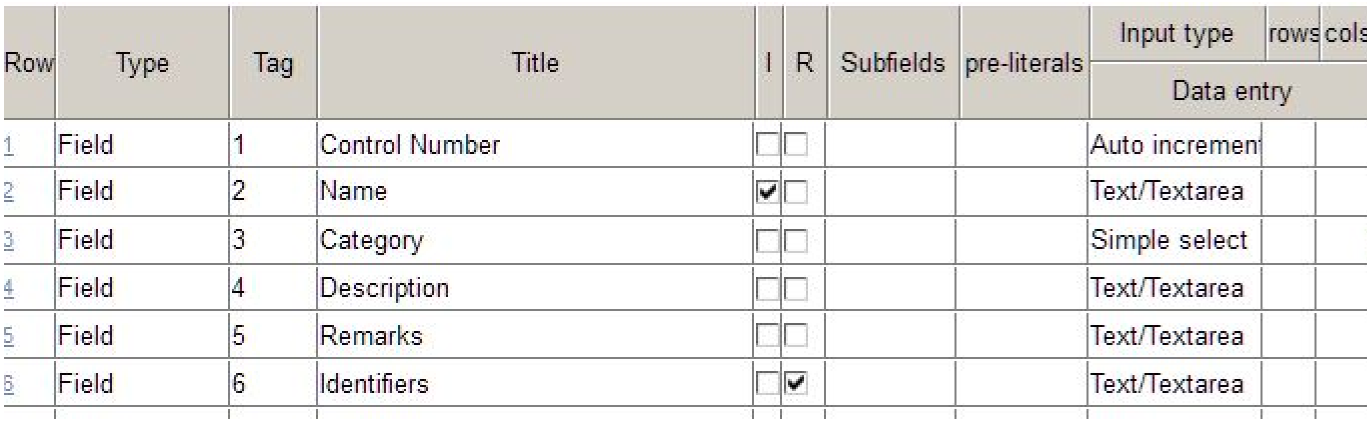
6.3.6.1. Creating the catalog

First of all, we need to create the small database which contains the AV materials. We do this following the instructions described in the section on Database Administration (creation of a new database) in this manual. All steps use the ABCD Central 'Cataloging' module.

1. Creating the Field Definition Table

   The database describing the AV materials can be as complicated as necessary, but here we keep it to its minimum to allow circulation, meaning that 1) there should be one field identifying the material itself ('control_number'), 2) one field indicating the category of AVM 3) one field to list the identifiers (barcodes) for each copy (so a 'repeatable' field) and 4) a field indicating the location of the object(s).

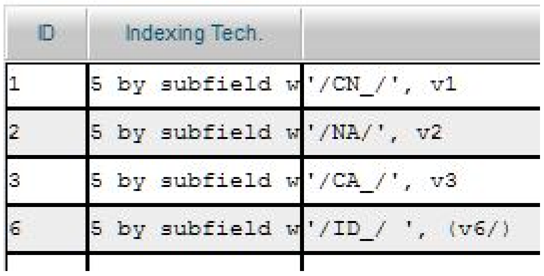
   In the FDT shown below there is only one small 'extra', i.e. a table from which the categories can be selected when creating an object. Remember that from the 'code' in this field the circulation policy will be derived in combination with the loaning user category, so keeping this well controlled with a picklist is important here.



Row	Type	Tag	Title	I	R	Subfields	pre-literals	Input type	rows	cols
								Data entry		
1	Field	1	Control Number	☐	☐			Auto incremen		
2	Field	2	Name	✓	☐			Text/Textarea		
3	Field	3	Category	☐	☐			Simple select		
4	Field	4	Description	☐	☐			Text/Textarea		
5	Field	5	Remarks	☐	☐			Text/Textarea		
6	Field	6	Identifiers	☐	✓			Text/Textarea		

2. Creating the Field Selection Table

   In the FST we have to ensure that we have at least 2 entries with specific prefixes : one for the catalog identifier (control number) and one for the copy identifier (barcode). If no control-number exists one could also use e.g. the 'classification number' if - and only if - this is unique for each catalog record. In our AVM example we use resp. 'CN_' and 'ID_'.



ID	Indexing Tech.	
1	5 by subfield w	'/CN_/', v1
2	5 by subfield w	'/NA/', v2
3	5 by subfield w	'/CA_/', v3
6	5 by subfield w	'/ID_/ ', (v6/)

3. Creating the Display Format (PFT)

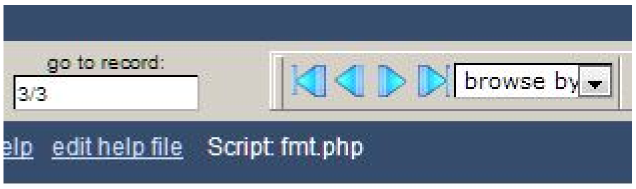
   Here we simply moved all fields (with the double arrow) to the right panel and chose 'Table' display format, after which - by clicking on 'Create Database' ABCD will create all necessary folders and files for the new database.

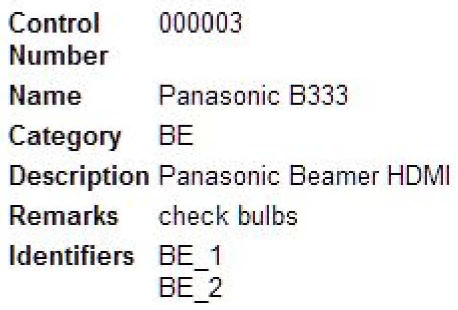
   Note

   We had to re-enter into the FDT in order to define the values for the 'AVM categories', since the first time at creating stage ABCD still has no folder in which to put the table file itself, but after database creation the option 'table' becomes available in the picklist types, and clicking on 'browse' allows creation and editing of the lists. The one category we will use for our case study is 'BE' (for beamer, of which we have two copies).

4. Entering some sample data

   Using the default data-entry form - kept quite simple with the categories menu as the only extra - we created a few records, one of these being of the 'BE(amer)' category in which we have put 2 occurrences for the 'ID'-field since we have two beamers of the same make and type BE_1 and BE_2.





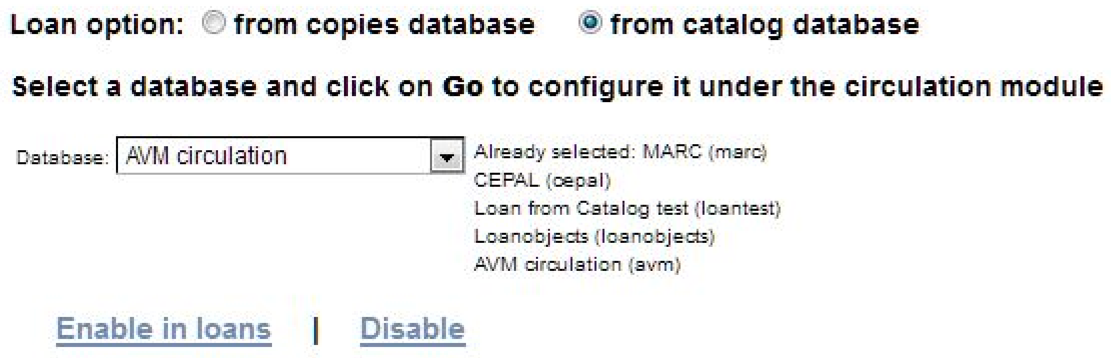
Control Number	000003
Name	Panasonic B333
Category	BE
Description	Panasonic Beamer HDMI
Remarks	check bulbs
Identifiers	BE_1 BE_2

6.3.6.2. Configuring the AVM database for circulation

Now, using the ABCD Central Loans module, we will have to configure our new database AVM for use as a 'catalog' in the circulation system. Here are the steps to be followed :

1. Defining AVM as a source database

When we go to the bottom part of the main circulation menu and select the 'configuration' option, only three of the options there need to be addressed, the first one being defining the exact formats to 'know' the copy information, in this case from the AVM database itself (no LOANOBJECTS used).

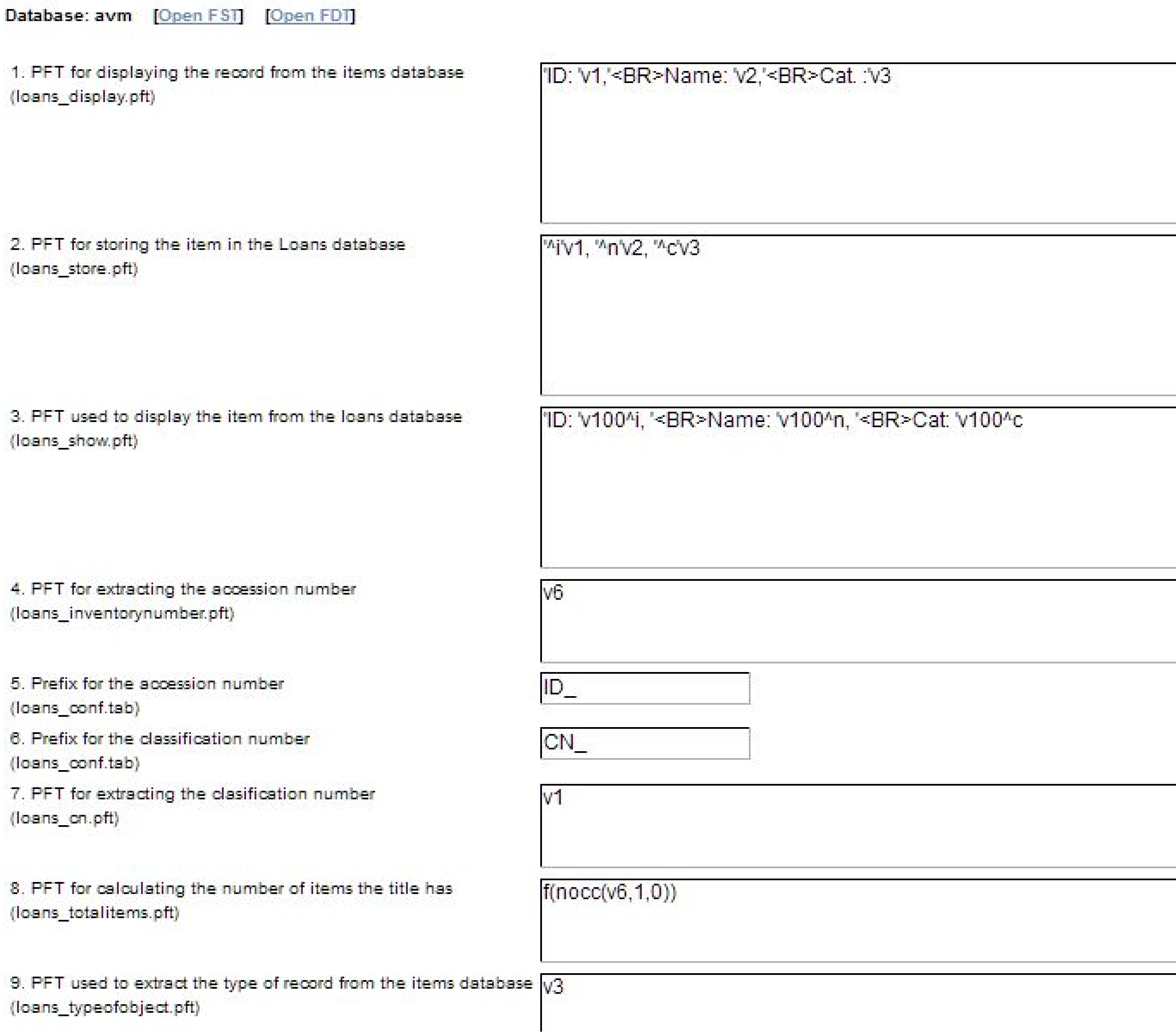

Loan option: from copies database [✓ from catalog database]

Select a database and click on Go to configure it under the circulation module

Database: AVM circulation

Already selected: MARC (marc)
CEPAL (cepal)
Loan from Catalog test (loantest)
Loanobjects (loanobjects)
AVM circulation (avm)

After selecting 'AVM circulation' as the source database and clicking on 'Enable in loans' we get the full form for extraction of copy information, full because ABCD cannot know automatically how the database was designed to contain copy information, so in comparison with the worksheet for 'LOANOBJECTS', some more fields are to be filled in.



Database: avm [Open FST] [Open FDT]

1. PFT for displaying the record from the items database (loans_display.pft)	'ID: 'v1,'<BR>Name: 'v2,'<BR>Cat. :'v3
2. PFT for storing the item in the Loans database (loans_store.pft)	'^i'v1, '^n'v2, '^c'v3
3. PFT used to display the item from the loans database (loans_show.pft)	'ID: 'v100^i, '<BR>Name: 'v100^n, '<BR>Cat: 'v100^c
4. PFT for extracting the accession number (loans_inventorynumber.pft)	v6
5. Prefix for the accession number (loans_conf.tab)	ID_
6. Prefix for the classification number (loans_conf.tab)	CN_
7. PFT for extracting the clasification number (loans_cn.pft)	v1
8. PFT for calculating the number of items the title has (loans_totalitems.pft)	f(nocc(v6,1,0))
9. PFT used to extract the type of record from the items database (loans_typeofobject.pft)	v3

Some brief comments on each field here :

- PFT to display the copy information : 'ID: 'v1,'<BR>Name: 'v2,'<BR>Cat. :'v3

  This is a very normal PFT to print the main three fields of the description of the object, i.e. control number (v1), name of the object (v2) and its category (v3).

- PFT for storing the item in the Loans database : '^i'v1, '^n'v2, '^c'v3,'^b'v6

  Here we create four subfields (^i, ^n, ^c and ^b) for resp. storing into v100 of the transactions database the control_number, the name and the category of the object and the barcode(s).

- PFT for extracting the accession number : v6

  Since we simply store the barcodes (or accession numbers of the physical copies) in v6, we simply put that v6, without making it repeatable (the system knows it is).

- Prefix for the accession number : ID_

  The prefix ID_ is used in our system to find the physical identifiers of the copies (mostly barcodes).

- Prefix for the accession number : CN_

  In case of using control numbers we index these with the prefix CN_. If no control numbers are used, and if they are unique for each record, we could use another field like e.g. classification number but using the same prefix.

- PFT for extracting the classification number : v1

  In addition to the prefixed used for indexing, ABCD needs to know where the get the control number (or classification number if used) from, in our case it was simply the first field or V1.

- PFT for calculating the number of items the title has : f(nocc(v6,1,0))

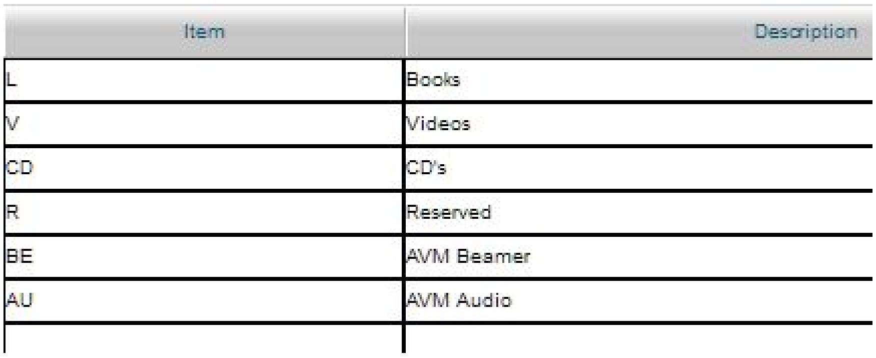
  Since v6 is the field containing, as repeats, the barcodes for each physical copy, the format to calculate the number of copies is f(nocc(v6,1,0)).

- PFT used to extract the type of record from the items database : v3

  Finally the field containing the object category in our database was defined as field with tag 3, so : v3.

2. Defining the AVM item types

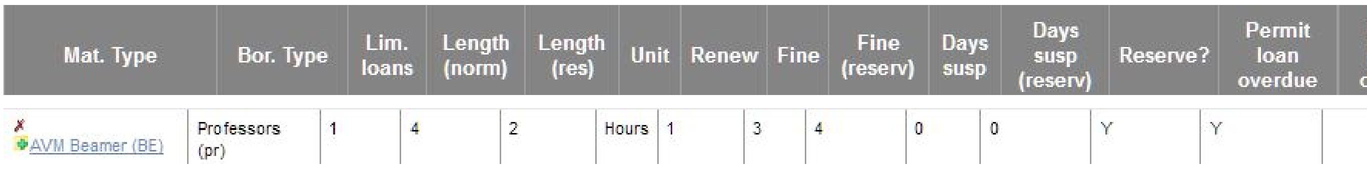
   Next in the circulation configuration - skipping the two options dealing with the borrowers database as not relevant here - we reach the definition of the item types. Here we simply added some specific AVM types like e.g. audio or beamer.



Item	Description
L	Books
V	Videos
CD	CD's
R	Reserved
BE	AVM Beamer
AU	AVM Audio

3. Defining the AVM loan policy

   The last action to be taken is to define for each users-category the exact parameters to be applied in the rules of the loans system. For our case we added (just) option to define the loan-rules for a professor taking a beamer on loan, therefore creating the policy 'BE-PR' :



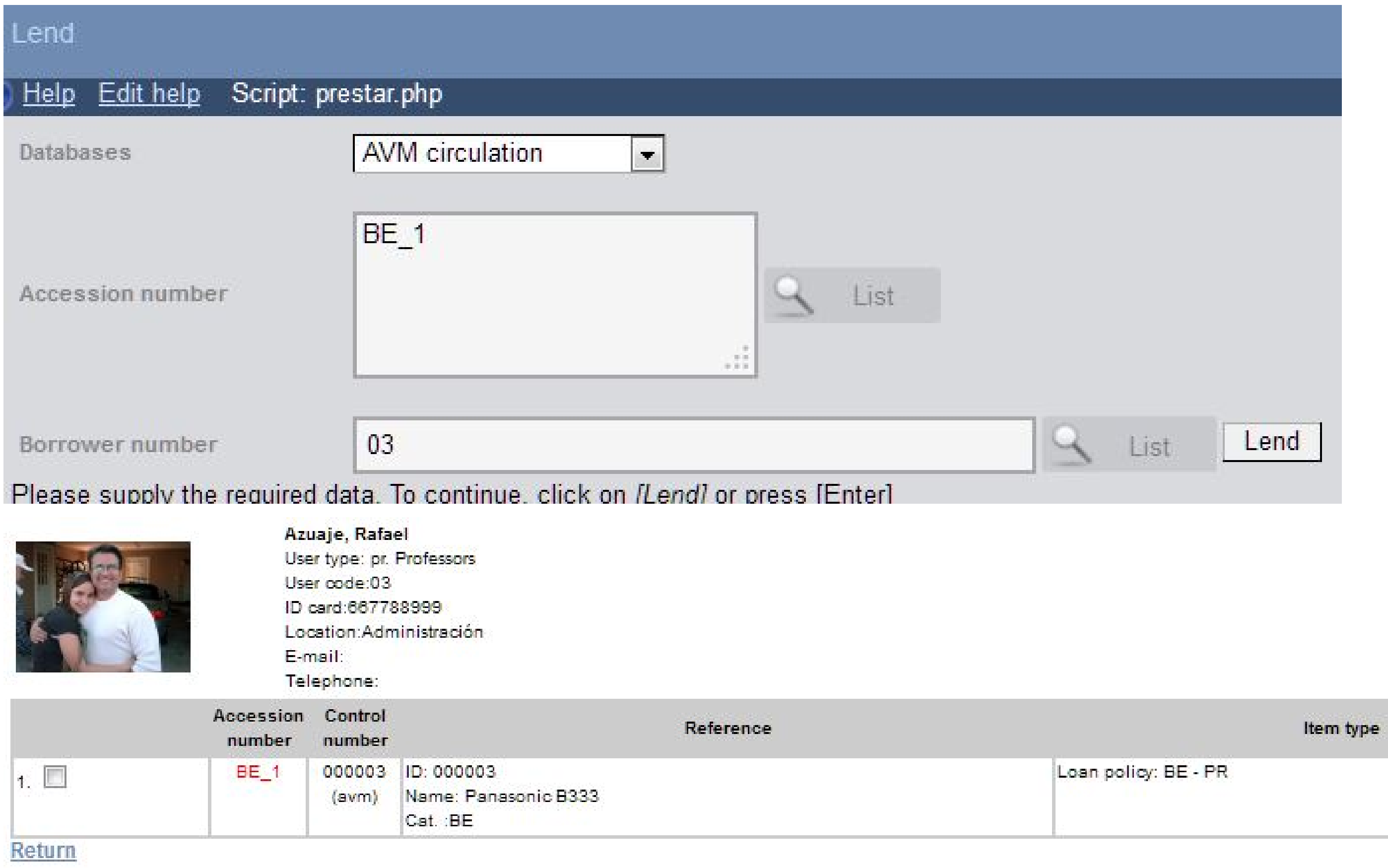
Mat. Type	Bor. Type	Lim. loans	Length (norm)	Length (res)	Unit	Renew	Fine	Fine (reserv)	Days susp	Days susp (reserv)	Reserve?	Permit loan overdue
✗ ✚ AVM Beamer (BE)	Professors (pr)	1	4	2	Hours	1	3	4	0	0	Y	Y

   From this screenshot you can see that the beamer is to be loaned by hours (4 hours) and max. 1 per professor. It can be reserved and loaned even if the professor has e.g. some books overdue.

6.3.6.3. Performing loans transactions with AVM

Finally we need to test our AVM loans system : so we will put a beamer on loan and later on return it.

The following screenshots illustrate 1) the issuing of the loan and the resulting screen, 2) the transaction record and 3) returning the beamer but late, so with fine indication.



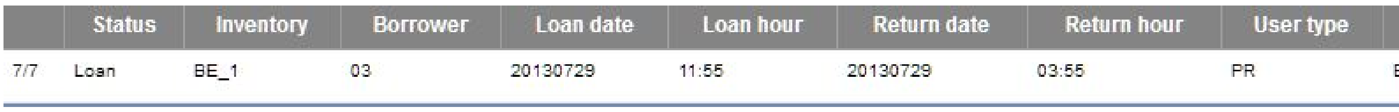
Databases	AVM circulation
Accession number	BE_1
Borrower number	03

Please supply the required data. To continue, click on [Lend] or press [Enter]

Azuaje, Rafael
User type: pr. Professors
User code:03
ID card:667788999
Location:Administración
E-mail:
Telephone:

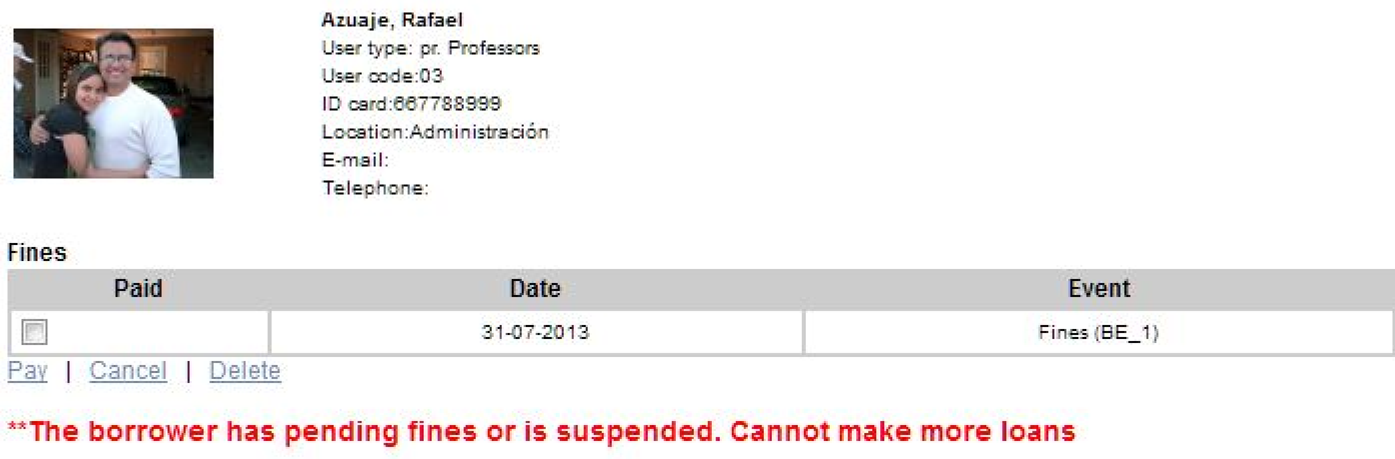
	Accession number	Control number	Reference		Item type
1. ☐	BE_1	000003 (avm)	ID: 000003 Name: Panasonic B333 Cat. :BE	Loan policy: BE - PR	

This transaction can be viewn from the 'databases' option in the Central Loans menu :



	Status	Inventory	Borrower	Loan date	Loan hour	Return date	Return hour	User type
7/7	Loan	BE_1	03	20130729	11:55	20130729	03:55	PR

Suppose the professor returns the beamer only the next day (which is too late) the resulting screen of the 'return' operation will look like the last screenshot :



Azuaje, Rafael
User type: pr. Professors
User code:03
ID card:667788999
Location:Administración
E-mail:
Telephone:

Fines

Paid	Date	Event
☐	31-07-2013	Fines (BE_1)

**The borrower has pending fines or is suspended. Cannot make more loans

After having the fine paid (clicking on the 'tickbox' for the 'Paid' column), the transaction is closed and a new transaction record of type 'return' will have been created in the transactions database.

6.4. The advanced loans module

The advanced loans module of ABCD is an add-on module which can be installed as a separate member of the 'ABCD Suite'. It requires additional software technology (e.g. JAVA, MySQL) to be installed and can also be run as an independent software. Since this module was originally programmed as a DOS-software named 'Emp' (Portugese for 'loans'= emprestímos', the module is called 'EmpWeb' as for ABCD it has been converted into a Web-software, using new web-technology like web-services. So we consider this 'E'mpweb as an 'E'xtra or 'E'nhanced module', maybe changing 'ABCD' into 'ABCDE' ?

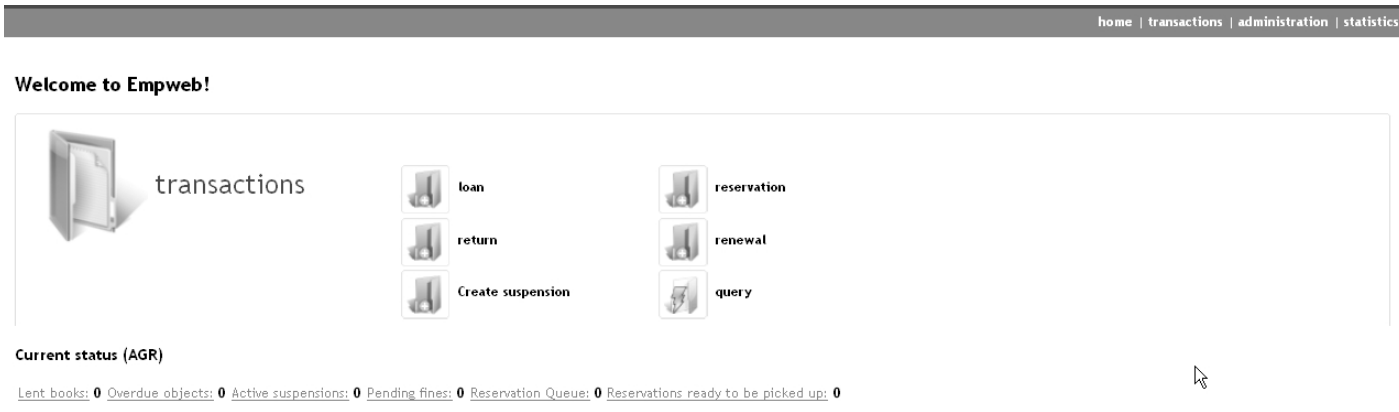
Extra functionalities as compared to the integrated loans module are :

- better capability to deal with complex organisational structures (multi-branch libraries with different loans policies, different servers e.g.)
- more robust handling of high-volume transactions situations
- more interaction with users form the OPAC module, e.g. the 'MySite' function to allow logged-in users to keep track of their own status etc from the OPAC.
- a 'MySite' function allows registered end-users (after logging in) to enter their own space in the loans system to check on their status as a library user and other interactive users. This function at this time is not yet available in the Central Loans module.

Some important concepts of EmpWeb are briefly presented here :

- web-services : instead of needing full access to external resources (databases), which can in some case create problems with the data-providers, web-based 'requests' are sent to the server to just deliver - as a response to the request - some specific data.
- pipe-lines : any transaction (like a loan, a reservation, a return...) goes through a pipe-line of conditions which can be defined. Only if all conditions are met throughout the pipe-line, the transaction will be 'granted', if not it just stops and returns the defined (by the software) error message or instruction (e.g. 'User has been suspended'). This allows any number of conditions and rules to be applied on any decision taken by the software.

EmpWeb therefore can run on any set of external data for which drivers are available or can be accessed by webservices and apply any set of rules onto these data and perform processes (like changing a record) in case of having succeeded to pass all rules and conditions. [!!] EmpWeb in this way is more of a generic transactional engine but used as a loans system in ABCD.



Welcome to Empweb!

Current status (AGR)

Lent books: 0 Overdue objects: 0 Active suspensions: 0 Pending fines: 0 Reservation Queue: 0 Reservations ready to be picked up: 0

[!!] This advanced loans module is discussed in detail in a separate Manual.

7. Central module : thesaurus management

In this section we discuss a specific database structure : a thesaurus, with the typical hierarchical setup of terms with their relations to 'broader', 'narrower' terms, synonyms, scope notes, 'use for' and 'used for' fields etc.

7.1. Explanation of the MTM4 model

7.2. Creation and linking of terms

7.3. Linking the thesaurus to the databases for browsing and indexing

8. ABCD OPAC [THIS DOCUMENT WILL BE A SEPARATE PDF]

8.1. Concepts and files

8.2. the Site Editor

8.2.1. Philosophy of Components

8.2.2. Content managment

8.2.3. management of the Site

8.3. the Search Interface (iAH)

T

8.3.1. Configuration

8.3.2. Indexes

8.3.3. Help messages

8.3.4. Display formats

8.3.5. Plug-ins

9. ABCD Site [THIS DOCUMENT IS AVAILABLE AS A SEPARATE PDF]

10. ABCD Serials Control [THIS DOCUMENT IS AVAILABLE AS A SEPARATE PDF]

10.1. ISSN Standard

10.2. Concept of Kardex

10.3. Creation and edition of serial titles

10.4. Data entry issues

10.5. Configuration and templates

10.6. Union catalogues

10.7. Utilities: export/import, statistics, etc

11. Added Services [THIS DOCUMENT WILL BE A SEPARATE PDF]

11.1. bar code printing

11.2. Selective dissemination of Information - SDI

11.3. Online References

11.4. Photocopies requests

Chapter 3. ABCD Administration and Maintenance [THIS DOCUMENT WILL BE A SEPARATE PDF]

1. ABCD interface configuration [EdS]

1.1. ABCD logical sequences of PHP scripts

1.1.1. DB Manager : index.php, inicio.php, homepage.php

1.1.2. Site and OPAC

1.1.3. SeCS

1.1.4. Circulation and Statistics

1.2. config.php

1.3. logo’s and responsibility statements (footer.php, homepage.php…)

1.4. styles - .css

1.5. Directory of tree view and administration of files

1.6. Utilities: lock/unlock, re-initialization

1.7. Global changes

1.8. Translation utility

1.9. Creation of customized online helps and manuals

2. Toolkit [ES]

2.1. CISIS in depth

2.2. Advanced PFT

3. Administration of the server and maintenance [ES]

3.1. Examples of shells or .bat for automatic maintenance

3.2. Backup, reindexing, compression of MST, automatic mails, checks, and other processes to run offline during night hours using CISIS

3.3. Handling big files