date and time, in standard format (`MM/DD/YYYY, HH:MM`)

GpclsValidDate—determines whether a date string's format is valid

2.3 COMPARISON CTR.

The "Comparison Ctr." job compares the contents of a specific customer's active fleet status report against SynchroNet's database. It produces a pair of spreadsheet files, containing matches and errors, which SynchroNet managers and the accounting team can use to reconcile any discrepancies.

2.3.1 INPUT AND OUTPUT FORMAT

Because container status reports are often copies of customers' own internal documents, their content and format varies considerably. For this reason, they are usually processed by the system's EDI interpreter (see page 9), which produces output files in a standard format.

Although many include additional columns or use different labels, every report must contain the same basic information for each container:

- supplier and receiver IDs
- container number (AssetId)
- pickup release number
- pickup date and location
- redelivery date and location (if any)

Reports from a given customer must also be formatted consistently, so that the same information is always presented in the same way. When this is not the case, the data manager may need to inspect the file and make manual adjustments to ensure that it conforms to a format that the interpreter can accept. The output from the "EDI Read" job then becomes the input for the "Comparison Ctr." job, in a form like that in figure 2.3 (page 19).

Cntr Numbers	Size/ Type	Lessee	Pick-up Date	Pick-up Location	Pick-up Ref	Return Date	Return Location
CAXU3367288	20/DC	SYN	12/14/2008	RULEDP01	UJ82502		
CAXU3367816	20/DC	SYN	12/10/2008	DKCPHP01	UJ81932	1/15/2009	NLRTMP05
YMLU2829587	20/DC	SYN	7/29/2008	DKCPHP01	UJ76731	1/7/2009	NLRTMP05
YMLU2829801	20/DC	SYN	2/9/2009	SEHELP01	UJ85265		
YMLU2920887	20/DC	SYN	11/25/2008	DKCPHP01	UJ82085	1/5/2009	NLRTMP05
YMLU3363868	20/DC	SYN	12/9/2008	DKCPHP01	UJ82764	1/5/2009	NLRTMP05
YMLU3365665	20/DC	SYN	11/25/2008	MACASY01	UJ81998		
YMLU8114003	40/HQ	SYN	11/18/2008	ITVNIY01	UJ81750	1/6/2009	NLRTMP05

Figure 2.3: “Comparison Ctr.” input data

Note: The system can also retrieve container data automatically from customer Web pages; in this case it creates the input file in the correct format without any manual intervention.

The output error spreadsheet summarizes the data that were processed, from both the customer and the database, and presents the error messages from the “Comparison Ctr.” job (figure 2.4, page 19).

				Customer Data						
Supplier		Receiver		Container #	Size/Type	Pickup Date	Pickup Location	Pickup Ref. #	Return Date	Return Location
YME	CC0003473	EIM	CC0000637	YMLU2920887	20/DC	25-Nov-2008	DKCPH	UJ82085	5-Jan-2009	
YME	CC0003473	EIM	CC0000637	YMLU3363868	20/DC	9-Dec-2008	DKCPH	UJ82764	5-Jan-2009	
YME	CC0003473	SSK	CC0002306	YMLU8114003	40/HQ	18-Nov-2008	ITVNI	UJ81750	6-Jan-2009	
YME	CC0003473	EIM	CC0000637	YMLU2829587	20/DC	29-Jul-2008	DKCPH	UJ76731	7-Jan-2009	
				CAXU3367816	20/DC	10-Dec-2008	DKCPH	UJ81932	15-Jan-2009	

SynchroNet Data							
Container #	Size/Type	Pickup Date	Pickup Location	Pickup Ref. #	Return Date	Return Location	
YMLU2920887	20FT Dry	25-Nov-2008	DKCPH	039-608226-001/010	2-Jan-2009	NLRTM	Error - Return Date Doesn't Match
YMLU3363868	20FT Dry	8-Dec-2008	DKCPH	039-613261-001/010	2-Jan-2009	NLRTM	Error - Pickup Date Doesn't Match
YMLU8114003	40FT High Cube	18-Nov-2008	ITVCE	UJ81750	16-Jan-2009	NLRTM	Warning - Pickup Location Not Found or Doesn't Match
YMLU2829587	20FT Dry	30-Jul-2008	DKCPH	039-558792-001/015	7-Jan-2009	NLRTM	Error - Pickup Date Doesn't Match
							Error - Container Not Unique

Figure 2.4: “Comparison Ctr.” error report

2.3.2 JOB PARAMETERS

The “Comparison Ctr.” job takes the usual CAS job parameters on the **General, Email Body, To/From,** and **Cc/Bcc** tabs. (As with all jobs that send results by email, the Subject: line of the email is set in the *Subject* field; see *The Job Scheduler,* page 1.)

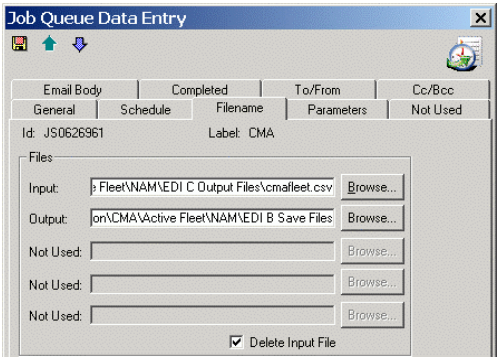

Figure 2.5: Comparison Ctr. job—file names

On the **Filename** tab (figure 2.5, page 20), *Input* (FILENAME) contains the path and name of the customer input data file, as processed by the EDI interpreter; *Output* (FILENAME2) is the SynchroNet system directory where this program will send its output spreadsheet files.

On the **Parameters** tab (figure 2.6), *Container Carrier* specifies the customer name.

Type is set to "Active Fleet" when the customer's report includes only containers that have not been returned. This limits the comparison to containers whose status in CAS is "Active," and generates an error when one does not appear in the customer's list. "Comparison" includes both active and returned containers, and does not check for this error.

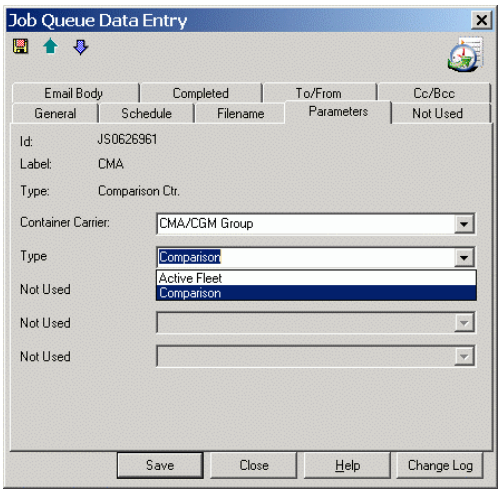

Figure 2.6: Comparison Ctr. job—parameters

2.3.3 PROGRAM DETAILS

Function name: CcpCompareContainerFile

The program's first operation is to read the value of *Type* from the job settings, determining whether the comparison is to be limited to only active containers (diagram 2-3, page 21). Next, it loads into memory all Active Transaction (Arrangement) records in which the customer appears as either the supplier or the receiver, and all Container records for the carrier.

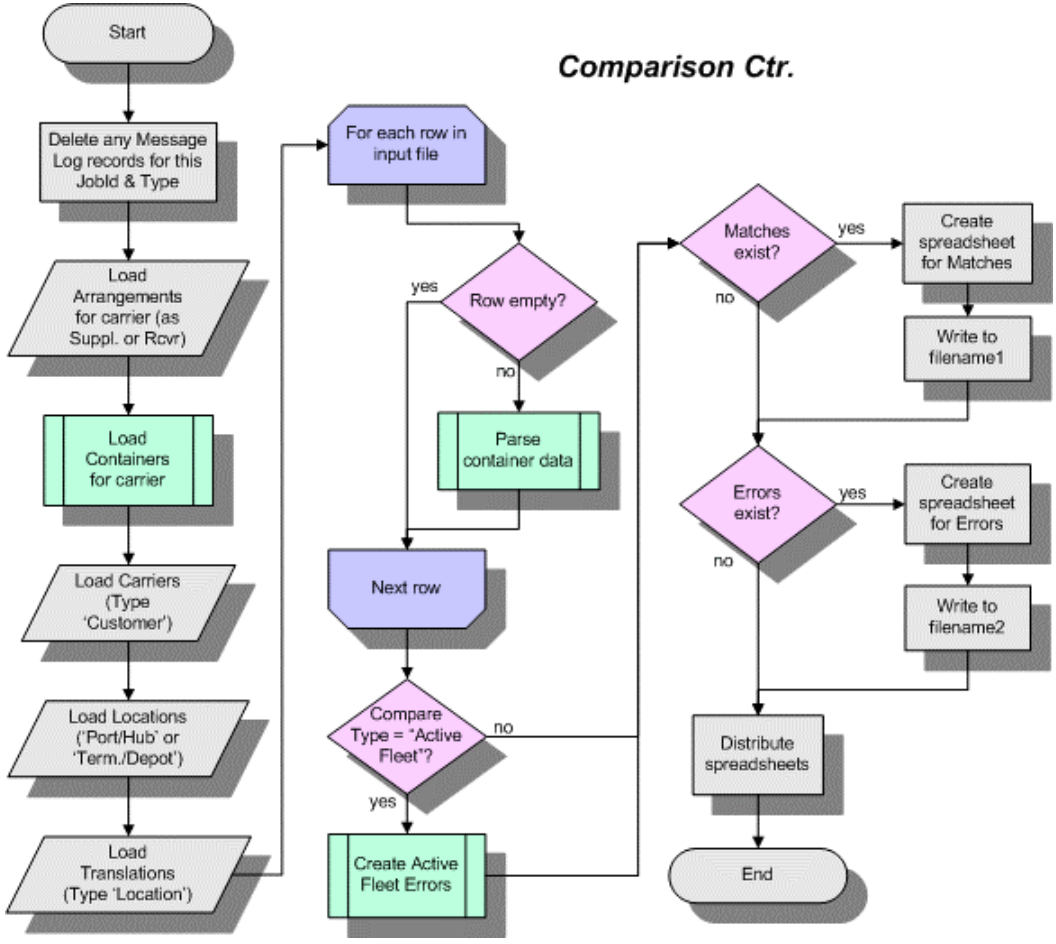

Diagram 2-3: Comparison Ctr.

2.3.3.1 Loading Container records

To retrieve the customer's containers, the program queries the database for those in which the carrier is either the owner or operator of record (OWNERNAMEID, OPERATORNAMEID), and that have nonempty pickup dates (diagram 2-4, page 22). If the comparison type specifies "active fleet" units only, the query also specifically excludes records that contain a return date (EMPTYREDELIVERY).

Next, the program examines each Container record that the query retrieved. If the container number (ASSETID) exceeds ten digits, the parser truncates it to remove the check digit, and adds the record to the array it is building. If the number is already on the list, it is added to a separate list of duplicates.

Once the containers are loaded in memory, the program continues and loads three more lists of records from the database:

1. All *Container Carrier* records of type "Customer"
2. All *Location* records of types "Port/Hub" or "Terminal/Depot"
3. All *Translation* records of type "Location"

It will refer to these when comparing lines from the input file against the database records.

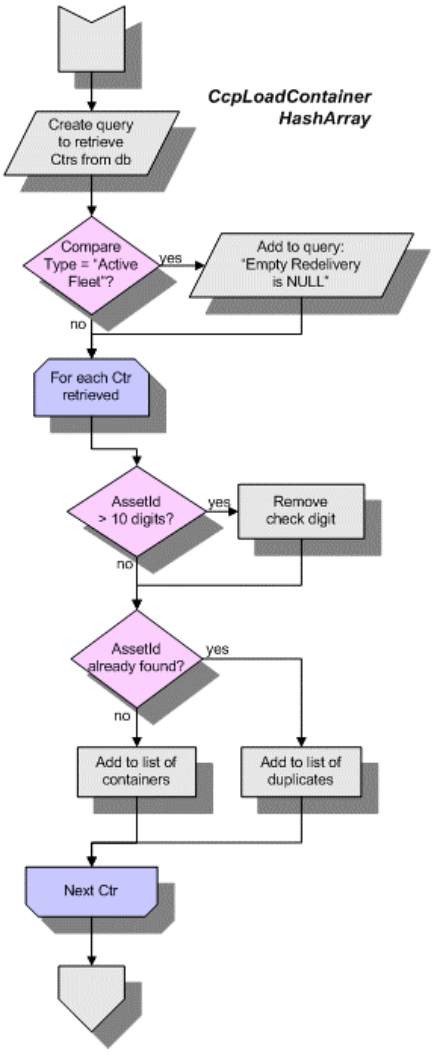

Diagram 2-4: Loading Container records

2.3.3.2 Parsing the input file

After it has loaded all the necessary database records, the program uses the function "CcpParseContainerData" (diagram 2-5, page 23) to read each line from the input file and compare it against the corresponding database record. When any comparison fails, it generates an appropriate message, which it stores as a string labeled ***Status.***

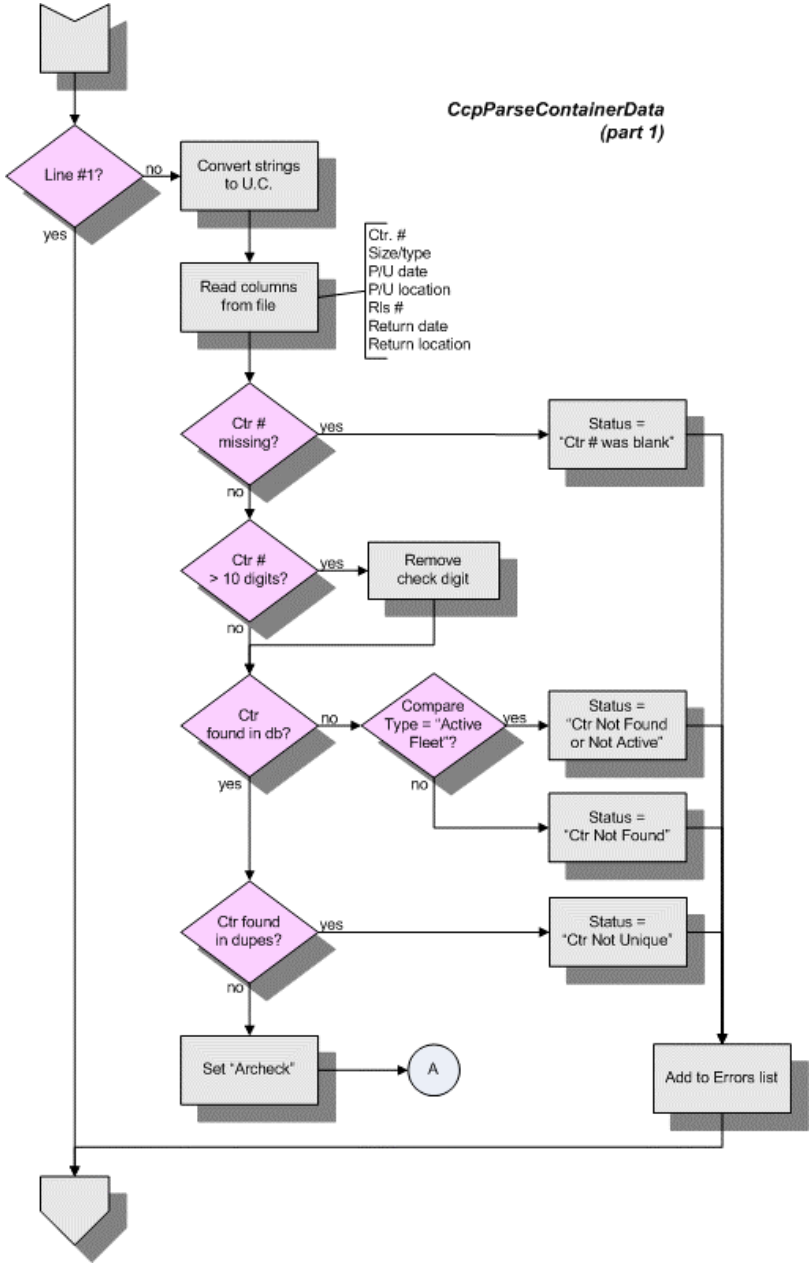

Diagram 2-5: Parsing the input file—Container number

The first part of the parser looks for errors in the container number. Ignoring the first line, which is a header row, it ensures that the first column is not empty, then

reads its contents as the container number (again truncated to remove any check digit). If the number is missing, the parser sets ***Status*** to "Container# was blank." If the number is present but not matched by a record in the database, the error message is "Container Not Found" (adding "... or Not Active" if the *Type* setting limits the job to active containers). Similarly, if the container appears in the list of duplicates (see *Loading Container records,* above), the ***Status*** message is "Container Not Unique." If an error occurs at any of these steps, the function adds the item to the Errors list. Otherwise, it sets a flag to indicate that the container appears uniquely in both CAS and the customer's list.

The program next looks for errors in the reported pickup date (diagram 2-6, page 24), verifying that it is present and is the same in both file and database. If not, it sets ***Status*** to "Pickup Date Doesn't Match."

The next check verifies that the transaction number (ARRANGEMENTID) for the container, as reported in the file, is among those retrieved for the customer. If not, and there was

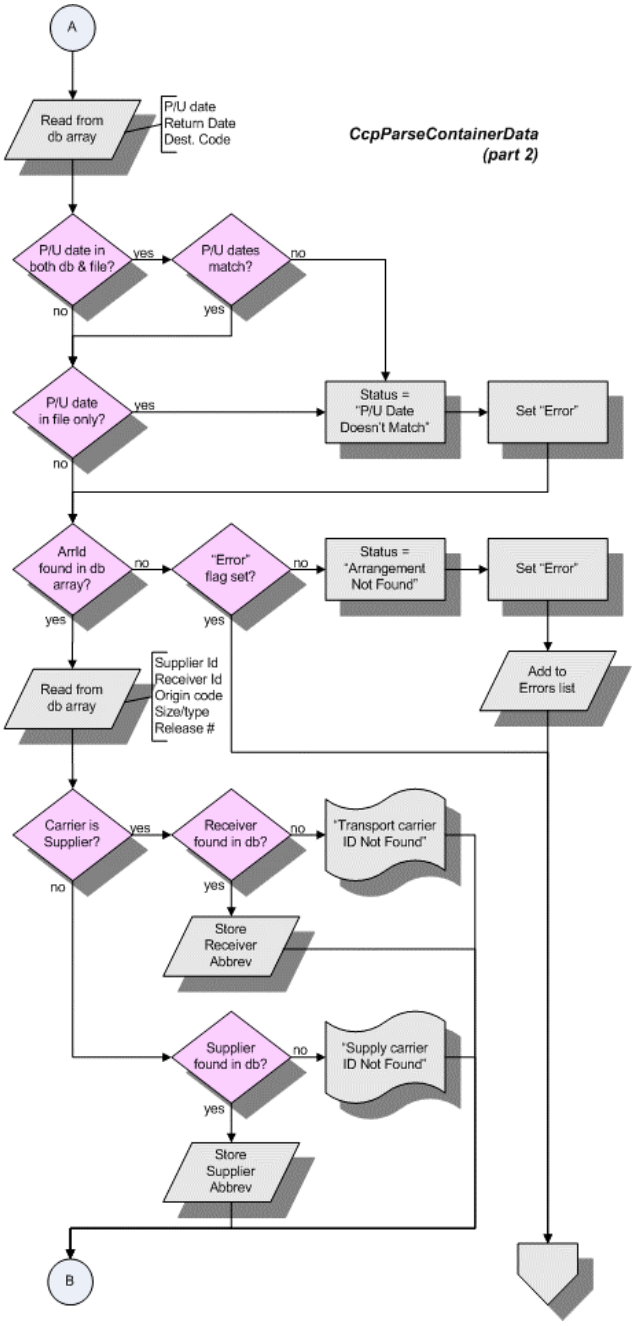

Diagram 2-6: Parsing the input file—2

no error in the pickup date, the program stores ***Status*** "Arrangement Not Found" and adds the line to the list of errors. (If there was also a pickup date error, it discards the item immediately and moves to the next input line, without issuing any error message.)

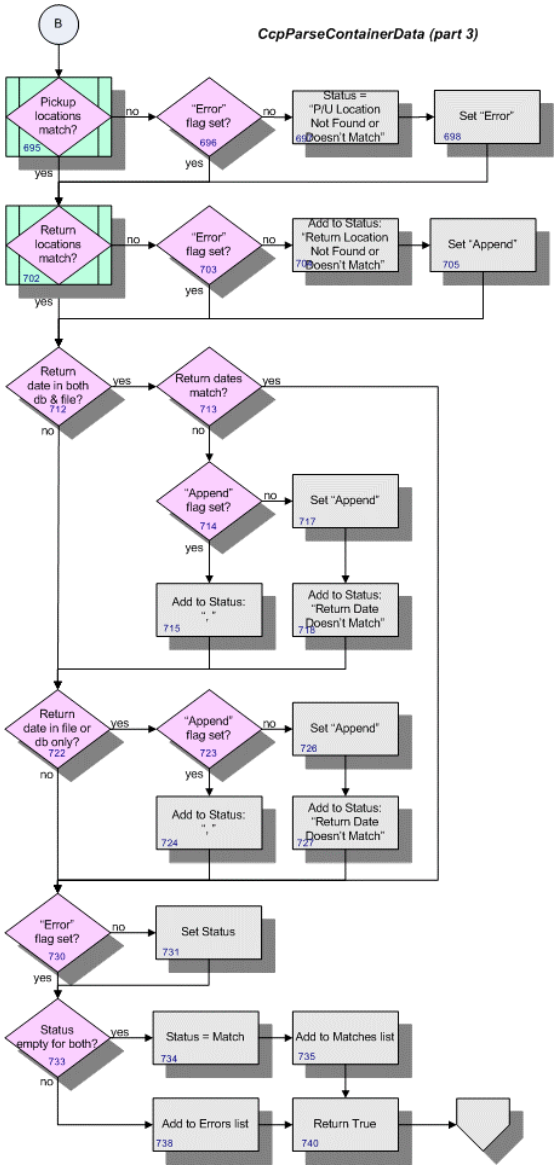

Diagram 2-7: Parsing the input file—3

If there were no container ID or pickup date errors and the Arrangement record was found in the database, the parser function proceeds to check the carrier IDs. Determining first whether the customer is the supplier or the receiver for the transaction, it retrieves the three-character system label for the other carrier, or generates an appropriate error message if that is not found. (The program will use this label later, when writing to the output spreadsheets.)

The parser next verifies that the pickup and return locations and the redelivery date in the file match those in the database, and assigns the item to either the Errors or the Matches list accordingly (diagram 2-7, page 25).

Validating locations

To validate the pickup and return locations, "Comparison Ctr." uses the function shown in diagram 2-8 (page 26), which takes as its input a location code from the input file. If the customer code is in the *TRANSLATION* table, the program gets the location from that; otherwise it verifies that the location in the file is a valid one. In either case, it then checks that the file and the database record both refer to the same location, and returns either True or False to the parser function. (Note that if the location is missing from the file, the function returns True and the program continues.)

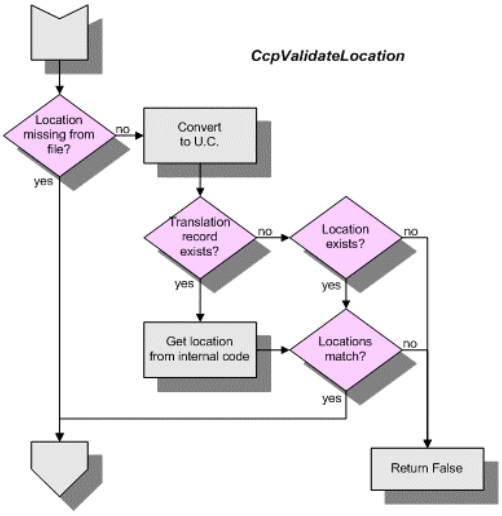

Diagram 2-8: Validating locations

The parser uses this function to validate first the pickup and then the redelivery location. If it returns False for either, and there was no pickup date or ArrId error, the program stores the appropriate "Location [Pickup or Return] Not Found or Doesn't Match" message in ***Status.*** If there was a pickup date error, it ignores the invalid location and proceeds to check the redelivery date.

Redelivery date

The final check of each line from the input file determines whether the reported redelivery date matches that in CAS. If not, or if the date is missing from the database, and the return location was valid, the program sets the error "Return Date Doesn't Match." If there was an invalid return location (and no pickup date error), it appends a comma to the existing message string ("Return Location Not Found or Doesn't Match, ").

Completing the parse

To finish the processing of each input line, if there was no error in pickup date, arrangement ID, or pickup location, the "CcpParseContainerData" function stores its final error message, which must be one of:

- return location error
- pickup date error
- no error

If ***Status*** is still empty, the program assigns it the value "Match," and adds the item to the list of matches. Otherwise, the item goes to the Errors list, from where it will be read into the errors spreadsheet.

2.3.3.3 Active fleet errors

Once the program has parsed all the lines from the input file, it checks the setting of *Comparison Type* again. If the job is limited to active fleet data, the function "CcpCreateActive-FleetErrors" examines the hash of containers retrieved from the database (diagram 2-9, page 27). For each one that the earlier parsing operation did not flag as found in both file and database (p. 24), it sets ***Status*** to "In CAS but not in file" and adds the item to the Errors list. When the *Comparison Type* setting is "Comparison," a container missing from the input file does not generate an error.

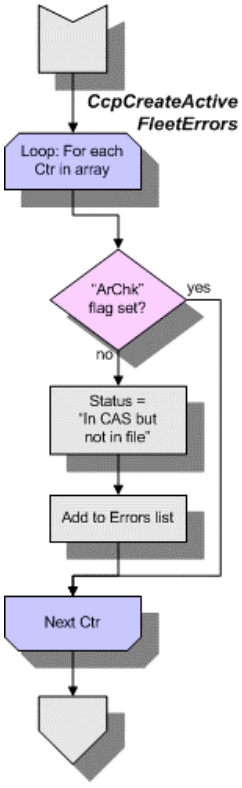

Diagram 2-9: Active fleet errors

2.3.3.4 Creating and distributing spreadsheet files

Finally, the program converts the "Errors" and "Matches" lists it has built to XML spreadsheet format, copying the error message from each line's ***Status*** to the *Status* column. It writes these to files, constructing the file names from the current date and the SynchroNet customer label (e.g. 04.30.2008 CMA-ERRORS.XLS). It completes the process by attaching these files to email messages, dispatched to the addresses specified in the **To/From** job settings.