Chapter2
The Arrange data model

This section describes the Arrange document data model, as viewed by a plug-in module. The data model specifies what types of information are contained in a document and how they relate to one another.

- Section 2.1,“Document building blocks,”describes the basic parts of an Arrange document and how they work together.
- Section 2.2,“Primitive data types,” describes the primary elements that plug-ins manipulate.
- Section 2.3,“Selections,” describes how Arrange keeps track of what objects are currently selected by the user.
- Section 2.4,“Current document,” explains how Arrange deals with multiple documents and switching between them.

2.1 Document building blocks

This section describes what a plug-in sees: the basic building blocks of an Arrange document.

A note is an object stored in an Arrange document. It has an arNoteID which is unique in that document, a type, a set of field instances, and a set of special attributes. A special attribute is any information stored in a note which is not its note type or a field instance. Once special attribute is visible to plugin modules: the user reference count, which indicates the number of places in which the note appears (this is the number displayed in the Get Info dialog).

A field instance consists of an arFieldID and the field contents. The nature of the field contents (styled text, date/time information, a picture, etc.) depends on the field type, which is determined by the field definition pointed to by the arFieldID.

Each note also has a visible field list, a special attribute listing the arFieldIDs for hose field instances which are visible in the note. Any field instance whose arFieldID does not appear in the visible field list, is a hidden field. By strict convention, the last entry in any note’s visible field list is the “subnotes” field. This is a note-link field, defined in all Arrange documents, in which a note’s subnotes are stored. The subnotes field is considered to be a visible field, because its contents are visible to the user (subject to folding and unfolding), even though the user does not think of it as being a “field”.

An Arrange document consists entirely of notes (along with a small amount of global information, such as the Preferences settings). Most notes are explicitly created by the user. Other notes, called system objects, are used internally by Arrange to represent entities such as type definitions and topics. System objects are treated just like notes internally, but they do not appear as notes to the user. Depending on their specific type, system objects may have additional special attributes in addition to the user reference count which appears in all notes.

Four types of system object are visible to plugins: field definitions, type definitions, topics, and windows. Folders, topics, and views are all represented by "topic" notes. A window note represents a document window, and stores settings such as the window position and the currently visible topic in that window.

2.2 Primitive data types

Plug-ins communicate with the data model using primitive data types. This section lists the primitive data types used in the Arrange plug-in module API.

Most of these types begin with the prefix "ar" (for Arrange), to avoid conflicts with type or function names used in the standard Mac interface files or in client source code.

arDocumentPtr	identifies a specific open document. An arDocumentPtr is assigned to a document when it is created or opened; it remains valid for as long as the document remains open.
arNoteID	identifies a note within the context of a document. See also section 2.2.1, below.
arFieldID	identifies a field definition. All arFieldIDs are also valid arNoteIDs.
arTypeID	identifies a type definition. All arTypeIDs are also valid arNoteIDs.
arTopicID	identifies a topic, folder or view. All arTopicIDs are also valid arNoteIDs.
arWindowID	identifies a document window. All arWindowIDs are also valid arNoteIDs.
arDate	a specific point in time, in the standard Macintosh date/time format (number of seconds since 1/1/1904).
arFloat	a standard 64-bit floating-point number (used in Number fields).
arListID	a "handle" to an array of arNoteIDs. Callback functions are provided to access the array. Note that an arListID is not a valid arNoteID. It

should be treated as a pointer to an in-memory data structure (although the actual implementation is more complex): it must be explicitly disposed of (unlike an arNoteID), and becomes invalid when the application terminates. Furthermore, an arListID is only valid within the context of a particular document, and becomes invalid when the document is closed.

arPathID	a "handle" to an object which represents a "path" into the document. A path points to either an note or a field instance, and specifies the location of that note or field instance by listing the chain of parent notes and fields all the way up to the topic in which the uppermost parent note appears. (A path can also be anchored in a shelf rather than a topic.) Like an arListID, an arPathID must be explicitly disposed of, is only valid within the context of a particular document, and becomes invalid when the document is closed. The major use of arPathIDs is to describe the location of the selected note, field, or text.
arGlobalID	a 96-bit identifier which is unique across all documents. arGlobalIDs are used to uniquely identify a note even if it is moved from one document to another; for example, Arrange's File Merge command uses arGlobalIDs to match up notes in the two documents being merged. Most manipulation of notes is done using the more compact arNoteID.

2.2.1 arNoteID

Each note in an Arrange document has a four-byte identifier called an arNoteID. (This applies only to documents which are open in memory, or saved in native format. Notes in a TAIL file do not have arNoteIDs; arNoteIDs are assigned when the TAIL file is imported.) The arNoteID is unique among all notes in a given document, but is not unique between documents. A note's arNoteID is assigned when the note is first created (or imported from a TAIL file), and remains fixed for the lifetime of the note in its document. When a note is transferred from one document to another (via copy/paste or dragging), a new arNoteID is assigned in the target document. The arNoteID is the standard way of referring to a note in a function call, and is how notes refer to one another in Arrange's internal data structures.

When an arNoteID is known to refer to a specific type of system object, it may be referred to as an arFieldID, arTypeID, arTopicID, or arWindowID, for field definitions, type definitions, topics, and windows, respectively.

2.3 Selections

In the Arrange user interface model, each window has its own selection. Commands apply to the selection in the active window. The selection can be in one of five states: a range of selected text (we consider an insertion point to be a text selection of zero extent); a non-textual selection in a field (e.g. a picture or file reference); one or more selected notes; one or more selected fields; or no selection. The basic selection types (text, other field contents, notes, and fields) cannot be mixed; thus it is not possible to simultaneously select a note and a field, or to simultaneously select two pictures. Also note that selecting all of the text in a field is different than selecting the field itself.

When a note is selected, we record not just the arNoteID of the selected note, but also the arNoteID of the parent note and field instance in which the note appears. This is represented as the triple (parent, field, child), called a selection triple. Thus, if a note A is a subnote of notes B and C, it can be selected inside note B, or inside note C, or both simultaneously; in the latter case, the selection would have two distinct entries, the triples (B, subnote, A) and (C, subnote, A).

Similarly, when a field instance is selected, we record the ID of the field and the ID of the note in which the instance appears. This is represented as a triple (parent note, nil, field). In general, a selection of notes or of field instances is represented as an array of these triples.

2.4 Current document

When a plug-in calls a field, it must have the entire context for that field. The current document mechanism keeps track of an implicit current document. This section describes the current document mechanism.

All Arrange operations are performed within the context of a current document. For example, arNoteIDs are only unique within a particular document; the same ID may be used for two different notes in two different documents. The current document is an implicit parameter to almost all hook and callback functions (just as the current GrafPtr is an implicit parameter to almost all QuickDraw functions). Callback functions are available to get and set the current document.

Whenever Arrange calls a plug-in module, the current document will be the document associated with the active window, unless explicitly noted otherwise in the documentation for the specific hook function being called. (When a module's root entry point is called with the initialization or exit messages, there are no open documents, so the current document will be nil.) All callback functions will preserve the current document. Plugin modules must be careful to set up the appropriate current document before calling a callback function, except certain document manipulation calls such as New Doc and Set Current Doc. In most cases, this is not an issue, because the plug-in will be working only with the active document, but if the plug-in module explicitly acts on more than one document at a time, or if it maintains internal state which applies to a particular document, then it must set up the current document properly.