invoice-processing

Type: Combined Analysis (Multiple Files) **Generated**: 2026-01-29 11:53:59 **Processing Time**: 29580ms

-

```
# Business Logic for Invoice Processing
This document outlines the business rules that govern the invoice processing workflow, based on an analysis of the RPG programs `FO605R` and `FO709R`. The primary focus is on the conditions that will **block** or influence the generation and processing of invoices and related order documents.
The core logic for invoice processing is contained within the `start` subroutine in `FO605R` and the `*inzsr` initialization subroutine in `FO709R`. These programs work together to ensure that invoices are generated, validated, and updated in the system while maintaining data integrity.
## Order and Invoice Validation Rules
1. **Direct Printing Trigger (`FO605R`)**
* **Logic:** Direct printing is triggered if an order number (`p_numm`) is provided as input.
* **File:** None (parameter-driven)
* **Field:** `p_numm`
* **Condition:** If `p_numm` is not equal to `*ZERO`, the program skips to the `f2taga` tag to begin processing. Otherwise, it defaults to indirect printing (`p_dirk = *ZERO`).
2. **Invoice Date Validation (`FO605R`)**
* **Logic:** Ensures the invoice date (`f2dato`) is valid and falls within the specified period (`f2peri`).
* **File:** None (parameter-driven)
* **Field:** `f2dato`, `f2peri`
* **Condition:** - The date is tested using `test(d)` to ensure it is valid.
- The month (`w_mnd2`) and year (`w_aar2`) extracted from the date must match the period's month (`w_mnd1`) and year (`w_aar1`).
- If the year difference exceeds 1 year, the process is blocked.
3. **Period Validation (`FO605R`)**
* **Logic:** Validates that the specified period (`f2peri`) is not blank and is within acceptable limits.
* **File:** None (parameter-driven)
* **Field:** `f2peri`, `w_sp_per`
* **Condition:**
- If `f2peri` is blank (`= 0`), the process is blocked.
- If the period is earlier than the "closed period" (`w_sp_per`), the
```

process is blocked.
4. **Concurrent Processing Check (`FO605R`)**
* **Logic:** Prevents invoice processing if another screen is already handling the same task.
* **File:** None (external program call)
* **Field:** `w_onof`
* **Condition:**
- Calls `FA730R` with `w_kode = 01` to check if invoicing is active.
- If `w_onof = 1`, the process is terminated.
5. **Order Locking (`FO709R`)**
* **Logic:** Updates the order header to indicate the user and workstation currently processing the order.
* **File:** `FOHELF` (Order Header File)
* **Field:** `FOKOUS`, `FOKOWS`, `FOKODA`, `FOKOTI`
* **Condition:**
- The order is locked by updating the fields with the current user (`l_user`), workstation (`l_wsid`), and timestamp (`w_koda`, `w_koti`).
- If the order is not found (`*IN90 = *ON`), no update occurs.
6. **Job Queue Assignment (`FO605R`)**
* **Logic:** Allows the user to place the job in a job queue for later processing.
* **File:** None (parameter-driven)
* **Field:** `p_dirk`
* **Condition:** If the user presses F4 (`*INKD = *ON`), `p_dirk` is set to `2`, indicating the job should be queued.
7. **Order Header Update (`FO709R`)**
* **Logic:** Updates the order header with tracking information, including the user, workstation, and timestamp.
* **File:** `FOHELF` (Order Header File)
* **Field:** `FOEDAT`, `FOETIM`, `FOEUSR`
* **Condition:** The order header is updated with the current date and time (`FOEDAT`, `FOETIM`) and the user (`FOEUSR`).
## Subprogram Calls Affecting Logic
Beyond the direct file checks, several external subprograms are called that play a significant role in the workflow:
1. **`FA730R` (Concurrent Processing Check)**
* **Trigger:** Called twice in `FO605R` to check if invoicing or accounting transfer is active.
* **Logic:** Sets the `w_onof` flag to `1` if another process is active.
* **Impact:** Blocks the current process if another screen is handling invoicing or accounting transfer.
2. **`FA720R` (Job Register Update)**

* **Trigger:** Called in `FO605R` to update the job register with an active flag.
* **Logic:** Marks the job as active by setting `w_onof = 1`.
* **Impact:** Ensures the job is registered as active for tracking purposes.
3. **`FA920R` (Member Number Generation)**
* **Trigger:** Called in `FO605R` during initialization to generate a new member number.
* **Logic:** Retrieves the next available member number for the file.
* **Impact:** Ensures unique member numbers for each invoice batch.
4. **`FO604C` (Member Usage Check)**
* **Trigger:** Called in `FO605R` to check if a member is already in use.
* **Logic:** Verifies if the generated member number is already used.
* **Impact:** Prevents duplicate member usage by regenerating a new member if necessary.
5. **`FO612C` (Print Start)**
* **Trigger:** Called in `FO605R` to initiate the printing process.
* **Logic:** Passes the member name (`d_memb`) to the CL program to start printing.
* **Impact:** Begins the actual printing of invoices or credit notes.
## Differences Between Programs
1. **Primary Purpose:**
- `FO605R` focuses on preparing and validating invoices for printing, including handling periods, dates, and concurrent processing.
- `FO709R` is responsible for updating the order header with tracking information, such as the user and workstation handling the order.
2. **File Usage:**
- `FO605R` interacts with multiple files, including `FPM1LUR` (Order Picking Register) and `RAA1L1R` (Status Codes), to validate periods and update parameters.
- `FO709R` exclusively updates the `FOHELF` (Order Header File) to lock the order for processing.
3. **Subprogram Calls:**
- `FO605R` makes extensive use of subprograms (`FA730R`, `FA720R`, `FA920R`, `FO604C`, `FO612C`) to handle validation, job registration, and printing.
- `FO709R` does not call any external subprograms but focuses solely on updating the order header.
4. **Blocking Conditions:**
- `FO605R` includes multiple blocking conditions related to invalid periods, dates, and concurrent processing.

- `FO709R` does not include blocking conditions but ensures the order is locked for processing.
## Summary of Workflow
1. **Initialization:**
- `FO605R` initializes parameters and validates input data, including periods and dates.
- `FO709R` initializes the order header keys and locks the order for processing.
2. **Validation:**
- `FO605R` performs extensive validation on periods, dates, and concurrent processing.
- `FO709R` ensures the order header is updated with tracking information.
3. **Execution:**
- `FO605R` triggers the printing process by calling `FO612C`.
- `FO709R` updates the order header to reflect the current user and workstation.
By combining the functionality of `FO605R` and `FO709R`, the system ensures that invoices are processed accurately, periods and dates are validated, and orders are locked to prevent concurrent modifications. These programs work together to maintain data integrity and streamline invoice processing.