Recipe: Integrating C&S into a group application

by Cathy Duffy and Ryan Jansen

[Editor's note: This article resides in "Notes Today", the technical Webzine located on the http://www.notes.net Web site produced by Iris Associates, the developers of Domino/Notes. Previously, Notes.net introduced readers to Calendar and Scheduling in two articles. In the first article, ***Creating a Calendar View in Lotus Notes****, Catherine Duffy of Iris Associates explained how to incorporate a calendar view into an application. The second article,* ***Using Time/Date Values in LotusScript*** *by fellow developer Ryan Jansen described the use of the time/date controls. Both of these articles are worth rereading before you tackle an application that integrates calendar and scheduling with the mail file. Cathy and Ryan have teamed up to write this article describing how to integrate C&S into your group applications. They've categorized this task into the application types described below. You can use these models to jump-start to your application.]*

Overview

Notes Release 4.5 contains a collection of calendar and scheduling (C&S) features that are available to users direcly from within the Notes environment. In addition to the capability to view information in calendar-style formats, Notes users can use the free-time lookup feature to schedule meetings. Instead of the usual round of memos and phone calls to find a suitable time for all participants, free-time lookup tells you who is busy when and helps you finds a time when everyone is free. When the invitation is distributed, auto-processing of notifications and acknowledgment from recipients accelerates the process of finalizing the meeting.

Once you have familiarized yourself with the C&S features, you may want to use them to develop or enhance your own group applications. While creating a Calendar view and using the Time/Date controls is fairly easy, integrating a C&S application with the Notes free time system and /or alarms requires integration with the mail file, and is not as straightforward. Not unexpectedly, application developers have asked for advice on this subject.

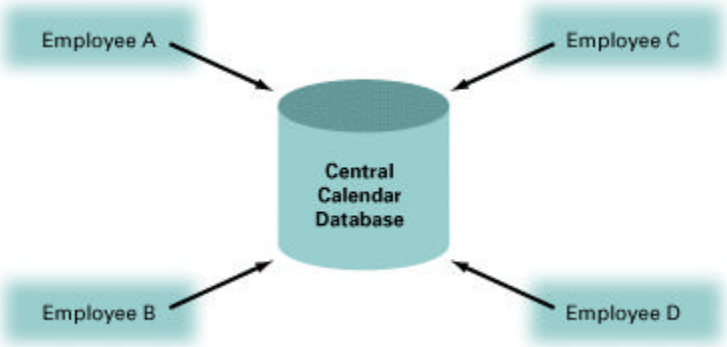
In this article, we will discuss three types of C&S applications:

1. Centralized group scheduling -- used when an organization wants to store everyone's appointments in one central and public location.
2. Event scheduling -- used in organizing an event, planning for and reserving the required components.
3. Ad hoc applications -- used for any other type of calendar application that interacts with C&S in mail.

Centralized Group Scheduling

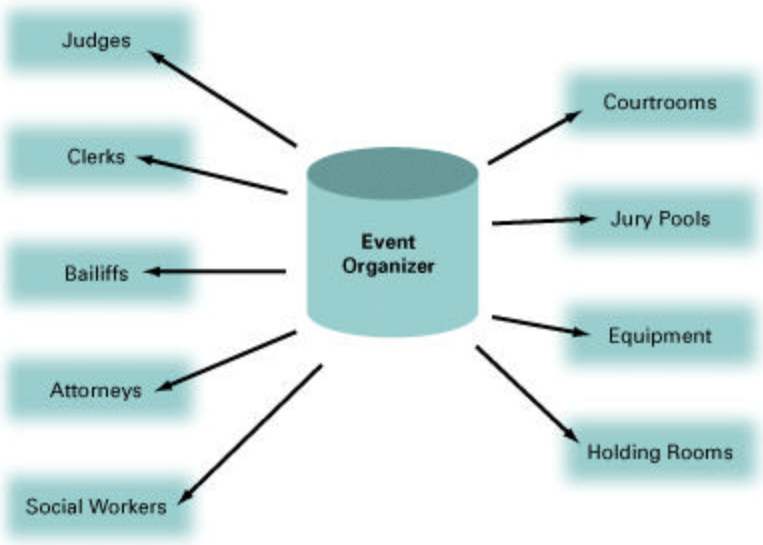
In the centralized group scheduling application, everyone's appointments are stored centrally making anyone's whereabouts known at a glance. Individuals schedule their own time on one group calendar.

An example is a business where group members schedule appointments on their own but want to see if the people that are needed for a group activity are busy before the activity is scheduled.



Event Scheduling

The event scheduler application is used to organize events with multiple participants and resources. The organizer invites people and reserves rooms from lists of available participants and resources. An excellent example of an event scheduler grew from a conversation at Lotusphere. A judicial administration was involved in scheduling trials or hearings at a courthouse. They needed to reserve a court room, a judge, attorneys, social workers, stenographers, and so on. Ideally the user would be able to choose from lists of available resources in each of these categories or allow the application to choose for her (similar to the way you choose a room on the invitation form or let one be chosen for you).



Ad Hoc Applications

The third type of application is any other type of calendar and scheduling application that interacts with mail. For this type, the developer needs to know what forms and fields are required and what the field values are. We are providing a full list of fields in the section "Ad Hoc Application Fields."

In the centralized database and the event scheduler types of applications, the developer could do all of the scheduling from the application database, doing the free time logic themselves and ignoring the mail file. But an individual might be involved in more than one group or project and might be scheduled in several scheduling applications. Furthermore, that would not take into consideration appointments scheduled using the mail file.

Therefore, our challenge is to find a way to integrate these application types with calendar and scheduling in the mail file. The centralized scheduling and the event scheduling application types are discussed in detail. As a reminder, our suggestions throughout this article are not meant as the only way to accomplish this and they are not complete specifications. Our suggestions are only meant as an example of what we feel might be the simplest approach to the problem.

Implementing the Centralized Schedule Database

In this application, all members schedule their appointments in a centralized database.

Premise:

The application works with the following assumptions:

- All calendar documents are created in the mail file and put in the central calendar file.
- Since mail is always on the Create menu, the user selects Create - Mail - Calendar Entry from anywhere in Notes, enters the appropriate information, and saves and closes the document.

- The document is deposited in the user's mail file, the invitees' mail files (if it is an invitation), and the central calendar database.

Considerations:
For the application to work, you must modify the mail template and create a central calendar database. Then you have two implementation decisions to make.

First, decide what types of Calendar Entry documents are stored in the central calendar file and which of these are personal. You could use the "Not for Public Viewing" checkbox to allow users to indicate that a document is private. That is, if OrgConfidential = 1, you will not execute the logic that puts this document in the central calendar. You could also add an action or checkbox for this purpose.

Second, decide how to "put" the calendar entry into the central database. You can choose between *direct write* to the central database and *e-mail* option, either of which are executed from the QueryClose event of both the Calendar Entry and Notice forms, after the logic determines that the document should be closed.

Direct write vs. e-mail:
The first option is direct write to the central database. In Query Close, copy the current Calendar Entry to the central database. This option works well for a new document or for the initial acceptance of invitations, but for updates, you have another decision to make from either of the following choices:

- You could choose to remove the copy of the original document from the central database and copy a new version from the mail file to the central database. In order to do this, the document in the central database must be read only. Otherwise, if someone changed the central copy of the document, this update would overwrite it.

- Or, you could apply the changes to the document in the central database. However, if someone edited the body of the central database's copy, you cannot easily determine what the change was and would not know what changes to apply. Therefore, you should disallow edits in the central database and require that any changes are made by the originator in the mail file. (Then they could either remove and copy as suggested above or replace item values.)

Two important things to consider when choosing the direct write option are the server of the central calendar database might not be accessible at the moment and it might be slow. Because of these, you might find this method unreliable.

The second option is e-mail to the central database. In this case QueryClose sends the document to the central calendar database. The central database would have to be set up such that it does not display these documents until they have been processed by a mail-in agent. In this agent the new document is marked as processed, perhaps by setting a field value that is in the selection formula of the view. Then changes would be posted by either replacing or updating the original document.

There are also two important considerations when choosing the e-mail option. First, there is a lag between the time of the update and when the entry is posted to the central database calendar. Second, several updates could be submitted before the agent picks up the first update, so the agent must be able to decide how to apply the changes. The agent can use any of the following methods:

- Apply all changes in the order they were made, by checking the modified date/time
- Apply all changes in the order they were sent, by checking the PostedDate
- Apply only the last change, identified by checking the modified date/time or the PostedDate
- Apply only the changes identified by a sequence number scheme
- Although the e-mail option requires more work, it may be more reliable in the long run.

Implementation:
Once you have decided on these considerations, you need to do the following work:

- Currently, the ApptUNID field, which contains the UNID of the original Calendar Entry document, is only posted for Invitations. You will need to create this field for all Calendar Entry types and then use it as the key to retrieve the correct document from the central calendar database in the event of a modification or deletion. To identify the originator's copy of an entry in the central calendar database, use the ApptUNID field combined with the Chair field. To identify an invitee's copy, use the ApptUNID combined with the InviteeName field.
- Add a routine to the QueryClose events in both the Calendar Entry and Notice forms in the mail template that does the copy, update or e-mail.
- Add a field that indicates if this entry should be displayed in the central database (if you are not using the OrgConfidential field as described above). Then use the value of this field in QueryClose to determine whether the above mentioned routine is executed.
- Create a central database which includes a Calendar view and a Calendar Entry form. These can be copied from the mail file then modified as appropriate.
- Remove all processing capabilities from the forms in this database, forcing modifications to be initiated in the originator's mail file. If you do not, you introduce another level of complexity because the put routine will need to know whether to put into the central calendar database or put into the user's mail database.
- Modify the "New Entry" action and the RegionDoubleClick event such that they create the document in the user's mail file (using **@MailDbName** with **@Command([Compose])** in the action and using **session.GetEnvironmentString** to get the MailServer and MailFile values and then using those in **uiworkspace.ComposeDocument** in the RegionDoubleClick event).
- If you are using the e-mail option for updates, access to this database should be reader. Otherwise, it should be author, but the Calendar Entry form should be modified such that it does not appear on the Create menu and it is not editable (either make all fields computed, or add logic to both QueryOpen and QueryModeChange that disallow edit mode).
- If you are using the e-mail option, create a mail-in agent in the central calendar database that processes new mail as described above (posting changes and marking the correct documents as processed so they will appear in the view). Of course, this database also needs a mail-in database document in the Public Address book.

The bottom line is that nothing originates in the central database. Documents are deposited into it but nothing leaves it.

Implementing an Event Scheduling Application

The event scheduler application is used to plan events to which people are invited and for which resources or rooms must be reserved. In the following example, one person at a court house schedules events -- trials, hearings, arraignments -- that require types of participants, rather than specific people. The organizer also needs to reserve rooms -- courtrooms and holding rooms -- and resources, such as audio-visual equipment or transportation.

Premise:
An event scheduling application is based on the following assumptions:

- All documents are created from a central scheduling database.
- The database does not represent any person, place or thing, except the role of the organizer.
- The database is a place for initiation and storage. It is a point of departure.
- The database uses the free time system to check on the availability of people, places, and things and invites them to an event.
- The invitees receive invitation notices in their mail files, which they accept, decline, or delegate using the existing calendar and scheduling workflow.

Considerations:

To create this type of calendar application, you can start with the mail template and remove everything except the Calendar View, the Calendar Entry, Notice and Calendar Profile forms, and the following subforms: ChangeRepeating, InviteeResponses, NamesPrompt, NoticeOptionsDlg, PeopleListDlg, RepeatForm, and RepeatInfo.

You might also want to modify the CalendarHelp form to revise the help as appropriate for your application.

Implementation:
You would have to do the following work to create this application.

- Change the face of the Calendar Entry form so it displays the correct information. For example, change Calendar Entry to "Schedule a Hearing." Do not change the field names unless you want to make major modifications to the LotusScript. Simply change the labels and add any new required fields.
- Change the titles of the actions and some of the words in the message boxes and notice subject lines to reflect what the application is doing. To do this, search through the Form Events and ScriptLibraries for any messagebox or print statements as well as for the word "subject". Below is a brief description of what each of the ScriptLibraries does:
- *AppointmentProcessing* handles actions that the chair might take (invite, reschedule, cancel, confirm, remove invitee/resource) and contains most of the text the chairperson sees when they take an action. This is what the person scheduling the event will see.

    AppointmentResponses handles generating the list of invitee responses and determines who should receive update notices. This also handles Accepting/Declining a counter proposal.

    NoticeProcessing handles actions that an invitee might take (accept, decline, etc.), so this is probably not relevant to this set of modifications, as people will be accepting and declining from mail.

    NoticeResponses searches for notice updates from the chair for a particular meeting.

    ResourceProcessing handles all activities for inviting, rescheduling, canceling, or removing a room/resource.

    SharedWorkflowLibrary contains common routines used in processing notices, either from the Chair's point of view or the Invitee's.
- Set up a mail-in database document for this database in the Public Address Book and give it a relevant name (for example Court House Scheduling).
- Delete everything on the CalendarProfile form below the Default appointment/meeting duration field.
- Change the Owner field to computed. Its value must be the same as the mail-in database name of the database or the application will not work.
- Assuming you are allowing the users to pick from lists of available people in a specific set of roles, there are several methods you can use to generate lists. The following methods are two possibilities:

    Generate lists for each role. You can use the **session.FreeTimeSearch** method to generate the lists of names of available people in each role. This method returns time ranges that are free for these people within the range that you pass to it, so you might need to check each person against the exact time slot, which could be slow. You might want to have an hourly agent generate availability lists, then use the method described below to double check, or to resolve late breaking conflicts.

    *Include a control button to check free time***.** It would probably be faster to have a button that checks the free time for a list for each role. For example, you would have a button called "Choose an Attorney" and another called "Choose a Judge", and a field called AttorneyList and another called JudgeList. The first button would do **@Command([FindFreeTimeDialog]; AttorneyList; ""; ""; ""; ""; ""; ""; StartDateTime; EndDateTime)**. This would allow the user to see who in that list is free during the time slot. They can click OK, then select the correct person from the list, which would add them the SendTo

field. The same process could be repeated for the other roles of people. The names of all the people you are inviting need to be put in the SendTo field or the invitation will not be sent.

- Handle categories of places and things in the same way rooms and resources are handled. The reservation file is discussed in detail in the section "Rooms and Resources" later in this article.
- You might want to write an agent that checks on the status of responses to each of these events. If all invitees accepted, change the value of _ViewIcon to something, if someone declined, change it to something else, and if not everyone responded, change it to a third thing. It might also send reminder notices to those who did not yet respond or create a report.

Once this work is done, the central scheduling application essentially acts as a chairperson "inviting" the correct elements to the event, and it should be able to use the existing workflow logic to keep everything synchronized.

Implementing Ad Hoc Applications

If you are creating any other application which needs to send Calendar Entries or Notices to your mail file, you will need to have a number of fields populated in a particular manner. The following sections contain tables with complete lists of the available fields in the Notes calendar and scheduling component, their value and type, what they do, and whether they are required. Use whatever fields are relevant to your application.

Appointments:
Table 1 shows the fields for the documents that appear on the Calendar view.

Table 1: Fields that appear on documents in the Calendar view

Field Name	Type	Values	Comments	Required?
$BusyName	text	fully distinguished username	Person that is busy for that timeslot. You could get it from @Username, but because of delegation we use the Owner field in the Calendar Profile	Only if you want the free time system to show this person as busy for this calendar entry
$NoPurge	date/time	EndDateTime	Prevents the note from being purged by replication before the appointment has occurred	Recommended
$REF	text	UNID of the parent	This field exists on repeat instances and subsequent notice documents. (For repeat instances, the parent holds the rules, RepeatDates, and RepeatIds of each instance in the repeat set.)	Only if the doc is supposed to be a response
$RefOptions	text	"1"	Ordinarily if you save a response doc with a form that is designated as type Document, the $Ref is deleted and the doc will no longer be a response. This tells Notes that although the form which this doc uses (Appointment) is of type Document, continue to treat it as a response.	Only if the doc is supposed to act as a response although its document type is "Document" (in our case, it is required only for repeat instances)
$BusyPriority	text	"1" = Busy "2" = Not Busy	Tells the scheduler whether this appointment is considered busy time or free time	If there is a $BusyName but no $BusyPriority the free time system still sees this time as busy
AppointmentType	text	"0" = Personal Appointment "1" = Anniversary "2" = Event "3" = Meeting "4" = Reminder	Controls HideWhens and other processing	Yes
Body	richtext			
BookFreeTime	text	"1" = Time is free "0" or "" = Time is	Default is "0". If set to "1", the $BusyPriority	

		booked	field is set to "2" otherwise the $BusyPriority is set to "1". Tells the scheduler if you are busy or free during that time.	
CalendarDateTime	date/time	StartDateTime	Causes something to show up in the Calendar view.	Only if you want the document to appear in the Calendar View of the mail file
Chair	text	fully distinguished name of the Owner of the mail file that originated the Calendar Entry	Helps to identify who originated this doc	Yes
CopyTo	textlist		Used for sending notices to OptionalAttendees	
DocAuthors	authornames	fully distinguished username	The person who created the document	
Duration	number		Number of days an event spans	Only if AppointmentType is event
EndDateTime	date/time		The end date & time of the meeting. Created from combining StartDate and TimeRange	For all AppointmentTypes except reminder

ExcludeFromView	text	"D"	Prevents appointments that are not sent from showing up in the drafts view	Unless you want the appointment to appear in the Drafts view
Form	text	"Appointment"	Determines what form to display the doc with	Yes
From	text	fully distinguished username	The person who created or who sent it. May include @domain	Yes
FromDomain	text		If sent from a different domain, the domain route will be in this field (posted by router)	
NoticeType	text	"I" = Invitation (original) "U" = Update (originator reschedules) "C" = Cancel (by originator) "D" = Delegator notifying originator "S" = StatusUpdate (by originator) "N" = Confirmation (by originator) "J" = Originator declining counter "A" = Accept (by invitee) "R" = Decline (by invitee) "L" = Delegator notifying delegate "T" = Counter Proposal (by invitee) "P" = Pencilin (by invitee)	Notices sometimes get converted to Calendar Entries. This is the value it had while it was a notice.	
OptionalAttendees	textlist		Full names of optional invitees. These are the names as "corrected" (expanded?) by the mailer.	
OrgConfidential	text	"" = False "1" = True	If True the $PublicAccess field is sent to "", otherwise the $PublicAccess item is set to "1"	
OrgDontDoubleBook	text	"" = False "1" = True	Causes conflict checking to occur. This item gets sets only if the appropriate setting in the profile	Only if you want to check for conflicts

			document for that appointment type is set	
OrgRepeat	text	"1" = this is a repeating appointment	If it is not a repeating appointment, this field will not be present. See below for more on repeating appointments.	
OrgTable	text	"C0" = Calendar (set for appointment) "T0" = ToDo "H0" = Calls "P0" = Planner (set for event) "D0" = Address "N0" = Notepad "A0" = Anniversary (set for anniversary)	Only "C0" is used	Yes
PostedDate	date/time		Mailer creates this when you send it	
Principal	text	fully distinguished username	Owner of mail db in which it was created	Yes
Recipients	textlist		Who mail got sent to - posted by router	
RequiredAttendees	textlist		Full names of required invitees. These are the names as "corrected" (expanded?) by the mailer.	Only on invitations
RequiredResources	textlist		List of resources that have been invited to a meeting	
ReserveRoom	text	"1" = Reserve room from Reservation's db "" = no room needed	Indicates if a room needs to be invited	
Room	text		Name of room to reserve	Only if ReserveRoom = "1"
SendTo	textlist		Who you are going to send notices to	Only if you are going to send this document
SequenceNum	number		Set to 1 on an original invite. Incremented on any reschedule. (see below for discussion)	Yes
StartDate	date		The date of the appointment, used in data entry.	Yes
StartDateTime	date/time		The start date/time of the appointment. We generate it by	Yes

			combining StartDate with either ReminderTime or TimeRange	
Subject	text		Appointment description. This is what shows in the view (more lengthy text goes in the Body field)	Recommended
TimeRange	date/time range		StartTime - EndTime. Used for data entry.	
_ViewIcon	number		Indicates which view icon to use	Only if you want a view icon to display
Uninvited	text list		People who were uninvited or were removed via the Freetime dialog	

Notices:
Table 2 shows the fields for notices. In general, a notice is sent to an invitee, and based upon the invitee's action, will be converted to an appointment or will generate another notice. Some of these fields are identical to those in the appointment.

Table 2: Fields used for Notices

Field Name	Type	Values	Comments	Required?
$Ref	text	unid of the parent	Since notices are responses, this is necessary to maintain the hierarchy	Yes
Chair	text	see above		Yes
Delegee	text		Same of person the delegator delegated to	
DeliveredDate	date/time		Posted by mail	
EndDateTime	date/time	see above		Yes

Form	text	"Notice"		Yes
From	text	see above		Yes
InviteeName	text	name	Tells the Chairperson who (which invitee, not who sent the mail) the notice response is from.	Only on notices sent to the chair.
NoticeType	text	see above		Yes
OrgState	text	"5" = room "6" = resource	Determines if a notice is being sent to/from a room or resource. This is used by the Reservation database.	Only on notices involving resources
OrgStatus	text	"2" = Accepted "5" = Removed	Sent on a Status Update notice to an invitee telling them they are either required to attend or removed from a meeting invitation	
ApptUNID	text	UNID of the appointment being referenced	Used by the Reservation database to 1) Create replies back as a response document, 2.) Find resources by a particular appointment UNID. Used to maintain the response hierarchy in case an UNID got overwritten	Only on notices being sent from the chair
PostedDate	date/time	see above		
Principal	text	see above		Yes
Room	text	see above		
SendTo	text	see above		Only if the notice is being sent
SequenceNum	number		this matches the SequenceNum on the invitation or reschedule notice to which you are responding	Yes
StartDateTime	date/time	see above		Yes
Subject	text	see above		Recommended
Topic	text	Original subject from the appointment	Subject ends up on the sent message to show in the view and Topic is displayed on the document	Yes
_ViewIcon	number	see above		see above

Sequence Numbers:
The SequenceNum field is used to make sure that meeting notices are not processed out of order. Simple examples of these are:

- I accept a meeting and then later decline - The chair should see that I declined.

- I receive an invitation and then a cancellation notice from the chair- I should be aware of the cancellation and should not be allowed to respond to the original invitation.

The SequenceNum field is found on all Meeting Appointments, Meeting Invitations, Meeting Updates (i.e. Reschedule, Cancel, Uninvite,etc.), and Meeting Replies (i.e. Accept, Decline, Delegate, Propose). When the Chair first creates a Meeting Appointment, this field is set to 1 and included in all invitations. The SequenceNum field only changes when the meeting has been Rescheduled. Rescheduling increments the SequenceNum field by 1 and sends out a Reschedule notice that contains the new SequenceNum.

Repeating Appointments:
The RepeatProcessing ScriptLibrary handles a lot of the following logic:

- When a user selects the Repeat action (on the Appointment form) a Repeat Options dialog is presented.
- Once the options are selected and the user clicks OK on the dialog, a field called OrgRepeat is written to the Appointment, indicating that this is a repeating appointment.
- When the user saves this appointment, we determine what the repeat dates should be (based on the options selected), and then create an Appointment document for each date. These repeat instance documents are all stored as responses to the original document.
- As the repeat instances are being created, the UNID of each is stored on the parent so that we can create an identical set in each invitee's mail.
- When inviting people to a repeating appointment, the original (parent) document is the one that is sent.
- When the invitee accepts the invitation, we create the repeat instance response documents (as above) and assign the appropriate UNID to them.
- The original (parent) document carries all the repeat parameters. The instances only have the OrgRepeat field. If that field is present and a repeat instance needs to be changed, it looks to the parent to see which other instances should also be changed. Once the instances are generated, only RepeatDates and RepeatIds are actually used.
- The repeat instances show up on the Calendar. The parent does not.

Table 3 shows the fields we used in the repeat logic.

Table 3: Fields used in repeat logic

Field	Value	Comments
OrgRepeat	"1" if this is a repeating appointment	Indicates that this is a repeating appointment. If it is not present then it is not a repeating appointment.
RepeatAdjust	numeric string	If RepeatUnit = "W" this represents day of the week, where Sunday = "0", Monday = "1", etc. If RepeatUnit = "MD" this represents a day of the month where the 5th of the month = "5" If RepeatUnit = "MP" this represents the week in the month and the day of the week where the first Monday = "1.1", the 2nd Sunday = "2.0
RepeatDates	array of dates (date/time list)	The dates of the repeat instances
RepeatFor	number	number of days, weeks, months, or years to repeat for (used in conjunction with RepeatForUnit)
RepeatForUnit	string	the unit used in RepeatFor (D, W, M, Y)
RepeatHow	"U" = Until "F" = For	If this = "F" then we use RepeatFor and RepeatForUnit to determine how long to repeat
RepeatIds	array (text list)	UNIDs of all of the repeat instances
RepeatInterval	numeric string	"3" would represent every third RepeatUnit (so if RepeatUnit = "D" then "3" means every third day)
RepeatUnit	"D" = Daily "W" = Weekly "MD" = Monthly By Date "MP" = Monthly by Day "YD" = Yearly "C" = Custom	
RepeatUntil	date/time	If RepeatHow = "U" we repeat until this date
RepeatWeekends	"D" = Don't move "F" = Move to Friday "M" = Move to Monday "N" = Move to nearest weekday "X" = Delete	represents what to do if a repeat date falls on a weekend

Alarms:
If you want to add alarms to a document, you must set the all of the fields shown in Table 4, put the document in the $Alarms folder of the mail file (using the **notesdoc.PutInFolder**), and enable the Alarm daemon (using **@EnableAlarms** or **uiworkspace.EnableAlarms**).

Table 4: Fields required for alarms

Field	Value	Comments
$Alarm	1	Alarm is on
$AlarmDescription	string	Shows up when alarm "rings"
$AlarmOffset	number	negative = x minutes before the StartDateTime positive = x minutes after
$AlarmTime	date/time	The time at which the alarm should go off. This and $AlarmOffset are mutually exclusive.

Rooms and Resources:
The first thing you need to do, is to create the reservation database. To do this, you can use the Resource Reservations (resrc45.ntf) template as a starting point. In order for the reservation database to work, you'll need to create one or more Site Profile documents. A Site Profile is used by resources for grouping and mail-addressing. You can use any granularity you like for the Site Profiles. Some ways you can define your sites are:

- Departmental (accounting, marketing) - useful if each department wants to maintain it's own resources
- Location (U.S., Japan, Europe) - useful if each location will maintain it's own resources

In terms of functionality, it doesn't matter how you break up your Site Profile documents. Keep in mind that all resources belong to only one Site Profile and the name of the site that a resource belongs to becomes part of that resource's name. For example, if you create two site profiles, named Accounting and Marketing, and then create a room "Conference Room A" that belongs to the Accounting site profile. The full name of this room would be "Conference Room A/Accounting."

The second thing you need to do is to add resources to your reservation database. To do this, simply choose Resource from the Create menu. You can create two basic types of resources -- a room and an object. Creating a room sets the ResourceType item to "1", while any type of object sets this item to "2." Rooms and objects differ in that rooms have a capacity while objects belong to a category. However, instead of creating rooms, you want people to be able to reserve limousines, for example. You can treat limousines like rooms because they have a capacity and you wouldn't want to overbook a limo!

The category value for objects allows you to group similar objects together. For example, you might create an object named "21 inch Color Monitor" that belongs to the category "Computer Monitors." Objects and categories can be used in free time searches. If you want to reserve a computer monitor, but you don't have a preference for which one you use, you can simply ask Notes to find you any available resource belonging to the "Computer Monitor" category.

If you recall, resources get their full name by combining the name you enter and appending the site it belongs to. When you create resources that have categories, the category value is also used to create the resources full name. In our color monitor example, assuming this monitor belonged to the Marketing site profile, the full name would be "21 in. Color Monitor/Computer Monitor/Marketing." Notice that this translates into a Notes hierarchical name, where CN=21 in. Color Monitor/OU=Computer Monitor/O=Marketing

This allows users to invite resources to events as well as perform free time searches.

We recommend that the ADMIN process be running on the server. When you create a resource document in the reservation database, the ADMIN process will create a mail-in database record in the public Name & Address Book so that users can invite the resource to events and can also view its free time information. If the ADMIN process is not running on the server, you'll have to manually create the mail-in database records in the public NAB.

Using the Calendar Entry form to "invite" your own resources:
In this application, do not change the underlying functionality of the buttons that appear in the Reservation section on the calendar entry form. If you do change the code, you'll want to make sure to set the following fields appropriately. Once you do that, you should get all of the workflow for free: inviting, rescheduling, canceling, removing.

When you are inviting a room (a resource that has a capacity), three fields are involved.

- *RoomRequired = "1."* This tells Notes that the user has requested to reserve a room and special processing needs to take place.
- *RoomToReserve = name of room*. This tells Notes that a room needs to be reserved. This field is removed after the invitation has been to the room.
- *Room = name of the room.* Currently, only one room can be invited to an event.

When you are inviting an object (a resource that has a category), one field is involved.

- *Resources = the new resources*. Resources the user wants to reserve.

Currently, users can reserve multiple resources, even at different points in the event.

If there are entries in the Resources item, then Notes will send an invitation notice to each entry listed. Each entry is removed from the Resources item and appended to the RequiredResources item. After the invitations have been sent to the resources, the Resources item will be blank. This item is really used as a trigger to know that resources need to be invited.

Once you have completed this work, the central scheduling application acts as the chairperson, inviting the correct elements to the event and the existing workflow logic keeps everything synchronized.

Summary

The suggestions that we have presented here are not complete solutions, but ideas on how to approach development of a C&S applicatsion. Our recommendations for your success with developing these applications are that you determine your own requirements, use whatever implementation suggestions apply to your application, and choose from the list of fields to create more complex ad hoc applications.

ABOUT THE AUTHORS

Cathy joined Iris in May 1995 to develop Notes Release 4 templates (approve4.ntf, discuss4.ntf, doclib4.ntf, doclibm4.ntf, and doclibl4.ntf). She has been working on the mail template since October 1995 and has been developing applications (both Notes and otherwise) for 15 years. Her area of expertise is workflow applications.

Ryan has been with Iris since June of 1996. He was previously employed by Lotus and has also worked for a Lotus Premium Business Partner. At Iris, Ryan is one of the developers who worked on Calendaring & Scheduling in the Notes R4.5 Mail template and will continue working on this feature for future releases. During off hours, he is often found playing Quake™ or Red Alert™ with fellow Iris developers.