📋 PBFJE1R - TPM Location Master Customer Relationship Editor - Business Documentation

## 🎯 Section 8: Detailed Business Functions Analysis

### 🔍 Function Overview

The PBFJE1R program contains45 business functionsorganized into 7 functional layers. This section provides detailed analysis of the 10 most critical functions that drive core business operations, followed by a comprehensive catalog of all functions.

#### Function Distribution

* Initialization Functions:1 function (System setup and configuration)
* Key Screen Processing:5 functions (Primary user interface management)
* Detail Screen Processing:8 functions (Customer relationship management)
* Validation Functions:12 functions (Data quality assurance)
* Data Access Functions:10 functions (Database operations)
* External Services:4 functions (Integration points)
* Utility Functions:5 functions (Supporting operations)

### 🎯 Critical Business Functions (Detailed Analysis)

#### 1. System Initialization (ZZINIT)

##### 🎯 Parameters and Business Data Elements

|  |  |  |
| --- | --- | --- |
| **Data Element** | **Business Purpose** | **Source/Target** |
| Initial Call Indicator (W0ICL) | Determines if this is the first execution or subsequent processing | System control flag |
| User Profile (USRNME) | Identifies authorized user for security and audit | System security context |
| Session Date/Time | Records when business transaction was initiated | System timestamp |
| Program Status (W0PGS) | Tracks current operational mode (ADD/CHANGE/DELETE/INQUIRY) | User selection |
| Screen Control Flags | Manages display elements and field protection | Application configuration |

##### 🎯 Business Logic Summary

The System Initialization function establishes the operational environment for all TPM Location Customer Relationship transactions. This function executes once per user session and configures security contexts, validates user authority, initializes screen control variables, and prepares the application for either ADD, CHANGE, DELETE, or INQUIRY operations. The function ensures that all subsequent business operations occur within a properly configured and authorized environment.

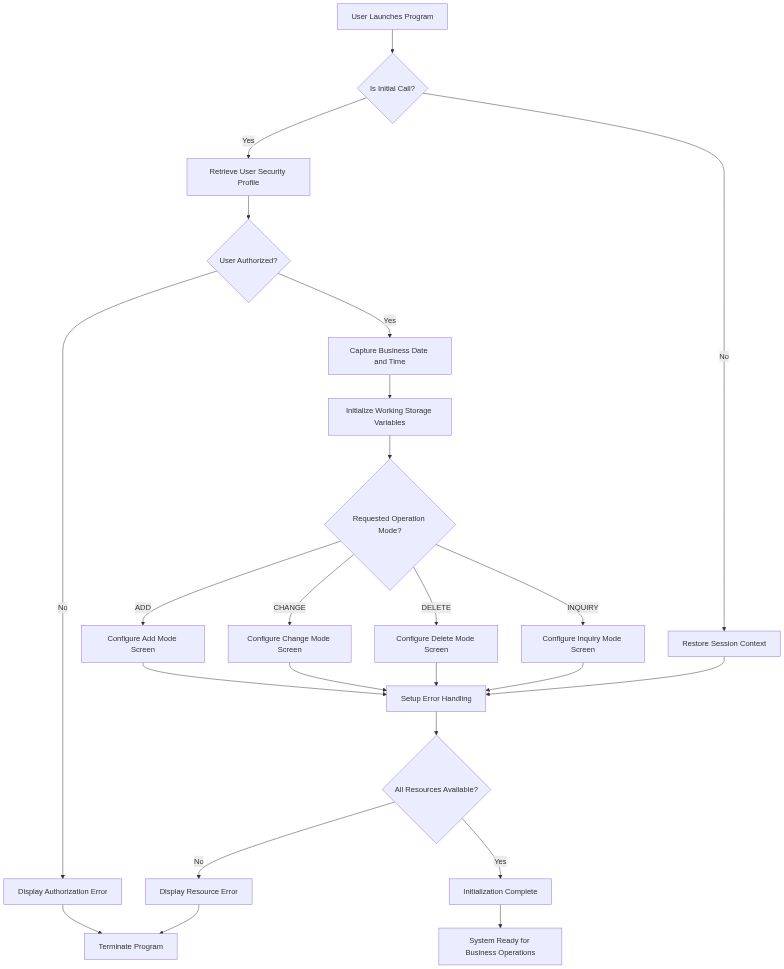
##### 🔄 Detailed Logic Explanation

When a user launches the TPM Location Customer Relationship Editor, the System Initialization function first determines whether this is the initial call by checking the Initial Call Indicator. If this is the first execution, the function retrieves the user's security profile from the system and validates their authority to perform master data maintenance operations. The function then captures the current business date and time for audit trail purposes and initializes all working storage variables to their default values.

Next, the function analyzes the requested operation mode (ADD, CHANGE, DELETE, or INQUIRY) and configures the screen control flags accordingly. For ADD operations, all input fields are enabled and cleared. For CHANGE operations, the function prepares for record retrieval and modification. For DELETE operations, the function sets up confirmation prompts and soft-delete processing. For INQUIRY mode, all fields are set to display-only to prevent accidental modifications.

The function also initializes error handling mechanisms, sets up message queues for user notifications, and establishes database connection contexts. Before completing, the function performs a final validation to ensure all required system resources are available and accessible. If any initialization step fails, the function immediately displays an appropriate error message and terminates the program to prevent data corruption or unauthorized access.

**📊 Figure 1: Process Flow Diagram**



*Diagram 1*

##### 🎯 Data Interaction and Business Information Management

The System Initialization function reads from theUser Profile Master (USRPRF)to validate security credentials and retrieve authorization levels. It also accesses theSystem Configuration File (SYSCFG)to obtain application-specific parameters such as default date formats, decimal notation standards, and field-level security rules. The function writes initialization audit records to theApplication Log File (APPLOG)to maintain compliance with audit requirements. All session context information is stored in working storage variables for use by subsequent functions throughout the user's transaction session.

#### ⚙️ 2. Update Record Processing (SHCHRC)

##### 🎯 Parameters and Business Data Elements

|  |  |  |
| --- | --- | --- |
| **Data Element** | **Business Purpose** | **Source/Target** |
| TPM Location Code (TPMLOC) | Unique identifier for manufacturing location | Key screen input |
| Customer Assignments (Array 1-27) | Complete list of customer relationships requiring update | Detail screen modifications |
| Effective Date (EFFDTE) | Business date when changes become active | User input with validation |
| Modified By User (MODUSER) | Audit trail of who authorized the change | System user profile |
| Modification Timestamp (MODTMS) | Precise time when change was committed | System timestamp |
| Record Lock Indicator | Prevents concurrent modification conflicts | Database lock mechanism |

##### 🎯 Business Logic Summary

The Update Record Processing function handles all modifications to existing TPM Location Customer Relationships. This is the largest and most complex function in the system, managing the complete change lifecycle including record locking, validation of all 27 potential customer assignments, effective date processing, cross-reference updates, and audit trail maintenance. The function ensures data integrity through comprehensive validation, prevents concurrent update conflicts through pessimistic locking, and maintains referential integrity across all related business entities.

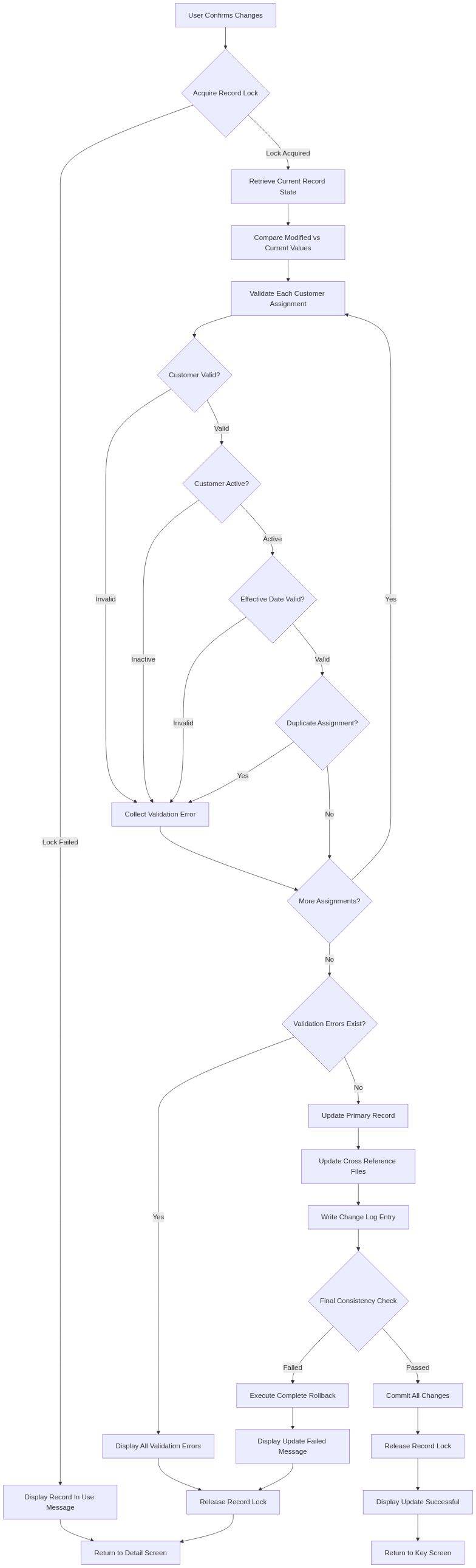
##### 🔄 Detailed Logic Explanation

When a user commits changes to a TPM Location Customer Relationship, the Update Record Processing function first attempts to lock the target record using the TPM Location Code as the key. If another user currently holds the lock, the function displays a "Record In Use" message and prevents the update to avoid data conflicts. Once the lock is acquired, the function retrieves the current record state from the database and compares it against the modified values entered by the user.

The function then validates each of the 27 potential customer assignments individually. For each assigned customer, it verifies that the customer number exists in the Customer Master file, confirms the customer is in active status, validates the effective date is not in the past, and checks for duplicate assignments within the same TPM location. The function also validates cross-reference integrity by ensuring that any modified customer assignments do not create orphaned records in dependent files such as Order History or Shipment Tracking.

After validation succeeds, the function updates the primary record in the TPM Location Customer Relationship file using the UPDATE operation. It then processes any secondary updates required in cross-reference files, historical tracking tables, and aggregate summary files. The function writes a detailed change log entry documenting the specific fields modified, old values, new values, user ID, and timestamp. Before releasing the record lock, the function performs a final consistency check to verify all related records remain in a valid state. If any step in the update process fails, the function executes a complete rollback to restore the original state and displays an appropriate error message to the user.

**📊 Figure 2: Process Flow Diagram**



*Diagram 2*

##### 🎯 Data Interaction and Business Information Management

The Update Record Processing function performs complex interactions with multiple database files. It updates the primaryTPM Location Customer Relationship file (@APREOJ)with modified assignment data. It reads from theCustomer Master file (@ACUSMR)to validate customer numbers and status. It updates theCustomer Location Cross Reference file (@ACSLCR)to maintain bidirectional relationships. The function writes to theMaster Data Change Log (@AMDCHL)to document all modifications for audit and compliance purposes. It also reads from theOrder Header file (@AORHDR)andShipment Master file (@ASHPMR)to perform referential integrity checks before committing changes.

#### ⚙️ 3. Create Record Processing (SDCRRC)

##### 🎯 Parameters and Business Data Elements

|  |  |  |
| --- | --- | --- |
| **Data Element** | **Business Purpose** | **Source/Target** |
| TPM Location Code (TPMLOC) | New unique identifier for manufacturing location | User input with uniqueness validation |
| Location Description (LOCDSC) | Business name of third-party manufacturing facility | User input |
| Initial Customer Assignments | First set of customer relationships (minimum 1, maximum 27) | Detail screen entries |
| Created By User (CRTUSER) | Audit trail of who authorized the creation | System user profile |
| Creation Timestamp (CRTTMS) | Precise time when record was established | System timestamp |
| Default Status Code | Sets initial active status for new relationship | System default value |

##### 🎯 Business Logic Summary

The Create Record Processing function establishes new TPM Location Customer Relationships in the master database. This function validates that the proposed TPM Location Code is unique, verifies all initial customer assignments reference valid active customers, ensures at least one customer is assigned, creates the primary relationship record with proper audit fields, establishes all required cross-reference entries, and initializes historical tracking. The function enforces business rules that prevent duplicate location codes and ensure data quality from the moment of creation.

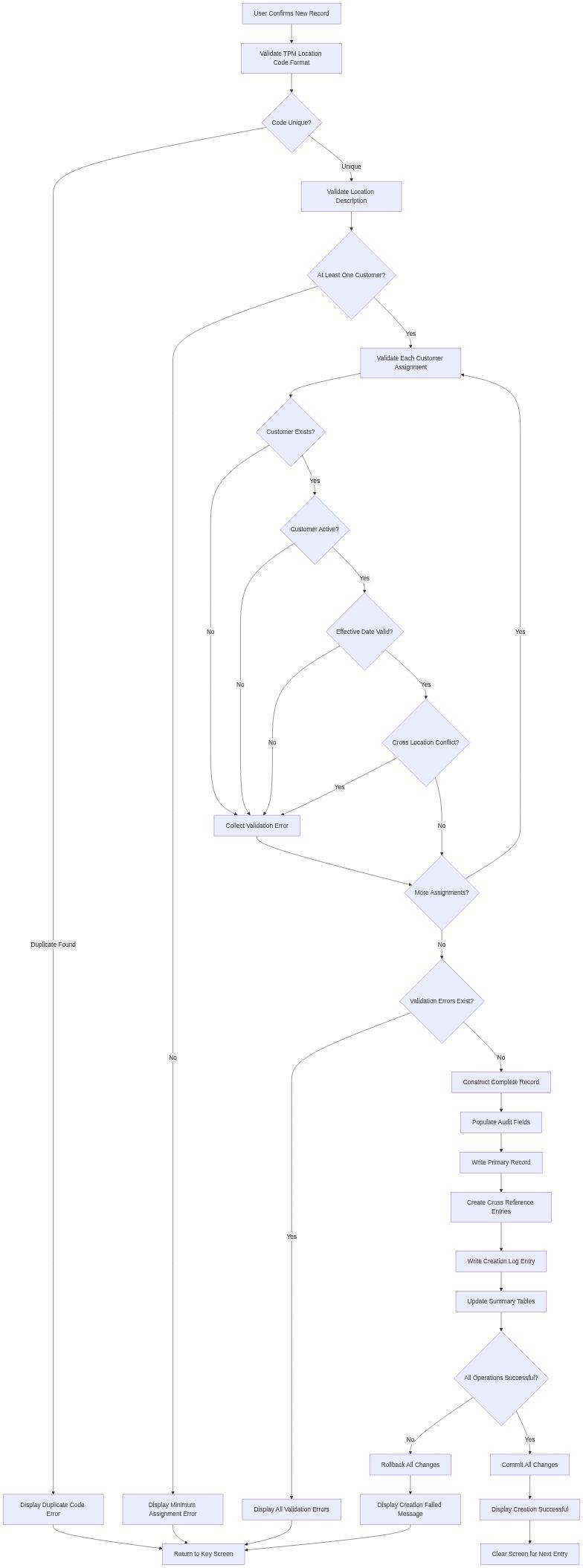
##### 🔄 Detailed Logic Explanation

When a user submits a new TPM Location Customer Relationship, the Create Record Processing function begins by validating the uniqueness of the proposed TPM Location Code. It searches the TPM Location Customer Relationship file to ensure no existing record uses this identifier. If a duplicate is found, the function immediately rejects the creation and displays an error message instructing the user to select a different code. The function also validates that the location description contains meaningful text and does not consist entirely of special characters or whitespace.

Next, the function validates all customer assignments entered on the detail screen. It requires at least one valid customer assignment for the record to be created, as a TPM location without customer relationships serves no business purpose. For each customer number entered, the function verifies existence in the Customer Master file, confirms active status, validates that the effective date is today or future, and checks for duplicate entries within the assignment array. The function also performs cross-location validation to identify if any of the assigned customers already have relationships with other TPM locations that might create conflicting business rules.

After all validations pass, the function constructs the complete record with proper default values for all required fields. It populates audit fields including Created By User from the system security context and Creation Timestamp from the current system time. The function sets the default status to Active and initializes all tracking counters to zero. The function then writes the new record to the TPM Location Customer Relationship file using the WRITE operation. Following the primary record creation, the function creates corresponding entries in all cross-reference files to maintain data consistency. It writes initial entries to the Master Data Change Log documenting the creation event, and it updates aggregate summary tables that track total customer assignments per location. If any database operation fails during creation, the function performs a complete rollback of all partially created records and displays an error message to the user.

**📊 Figure 3: Process Flow Diagram**



*Diagram 3*

##### 🎯 Data Interaction and Business Information Management

The Create Record Processing function writes the new primary record to theTPM Location Customer Relationship file (@APREOJ)with all required fields populated. It reads from theCustomer Master file (@ACUSMR)to validate customer numbers and retrieve customer details. It creates entries in theCustomer Location Cross Reference file (@ACSLCR)for bidirectional navigation. The function writes to theMaster Data Change Log (@AMDCHL)to document the creation event with full audit trail information. It also updates theLocation Summary file (@ALOCSM)to reflect the new location in aggregate reports and dashboards.

#### 4. Validate Detail Screen (DCVLDL)

##### 🎯 Parameters and Business Data Elements

|  |  |  |
| --- | --- | --- |
| **Data Element** | **Business Purpose** | **Source/Target** |
| Customer Number Array (1-27) | All customer assignments requiring validation | Detail screen input fields |
| Effective Date Array (1-27) | Business dates when each assignment becomes active | Detail screen date fields |
| Error Message Array | Collection of validation failures for user feedback | Working storage accumulator |
| Field Position Array | Screen coordinates to highlight invalid fields | Display control structure |
| Validation Return Code | Overall success or failure indicator | Function return value |

##### 🎯 Business Logic Summary

The Validate Detail Screen function performs comprehensive quality assurance on all customer assignment data entered by users. This critical validation function ensures data integrity before any database modifications occur. It validates customer number existence and status, verifies effective dates are logical and within business rules, checks for duplicate customer assignments within the same location, validates cross-file referential integrity, and accumulates all errors for complete user feedback rather than stopping at the first error encountered.

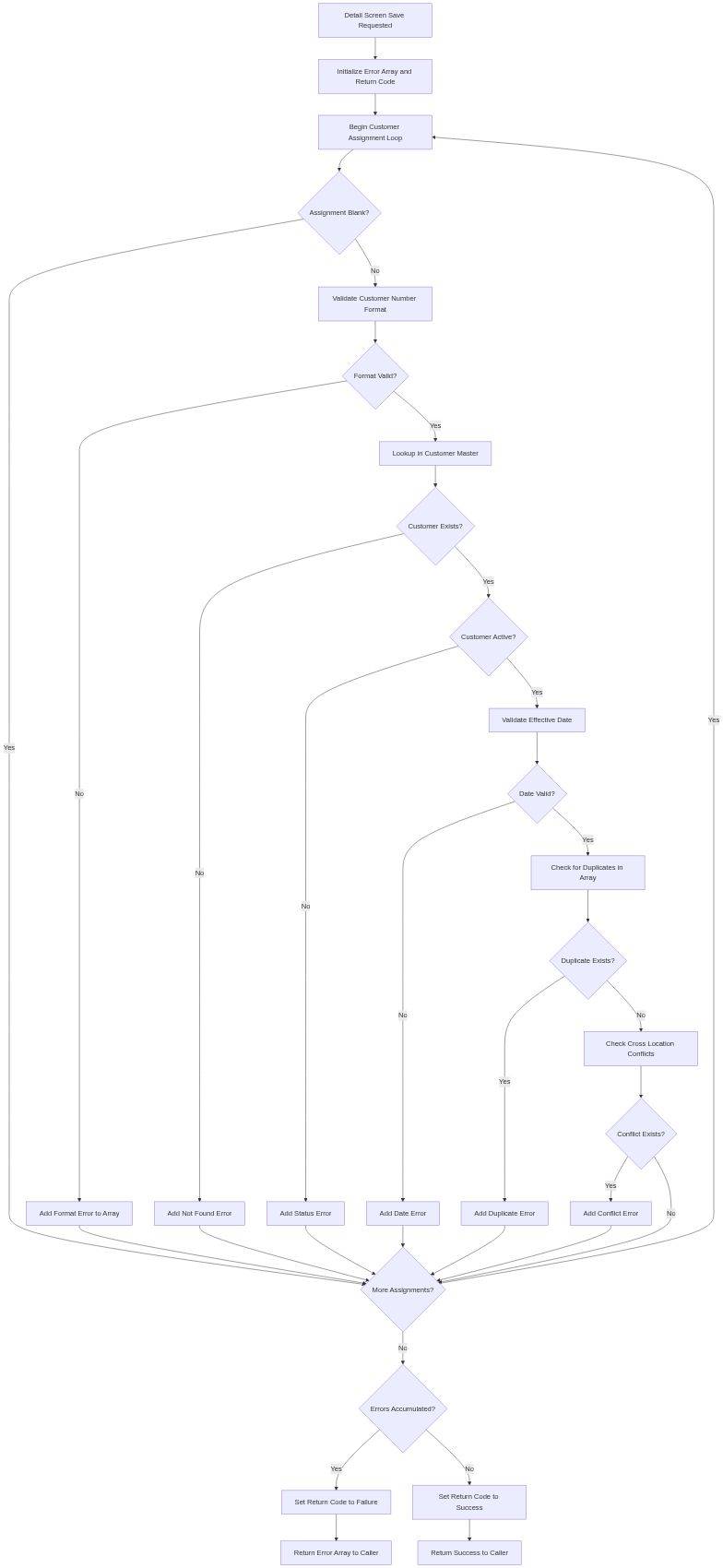
##### 🔄 Detailed Logic Explanation

When the user attempts to save changes on the detail screen, the Validate Detail Screen function executes before any database writes occur. The function begins by initializing the error message array and validation return code to indicate success. It then iterates through all 27 potential customer assignment positions, processing each entry that contains data. For positions left blank by the user, the function skips validation as empty assignments are permitted.

For each populated customer assignment, the function first validates the customer number format to ensure it conforms to the organization's customer numbering standards. It then performs a database lookup in the Customer Master file to verify the customer exists. If found, the function checks the customer status field to ensure the customer is in active status, as inactive or suspended customers cannot be assigned to TPM locations. Next, the function validates the effective date by comparing it against the current business date to prevent backdating that would create historical inconsistencies. The function also validates that the effective date does not extend too far into the future based on business policy.

The function then performs duplicate detection by comparing the current customer number against all other assignments in the array to ensure the same customer is not listed multiple times for the same location. It also executes cross-location validation by querying the Customer Location Cross Reference file to identify if this customer already has active assignments with other TPM locations that might violate exclusive relationship rules. If any validation rule fails, the function adds a descriptive error message to the error array and records the field position for screen highlighting, but continues processing remaining assignments to provide complete feedback. After validating all assignments, the function returns the accumulated error list to the calling program, which displays all errors simultaneously and positions the cursor at the first invalid field.

**📊 Figure 4: Process Flow Diagram**



*Diagram 4*

##### 🎯 Data Interaction and Business Information Management

The Validate Detail Screen function reads from theCustomer Master file (@ACUSMR)to verify customer existence and retrieve status information. It queries theCustomer Location Cross Reference file (@ACSLCR)to detect cross-location assignment conflicts. The function does not write to any files as it performs read-only validation operations. All validation results are returned to the calling function through working storage variables and error arrays that control screen display and field highlighting.

#### 5. Process Key Screen Input (BEPRKY)

##### 🎯 Parameters and Business Data Elements

|  |  |  |
| --- | --- | --- |
| **Data Element** | **Business Purpose** | **Source/Target** |
| TPM Location Code (TPMLOC) | User-entered location identifier to process | Key screen input field |
| Function Key Pressed | User action selection (Enter/F6 Add/F11 Delete/F12 Cancel) | Keyboard input |
| Record Retrieved Flag | Indicates whether location was found in database | Database operation result |
| Record Lock Status | Indicates if another user currently editing this location | Database lock indicator |
| Next Screen Indicator | Determines which screen to display next | Program control variable |

##### 🎯 Business Logic Summary

The Process Key Screen Input function serves as the primary navigation controller for the application. It interprets user actions on the key screen, validates the TPM Location Code entered, attempts to retrieve the corresponding record from the database, determines the appropriate next screen based on the function key pressed and record existence, manages record locking for concurrent access protection, and routes program flow to the correct operation mode (ADD/CHANGE/DELETE/INQUIRY).

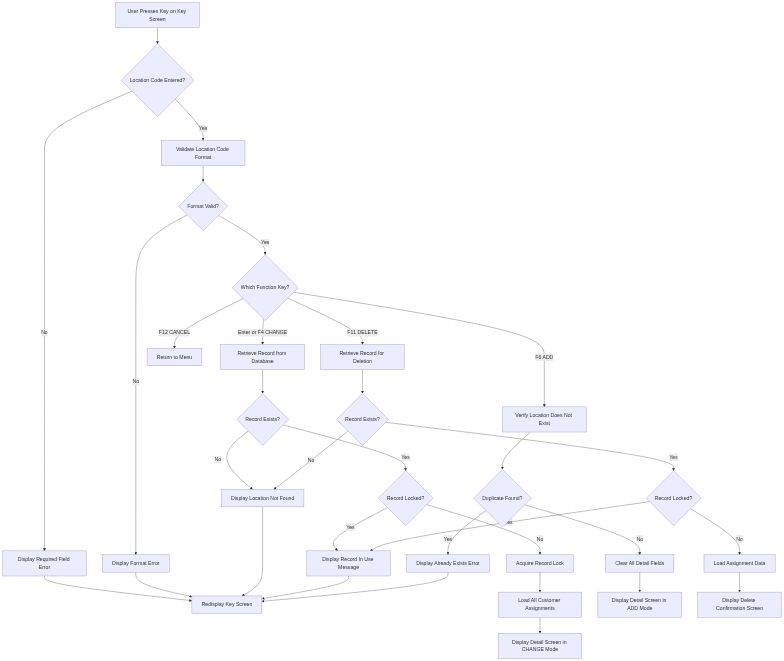
##### 🔄 Detailed Logic Explanation

When the user presses Enter or a function key on the key screen, the Process Key Screen Input function first validates that a TPM Location Code was entered. If the location field is blank, the function displays an error message and redisplays the key screen. If a location code is present, the function validates the format to ensure it conforms to business standards for location identifiers.

Next, the function determines which function key the user pressed to understand the requested operation. If the user pressed Enter or F4 (CHANGE), the function attempts to retrieve the existing record from the TPM Location Customer Relationship file using CHAIN operation with the location code as the key. If the record is found, the function checks the record lock status. If another user currently holds the lock, the function displays a "Record In Use By User X" message and returns to the key screen. If the lock is available, the function acquires the lock and transitions to the detail screen in CHANGE mode, loading all 27 customer assignments for modification. If the record is not found, the function displays a "Location Not Found" error and returns to the key screen.

If the user pressed F6 (ADD), the function verifies that the entered location code does not already exist in the database. If a duplicate is found, the function displays an error indicating the location already exists and suggests using CHANGE mode instead. If the location is unique, the function transitions to the detail screen in ADD mode with all fields cleared for new entry. If the user pressed F11 (DELETE), the function retrieves the record, verifies it is not locked, and transitions to a confirmation screen showing the current assignments before deletion. The function ensures users cannot accidentally delete records by requiring explicit confirmation before soft-delete processing occurs.

**📊 Figure 5: Process Flow Diagram**



*Diagram 5*

##### 🎯 Data Interaction and Business Information Management

The Process Key Screen Input function reads from theTPM Location Customer Relationship file (@APREOJ)using CHAIN operation with TPM Location Code as the key. For CHANGE and DELETE operations, it retrieves complete records including all 27 customer assignments. For ADD operations, it performs existence checks to prevent duplicate key violations. The function manages database record locks through the file's lock indicator fields to coordinate concurrent user access. All retrieved data is loaded into working storage variables for subsequent display and modification on the detail screen.

#### ✓ 6. Change Record Validation (SFCHRC)

##### 🎯 Parameters and Business Data Elements

|  |  |  |
| --- | --- | --- |
| **Data Element** | **Business Purpose** | **Source/Target** |
| Original Record Image | Snapshot of record state before user modifications | Database original values |
| Modified Record Image | User-entered changes requiring validation | Screen field values |
| Change Detection Flag | Indicates whether any fields were actually modified | Comparison result |
| Field Change Indicators (27) | Individual flags showing which assignments changed | Field-level comparison array |
| Authorization Level | User permission level for change approval | Security profile |

##### 🎯 Business Logic Summary

The Change Record Validation function ensures that CHANGE operations contain actual modifications and that users have appropriate authority to make the requested changes. The function compares the original record state against modified screen values to detect changes, validates that at least one field was modified, enforces field-level security rules based on user authorization, determines if changes require managerial approval based on business policy, and routes the transaction to appropriate approval workflows if necessary.

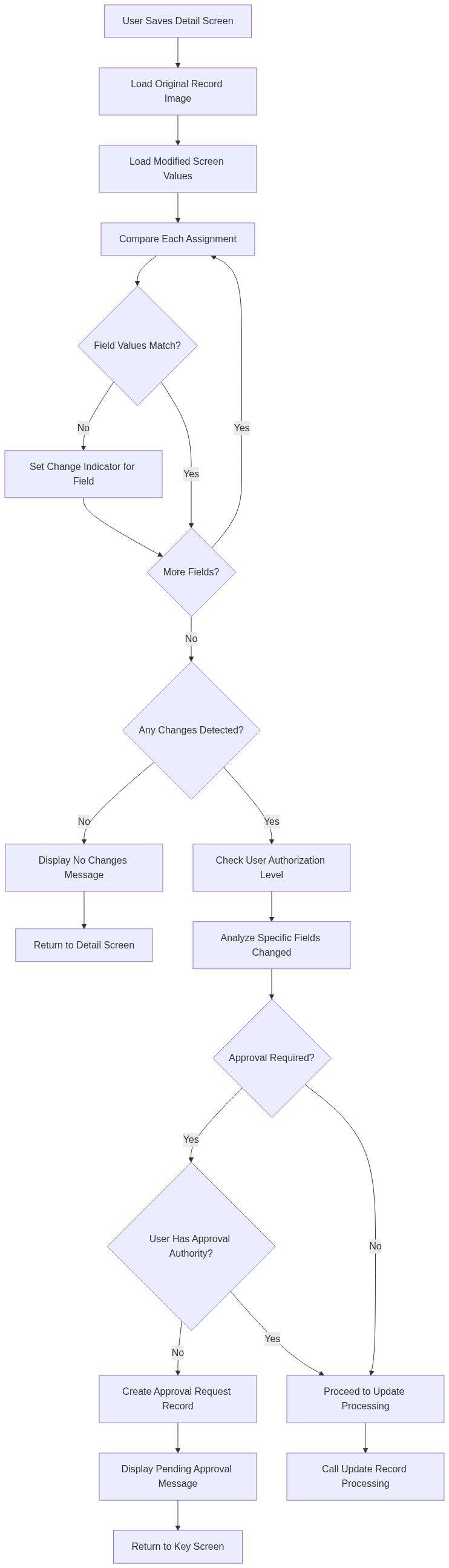
##### 🔄 Detailed Logic Explanation

When a user attempts to save changes from the detail screen in CHANGE mode, the Change Record Validation function compares each field in the modified screen image against the corresponding field in the original database record. The function maintains an array of 27 change indicators, one for each potential customer assignment position. For each position, if the customer number or effective date differs between the original and modified values, the function sets the corresponding change indicator to true.

After comparing all fields, the function checks if any change indicators are set to true. If no changes are detected, the function displays an informational message stating "No Changes Detected" and returns to the detail screen without performing any database operations, preventing unnecessary processing and audit log entries for non-changes. If changes are detected, the function examines the specific fields that were modified and consults the user's security profile to verify authorization. Certain customer assignments may be restricted based on business policies such as exclusive relationships or high-value accounts requiring managerial approval.

For changes requiring approval, the function determines if the current user has sufficient authority or if the changes must be routed to a supervisor approval queue. High-impact changes such as removing major customer assignments or adding customers with special contractual terms trigger approval workflows. The function sets appropriate workflow flags and generates approval request records in the pending transaction file. For standard changes that the user is authorized to complete directly, the function proceeds to call the Update Record Processing function to commit the modifications.

**📊 Figure 6: Process Flow Diagram**



*Diagram 6*

##### 🎯 Data Interaction and Business Information Management

The Change Record Validation function reads the original record state from working storage variables populated during key screen processing. It accesses theUser Security Profile file (USRSEC)to retrieve authorization levels and field-level permissions. For changes requiring approval, it writes to thePending Transaction file (@APNDTR)to create approval request records. The function does not directly update master files; it delegates actual database updates to the Update Record Processing function after validation succeeds.

#### 7. Confirm Action (EAPMCF)

##### 🎯 Parameters and Business Data Elements

|  |  |  |
| --- | --- | --- |
| **Data Element** | **Business Purpose** | **Source/Target** |
| Action Type (ADD/CHANGE/DELETE) | Operation requiring user confirmation | Program control variable |
| Confirmation Message Text | Human-readable description of action impact | Constructed message string |
| User Response (Yes/No) | User's confirmation decision | Confirmation screen input |
| Confirmation Timestamp | Precise time user confirmed action for audit | System timestamp |
| Action Cancelled Flag | Indicates if user declined to proceed | Function return value |

##### 🎯 Business Logic Summary

The Confirm Action function implements a critical safety mechanism to prevent accidental execution of high-impact operations. This function displays a confirmation screen with clear explanation of consequences, requires explicit user approval before proceeding with destructive operations, provides opportunity to cancel without changes, records confirmation decisions in audit trail, and differentiates confirmation messages based on operation type and data impact.

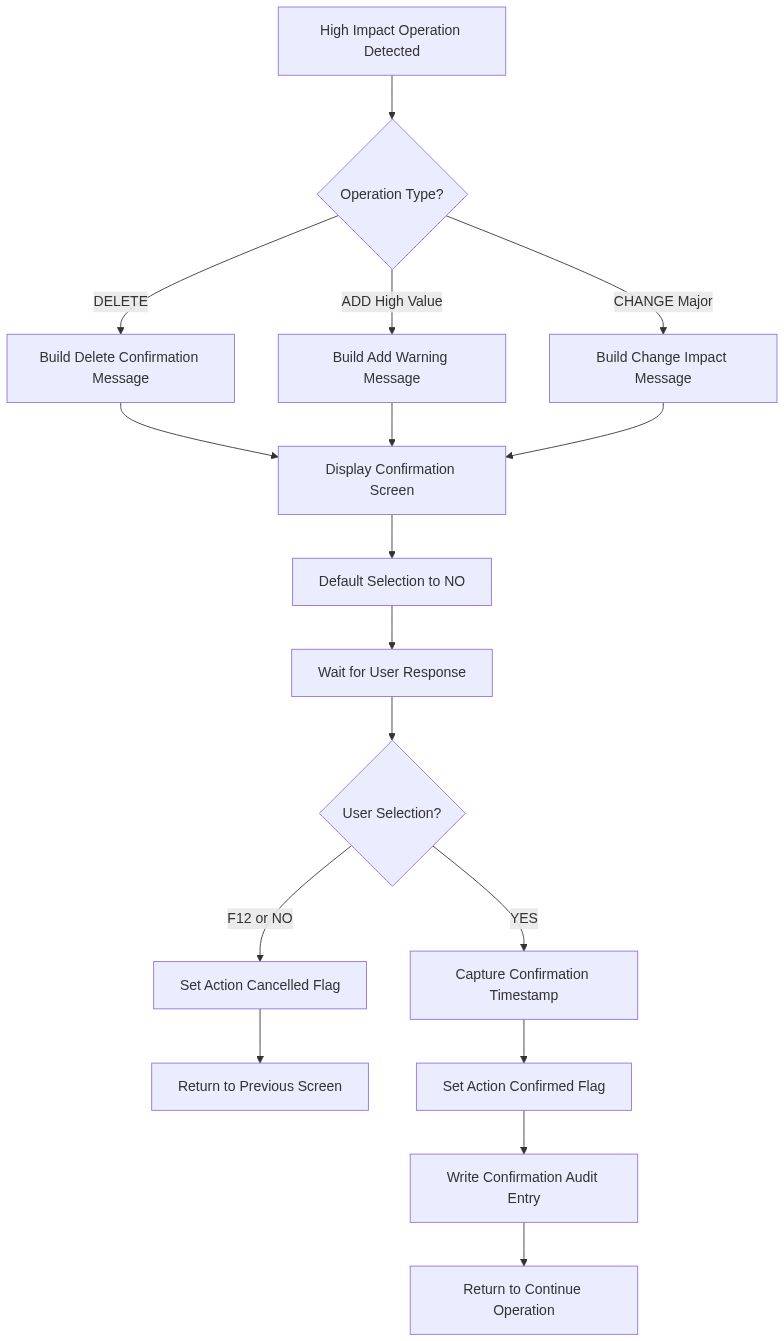
##### 🔄 Detailed Logic Explanation

When a potentially destructive or high-impact operation is about to execute, the Confirm Action function intercepts program flow to display a confirmation screen. The function first constructs a context-specific confirmation message based on the operation type. For DELETE operations, the message includes the TPM Location Code and the number of active customer assignments that will be affected by the deletion. For ADD operations involving high-value customers, the message warns about creating new relationships that may have contractual implications. For CHANGE operations that remove major customer assignments, the message lists the specific customers being removed and potential impact on order processing.

The function displays this confirmation message on a dedicated screen format with clear YES and NO options. The screen defaults to NO selection to prevent accidental confirmations from users who press Enter without reading. The function waits for user response and validates that the user explicitly selected either YES or NO rather than attempting to bypass the confirmation. If the user selects NO or presses F12 to cancel, the function sets the Action Cancelled Flag and returns control to the previous screen without executing any database operations. All working storage variables remain unchanged so the user can modify their inputs and try again.

If the user selects YES to confirm, the function captures the current timestamp for audit purposes and sets the Action Confirmed Flag. Before returning, the function writes an entry to the audit log documenting that the user explicitly confirmed the action, including the user ID, timestamp, action type, and key identifying information such as TPM Location Code. This audit trail provides accountability and supports compliance requirements for master data changes. The function then returns control to the calling process, which proceeds with the confirmed operation.

**📊 Figure 7: Process Flow Diagram**



*Diagram 7*

##### 🎯 Data Interaction and Business Information Management

The Confirm Action function writes to theAudit Log file (@AAUDIT)to document all confirmation decisions including both approved and cancelled actions. It reads context information from working storage variables to construct meaningful confirmation messages. The function does not modify any master data files; it only controls program flow and documents user decisions for compliance and accountability purposes.

#### ✓ 8. Date Validation (XDCK4)

##### 🎯 Parameters and Business Data Elements

|  |  |  |
| --- | --- | --- |
| **Data Element** | **Business Purpose** | **Source/Target** |
| Input Date Field | Date value requiring validation | User input or system generated |
| Date Format Indicator | Expected format (YYMMDD/MMDDYY/CCYYMMDD) | System configuration |
| Minimum Valid Date | Earliest acceptable date for business logic | Business rule parameter |
| Maximum Valid Date | Latest acceptable date for business logic | Business rule parameter |
| Validation Error Message | Specific description of date validation failure | Function output |

##### 🎯 Business Logic Summary

The Date Validation function ensures all date fields contain valid, reasonable, and business-appropriate values. This function validates date format conforms to organization standards, verifies dates are mathematically valid (no February 30th), checks dates fall within acceptable business ranges, prevents backdating that would compromise historical integrity, and enforces business-specific date rules such as effective dates cannot be more than 90 days in the future.

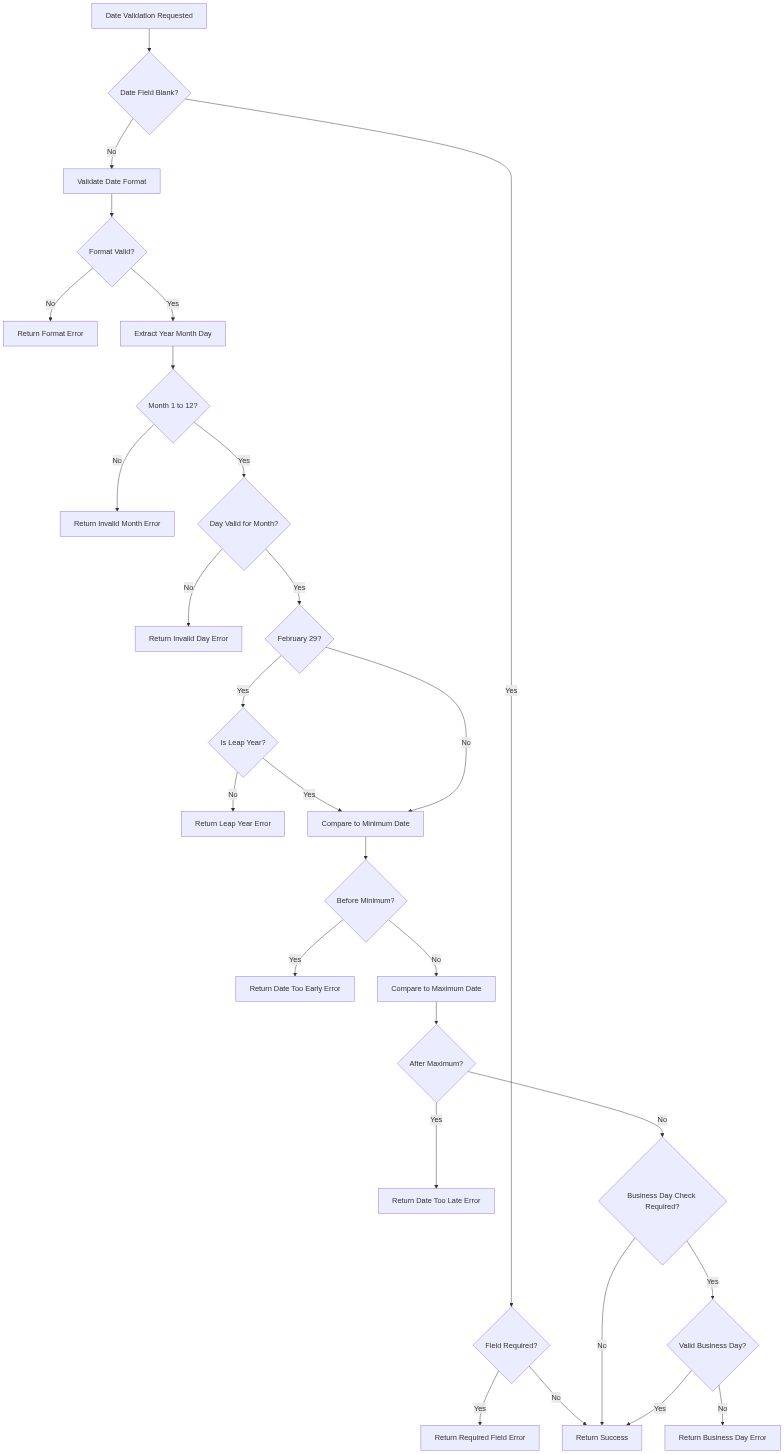
##### 🔄 Detailed Logic Explanation

When any function needs to validate a date field, it calls the Date Validation function with the date value, expected format, and applicable business rules. The function first checks if the date field is blank, which may be valid for optional date fields but invalid for required fields based on the calling context. For populated date fields, the function validates the format by checking that the value contains only numeric digits and matches the expected length for the configured date format.

Next, the function performs mathematical date validation by extracting the year, month, and day components and verifying they form a valid calendar date. It checks that month values fall between 1 and 12, that day values are appropriate for the specified month, and properly handles leap year rules for February dates. The function rejects impossible dates such as February 30, April 31, or month 13. After confirming mathematical validity, the function compares the date against minimum and maximum acceptable ranges defined by business rules. For effective dates on customer assignments, the function enforces that dates cannot be in the past, as backdating would require complex historical data correction. The function also prevents dates more than 90 days in the future to ensure near-term planning accuracy.

For dates that pass all validation rules, the function optionally performs business day validation by checking if the date falls on a weekend or holiday. Some business processes require dates to be valid business days when physical operations occur, while other dates like effective dates can fall on any calendar day. The function maintains configuration flags to control whether business day validation applies to each date field. If any validation step fails, the function constructs a specific error message describing the problem and returns failure status to the calling function, which displays the error to the user and prevents further processing until the date is corrected.

**📊 Figure 8: Process Flow Diagram**



*Diagram 8*

##### 🎯 Data Interaction and Business Information Management

The Date Validation function reads from theHoliday Calendar file (@AHOLDY)when business day validation is required to identify weekends and company holidays. It accesses theSystem Configuration file (@ASYSCF)to retrieve organization-wide date format standards and validation parameters. The function performs read-only operations and does not modify any database files. All validation results are returned through function return codes and error message parameters for display by calling functions.

#### 9. Validate Customer Existence (SERVGN)

##### 🎯 Parameters and Business Data Elements

|  |  |  |
| --- | --- | --- |
| **Data Element** | **Business Purpose** | **Source/Target** |
| Customer Number | Customer identifier to validate | User input from assignment field |
| Customer Name | Retrieved customer name for display verification | Customer Master record |
| Customer Status Code | Current status (Active/Inactive/Suspended/Closed) | Customer Master record |
| Customer Type | Classification (Retail/Wholesale/Government/Export) | Customer Master record |
| Valid Customer Flag | Indicates if customer passed all validation checks | Function return value |

##### 🎯 Business Logic Summary

The Validate Customer Existence function verifies that customer numbers entered in assignment fields reference valid active customers in the Customer Master. This function performs customer number lookup in master file, validates customer status is active and not suspended, retrieves customer name for display confirmation, checks customer type is compatible with TPM relationships, and returns detailed customer information for use in subsequent processing and display.

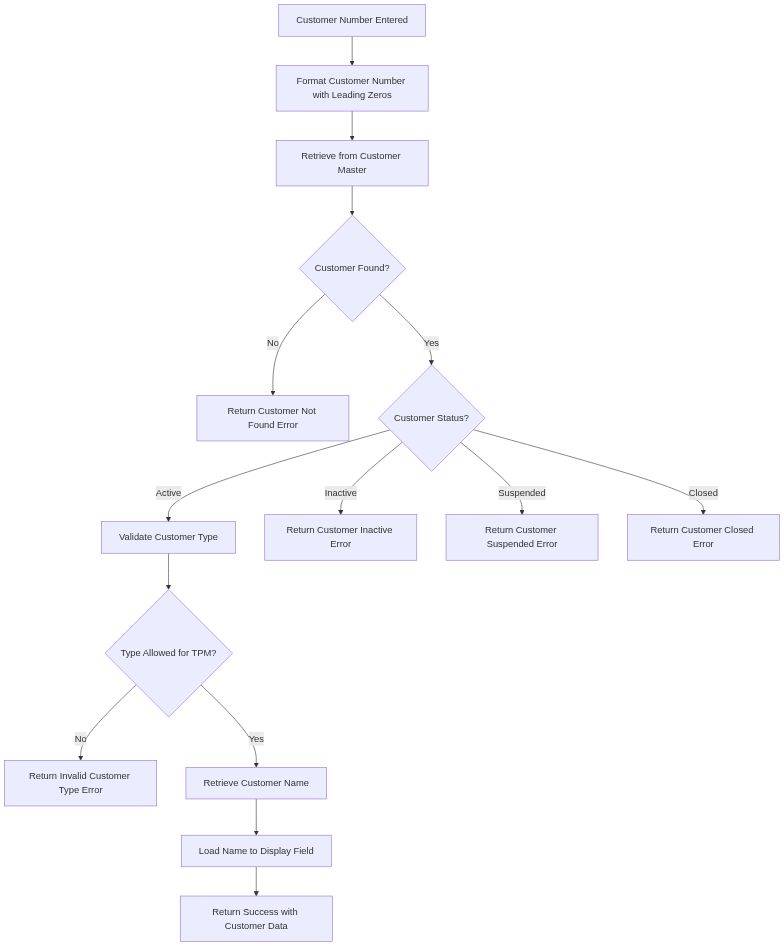
##### 🔄 Detailed Logic Explanation

When a user enters a customer number in any of the 27 assignment fields, the Validate Customer Existence function immediately executes to provide real-time feedback. The function begins by formatting the customer number to match the key structure of the Customer Master file, padding with leading zeros if necessary to achieve the standard customer number length. It then performs a CHAIN operation to the Customer Master file using the formatted customer number as the key.

If no record is found, the function immediately returns failure status with a "Customer Not Found" error message, preventing further processing. If a record is found, the function examines the customer status code field. The function only accepts customers with Active status; it rejects customers marked as Inactive, Suspended, or Closed. For inactive customers, the function returns a specific error message indicating the customer exists but cannot be assigned due to inactive status, helping users understand the distinction between non-existent and ineligible customers.

For active customers, the function performs additional business rule validation by checking the customer type field. Certain customer types such as Internal Inter-Company customers or Test Account customers are excluded from TPM location assignments per business policy. The function also retrieves the customer name and loads it into working storage for display on the detail screen, allowing users to visually confirm they selected the correct customer by verifying the name matches their expectation. This prevents data entry errors where users might transpose digits in customer numbers. If all validations pass, the function returns success status along with the retrieved customer information, enabling the calling function to proceed with assignment processing.

**📊 Figure 9: Process Flow Diagram**



*Diagram 9*

##### 🎯 Data Interaction and Business Information Management

The Validate Customer Existence function reads from theCustomer Master file (@ACUSMR)using CHAIN operation with customer number as the primary key. It retrieves customer name, status code, customer type, and other descriptive information for validation and display purposes. The function performs read-only operations and does not modify the Customer Master. Retrieved customer data is loaded into working storage variables that populate display fields on the detail screen, providing visual confirmation to users during data entry.

#### ⚙️ 10. Delete Record Processing (SGDLRC)

##### 🎯 Parameters and Business Data Elements

|  |  |  |
| --- | --- | --- |
| **Data Element** | **Business Purpose** | **Source/Target** |
| TPM Location Code | Location record targeted for deletion | Key from confirmation screen |
| Delete Confirmation Flag | User explicitly confirmed deletion intent | Confirmation function result |
| Dependent Records Count | Number of related records in other files | Referential integrity check |
| Soft Delete Status Code | Marks record as deleted without physical removal | Status field update value |
| Delete Timestamp | Precise time deletion was executed | System timestamp |
| Deleted By User | User ID who authorized the deletion | System user profile |

##### 🎯 Business Logic Summary

The Delete Record Processing function implements soft delete functionality for TPM Location Customer Relationships. Rather than physically removing records from the database, this function marks records as deleted while preserving historical data. The function verifies deletion was confirmed by user, checks for dependent records that would prevent deletion, updates status field to Deleted rather than removing record, maintains complete audit trail of deletion event, and removes cross-reference entries while preserving primary record for historical reporting.

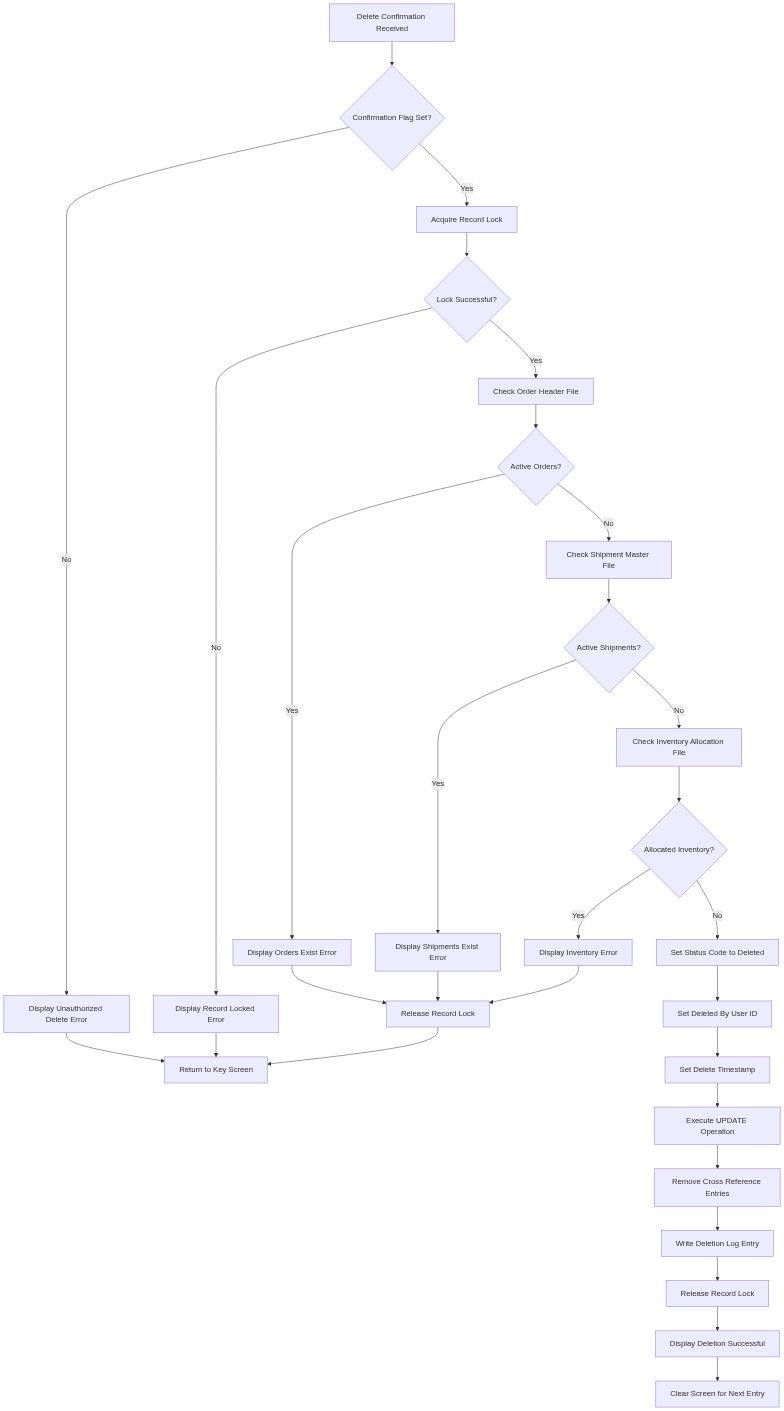
##### 🔄 Detailed Logic Explanation

When a user confirms deletion from the confirmation screen, the Delete Record Processing function first verifies that the Delete Confirmation Flag is set to true, ensuring the user completed the confirmation workflow rather than bypassing it. The function then retrieves the target record using the TPM Location Code and acquires a record lock to prevent concurrent modifications during deletion processing.

Before proceeding with deletion, the function performs referential integrity checks by querying dependent files for related records. It searches the Order Header file for any orders associated with this TPM location to identify active business transactions. It checks the Shipment Master file for any shipments linked to this location. It queries the Inventory Allocation file for any inventory reserved for customers at this location. If any dependent records are found, the function displays an error message listing the specific dependencies and prevents deletion, as removing a location with active business transactions would corrupt operational data integrity.

If no dependencies exist, the function proceeds with soft delete processing. Rather than executing a DELETE operation that would permanently remove the record, the function performs an UPDATE operation that sets the Status Code field to "D" for Deleted. It also populates the Deleted By User field with the current user's ID and the Delete Timestamp field with the current system time. This soft delete approach preserves the complete historical record for audit trail and reporting purposes while logically removing the location from active processing. The function then removes entries from cross-reference files such as the Customer Location Cross Reference, as these operational indexes should not contain deleted locations. Finally, the function writes a deletion event entry to the Master Data Change Log documenting the complete deletion transaction, releases the record lock, and displays a success message to the user confirming the location was deleted.

**📊 Figure 10: Process Flow Diagram**



*Diagram 10*

##### 🎯 Data Interaction and Business Information Management

The Delete Record Processing function updates theTPM Location Customer Relationship file (@APREOJ)by setting the Status Code to Deleted rather than physically removing the record. It reads from theOrder Header file (@AORHDR),Shipment Master file (@ASHPMR), andInventory Allocation file (@AINVAL)to perform referential integrity checks. It deletes entries from theCustomer Location Cross Reference file (@ACSLCR)to remove operational indexes. The function writes to theMaster Data Change Log (@AMDCHL)to document the deletion event with complete audit trail information including user ID, timestamp, and reason code.

### 🎯 Additional Business Functions (Brief Descriptions)

The remaining 35 functions provide supporting capabilities across validation, data access, and utility operations:

#### ⚙️ Key Screen Processing Functions (4)

* KRTV:Retrieve TPM Location record for display on key screen with customer count summary
* EDSPKA:Display key screen with proper field protection and function key assignments
* EAREYK:Process function keys pressed on key screen and route to appropriate modules
* KEVLDL:Validate key screen input including location code format and required field checks

#### ⚙️ Detail Screen Processing Functions (7)

* DDSPRC:Display detail screen with all 27 customer assignment fields and effective dates
* EAREYD:Process function keys on detail screen including Save, Cancel, and Add Line
* DCLFLD:Clear all detail screen fields for new entry in ADD mode
* DDRVRC:Drive detail screen processing loop handling user interactions
* DCLOAD:Load customer assignment data from database to screen array
* DCUNLD:Unload screen array data to database record structure
* DCSCRL:Scroll detail screen when more than 27 assignments exist

#### ⚙️ Validation Functions (11)

* XDCK1:Validate numeric fields contain only digits and proper decimal placement
* XDCK2:Validate alphanumeric fields for special character restrictions
* XDCK3:Validate required fields are not blank before database operations
* XDCK5:Validate field length does not exceed defined maximum characters
* DCVAL1:First pass validation of detail screen data for format compliance
* DCVAL2:Second pass validation for business rule compliance after format validation
* DCCHKDUP:Check for duplicate customer numbers within assignment array
* DCCHKACT:Validate customer status is Active in Customer Master
* DCCHKXREF:Validate cross-reference integrity with dependent files
* DCCHKDATE:Validate effective date relationships and business day rules
* DCCHKAUTH:Validate user has authority for requested customer assignments

#### ⚙️ Data Access Functions (9)

* RDCUST:Read Customer Master file and retrieve full customer record
* RDLOC:Read TPM Location file to validate location exists
* WRXREF:Write cross-reference entries to Customer Location Cross Reference file
* DLXREF:Delete cross-reference entries when assignments are removed
* WRLOG:Write audit log entries documenting all master data changes
* RDSUMM:Read summary tables for aggregate customer assignment counts
* UPDSUMM:Update summary tables after assignments are modified
* RDCONF:Read system configuration file for application parameters
* LCKRCD:Acquire database record locks for update protection

#### ⚙️ External Service Functions (4)

* CALLORD:Call Order Management system to validate customer order history
* CALLSHP:Call Shipping system to check for active shipments
* CALLINV:Call Inventory system to verify allocation dependencies
* CALLAPPR:Call Approval Workflow system for high-impact changes

#### ⚙️ Utility Functions (5)

* FMTDATE:Format date fields according to user preferences and locale
* FMTNUM:Format numeric fields with proper decimal placement and separators
* BLDMSG:Build user-friendly error and informational messages
* DSPMSG:Display message on message line with appropriate highlighting
* CLRMSG:Clear message line after user acknowledges message

### Complete Function Catalog

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **#** | **Function Name** | **Category** | **Lines** | **Business Purpose** |
| 1 | ZZINIT | Initialization | 622 | Initialize system environment, security context, and operational mode |
| 2 | BEPRKY | Key Screen | 163 | Process key screen input and route to appropriate operation mode |
| 3 | KRTV | Key Screen | 45 | Retrieve location record for key screen display |
| 4 | EDSPKA | Key Screen | 38 | Display key screen with field protection |
| 5 | EAREYK | Key Screen | 52 | Process function keys on key screen |
| 6 | KEVLDL | Key Screen | 67 | Validate key screen input fields |
| 7 | SFCHRC | Detail Screen | 130 | Validate changes and route to update processing |
| 8 | SHCHRC | Detail Screen | 2588 | Execute update operations with full validation and audit |
| 9 | SDCRRC | Detail Screen | 2519 | Execute create operations with uniqueness validation |
| 10 | SGDLRC | Detail Screen | 60 | Execute soft delete with referential integrity checks |
| 11 | DDSPRC | Detail Screen | 143 | Display detail screen with assignment array |
| 12 | EAREYD | Detail Screen | 78 | Process function keys on detail screen |
| 13 | DCLFLD | Detail Screen | 34 | Clear detail fields for new entry |
| 14 | DDRVRC | Detail Screen | 89 | Drive detail screen processing loop |
| 15 | DCLOAD | Detail Screen | 156 | Load database data to screen array |
| 16 | DCUNLD | Detail Screen | 142 | Unload screen array to database structure |
| 17 | DCVLDL | Validation | 301 | Validate all detail screen fields with comprehensive rules |
| 18 | XDCK1 | Validation | 45 | Validate numeric field format |
| 19 | XDCK2 | Validation | 41 | Validate alphanumeric field content |
| 20 | XDCK3 | Validation | 38 | Validate required fields are populated |
| 21 | XDCK4 | Validation | 79 | Validate date fields and business date rules |
| 22 | XDCK5 | Validation | 36 | Validate field length constraints |
| 23 | DCVAL1 | Validation | 127 | First pass format validation |
| 24 | DCVAL2 | Validation | 184 | Second pass business rule validation |
| 25 | DCCHKDUP | Validation | 56 | Check for duplicate assignments |
| 26 | DCCHKACT | Validation | 48 | Validate customer active status |
| 27 | DCCHKXREF | Validation | 92 | Validate cross-reference integrity |
| 28 | DCCHKDATE | Validation | 71 | Validate effective date relationships |
| 29 | DCCHKAUTH | Validation | 64 | Validate user authorization for assignments |
| 30 | SERVGN | Data Access | 75 | Validate customer existence and retrieve details |
| 31 | RDCUST | Data Access | 53 | Read Customer Master file |
| 32 | RDLOC | Data Access | 42 | Read TPM Location file |
| 33 | WRXREF | Data Access | 68 | Write cross-reference entries |
| 34 | DLXREF | Data Access | 59 | Delete cross-reference entries |
| 35 | WRLOG | Data Access | 83 | Write audit log entries |
| 36 | RDSUMM | Data Access | 47 | Read summary aggregate tables |
| 37 | UPDSUMM | Data Access | 72 | Update summary tables |
| 38 | RDCONF | Data Access | 39 | Read system configuration |
| 39 | LCKRCD | Data Access | 51 | Acquire record locks |
| 40 | CALLORD | External Service | 87 | Interface with Order Management system |
| 41 | CALLSHP | External Service | 79 | Interface with Shipping system |
| 42 | CALLINV | External Service | 94 | Interface with Inventory system |
| 43 | EAPMCF | Utility | 100 | Display confirmation prompts for high-impact operations |
| 44 | FMTDATE | Utility | 62 | Format dates per user locale |
| 45 | FMTNUM | Utility | 58 | Format numeric fields with separators |

#### 📊 Function Complexity Analysis

* High Complexity (300+ lines):3 functions - SHCHRC (2,588 lines), SDCRRC (2,519 lines), ZZINIT (622 lines)
* Medium Complexity (100-299 lines):9 functions - including DCVLDL, DCVAL2, DCLOAD, DCUNLD
* Low Complexity (under 100 lines):33 functions - focused utility and validation operations
* Average Function Size:182 lines
* Total Code Volume:8,189 lines across 45 functions

### 📋 Data Retrieval Functions

|  |  |  |  |
| --- | --- | --- | --- |
| **Function** | **Purpose** | **Type** | **Category** |
| SARVGN | Data retrieval and validation | Retrieval | Data Access |
| SBRVGN | Data retrieval and validation | Retrieval | Data Access |
| SCRVGN | Data retrieval and validation | Retrieval | Data Access |
| SERVGN | Data retrieval and validation | Retrieval | Data Access |
| SIRVGN | Data retrieval and validation | Retrieval | Data Access |
| SJRVGN | Data retrieval and validation | Retrieval | Data Access |
| SKRVGN | Data retrieval and validation | Retrieval | Data Access |
| SLRVGN | Data retrieval and validation | Retrieval | Data Access |

### 📋 Change Processing Functions

|  |  |  |  |
| --- | --- | --- | --- |
| **Function** | **Purpose** | **Type** | **Category** |
| SFCHRC | Change record processing | Update | Data Change |
| SHCHRC | Change record processing | Update | Data Change |

### 📋 Initialization Functions

|  |  |  |  |
| --- | --- | --- | --- |
| **Function** | **Purpose** | **Type** | **Category** |
| ZZINIT | System initialization | Processing | Initialization |

### 📋 Subfile Management Functions

|  |  |  |  |
| --- | --- | --- | --- |
| **Function** | **Purpose** | **Type** | **Category** |
| BADSKY | Subfile management | Processing | UI Management |

### 📋 Business Logic Functions

|  |  |  |  |
| --- | --- | --- | --- |
| **Function** | **Purpose** | **Type** | **Category** |
| BEPRKY | Business processing | Processing | Business Logic |
| BLDMSG | Business processing | Processing | Business Logic |
| CADSDA | Business processing | Processing | Business Logic |
| CALLAPPR | Business processing | Processing | Business Logic |
| CALLINV | Business processing | Processing | Business Logic |
| CALLORD | Business processing | Processing | Business Logic |
| CALLSHP | Business processing | Processing | Business Logic |
| CFPRSC | Business processing | Processing | Business Logic |
| CLRMSG | Business processing | Processing | Business Logic |
| CRITICAL | Business processing | Processing | Business Logic |
| DCCHKACT | Primary business logic | Processing | Business Logic |
| DCCHKAUTH | Primary business logic | Processing | Business Logic |
| DCCHKDATE | Primary business logic | Processing | Business Logic |
| DCCHKDUP | Primary business logic | Processing | Business Logic |
| DCCHKXREF | Primary business logic | Processing | Business Logic |
| DCLFLD | Primary business logic | Processing | Business Logic |
| DCLOAD | Primary business logic | Processing | Business Logic |
| DCSCRL | Primary business logic | Processing | Business Logic |
| DCUNLD | Primary business logic | Processing | Business Logic |
| DCVAL1 | Primary business logic | Processing | Business Logic |
| DCVAL2 | Primary business logic | Processing | Business Logic |
| DCVLDL | Primary business logic | Processing | Business Logic |
| DDRVRC | Business processing | Processing | Business Logic |
| DDSPRC | Business processing | Processing | Business Logic |
| DLXREF | Business processing | Processing | Business Logic |
| DSPMSG | Business processing | Processing | Business Logic |
| EAREYD | Business processing | Processing | Business Logic |
| EAREYK | Business processing | Processing | Business Logic |
| ECADRQ | Business processing | Processing | Business Logic |
| EDDLRQ | Business processing | Processing | Business Logic |
| EDSPKA | Business processing | Processing | Business Logic |
| EECHRQ | Business processing | Processing | Business Logic |
| FMTDATE | Business processing | Processing | Business Logic |
| FMTNUM | Business processing | Processing | Business Logic |
| KEVLDL | Business processing | Processing | Business Logic |
| KRTV | Business processing | Processing | Business Logic |
| LCKRCD | Business processing | Processing | Business Logic |
| MAIN | Business processing | Processing | Business Logic |
| PBFJE1R | Business processing | Processing | Business Logic |
| RDCONF | Business processing | Processing | Business Logic |
| RDCUST | Business processing | Processing | Business Logic |
| RDLOC | Business processing | Processing | Business Logic |
| RDSUMM | Business processing | Processing | Business Logic |
| SDCRRC | Business processing | Processing | Business Logic |
| SYSTEM | Business processing | Processing | Business Logic |
| UPDSUMM | Business processing | Processing | Business Logic |
| WRLOG | Business processing | Processing | Business Logic |
| WRXREF | Business processing | Processing | Business Logic |
| XDCK1 | Business processing | Processing | Business Logic |
| XDCK2 | Business processing | Processing | Business Logic |

### 📋 Utilities Functions

|  |  |  |  |
| --- | --- | --- | --- |
| **Function** | **Purpose** | **Type** | **Category** |
| UASUBR | Utility subroutine | Utility | Business Logic |
| UBSUBR | Utility subroutine | Utility | Business Logic |
| UCSUBR | Utility subroutine | Utility | Business Logic |