The file containing the `log_module_message` procedure appears to be a PL/SQL script designed for logging events or messages within an Oracle database system. Here is a high-level summary of what this file does:

1. **Purpose**: The primary function is to log messages related to a software module's execution. These messages can include various details, such as the application suite, application, business entity, phase of execution, and errors if any.
2. **Parameters Handling**: The procedure accepts a range of parameters that specify details about the message to be logged, including severity levels (such as notifications, warnings, and errors) and identifiers for business processes.
3. **Validations**: The procedure includes validation checks for parameters to ensure they meet certain criteria (e.g., character lengths, valid phase names).
4. **Independence**: It uses the `AUTONOMOUS_TRANSACTION` pragma, meaning it operates independently of any other transactions that might be occurring, preventing it from interfering with other database operations.
5. **Debugging Support**: The procedure has the capability to handle debug messages, depending on certain conditions.
6. **Database Interaction**: It inserts messages into a specific database table (`xxmx_module_messages`), essentially writing logs to the database.
7. **Committing Changes**: After logging the message, it commits the transaction, ensuring that the log is saved.
8. **Error Handling**: The script can handle errors specifically related to the module as well as other unexpected exceptions, logging them appropriately and providing feedback for debugging purposes.
9. **Usage Scenario**: An example usage scenario is provided to demonstrate how the procedure might be called within another PL/SQL block or a stored procedure to log progress or issues during operations like payroll calculations.

Overall, the file serves as an error and event logging utility for an Oracle database, particularly focusing on supporting database operations by providing a means of tracking execution phases, monitoring progress, and recording errors for system administration and debugging efforts.

## log_module_message

Technical Documentation for `log_module_message` Procedure

The `log_module_message` procedure is a PL/SQL block that logs messages related to various phases of a software module's execution in an Oracle database. The messages contain information about the application suite, application, business entity, sub-entity, migration set ID, and more. Below, the procedure is documented in detail, explaining its structure, logic, and usage.

Procedure Definition

```
PROCEDURE log_module_message
...
END log_module_message;
```

The `log_module_message` is a standalone stored procedure that accepts multiple parameters to record detailed module execution messages. It supports a default value for `pt_i_FileSetID` and defines data types according to the columns of the `xxmx_module_messages` table.

Parameters

The procedure accepts the following parameters:

- `pt_i_ApplicationSuite`: The application suite name (must be 4 characters or less).
- `pt_i_Application`: The application name (must be 4 characters or less).
- `pt_i_BusinessEntity`: The business entity name involved.
- `pt_i_SubEntity`: The sub-entity name involved.
- `pt_i_FileSetID`: The file set identifier with a default value of 0.
- `pt_i_MigrationSetID`: The migration set identifier.
- `pt_i_Phase`: The phase of the module (`CORE`, `EXTRACT`, `TRANSFORM`, `EXPORT`, `VALIDATE`).
- `pt_i_Severity`: The severity of the message (`NOTIFICATION`, `WARNING`, `ERROR`).
- `pt_i_PackageName`: The package name where the procedure is contained.
- `pt_i_ProcOrFuncName`: The name of the procedure or function being executed.
- `pt_i_ProgressIndicator`: An indicator of the progress.
- `pt_i_ModuleMessage`: The detailed message of the module.
- `pt_i_OracleError`: Any Oracle error encountered.

Local Declarations

Constants

- `ct_ProcOrFuncName`: The constant name of the procedure for internal use.

Variables

- `v_debug_message`: A temporary variable to store the debug message indicator.

Exceptions

- `e_ModuleError`: A custom exception used to handle specific module errors.

Logic

Autonomous Transaction Pragma

The pragma `AUTONOMOUS_TRANSACTION` ensures that the transaction within the procedure is independent of any calling transaction.

Validations

The procedure performs several validations on input parameters to ensure their lengths and values meet expected criteria. If not, it raises the custom `e_ModuleError` exception.

```
IF LENGTH(pt_i_ApplicationSuite) > 4 THEN
    ...
END IF;
...
IF UPPER(pt_i_Phase) NOT IN ('CORE', 'EXTRACT', 'TRANSFORM', 'EXPORT','VALIDATE') THEN
    ...
END IF;
```

Debug Message Handling

The procedure retrieves a debug message indicator from another package function `xxmx_utilities_pkg.get_single_parameter_value`. Depending on whether this indicator is `Y` and the severity is `NOTIFICATION`, different handling is executed.

Insertion Logic

The procedure contains logic to insert a new record into the `xxmx_module_messages` table with all necessary data populated from the input parameters. If the severity is `WARNING` or `ERROR`, or if the debug message is not `Y`, the record is inserted without additional conditional checks.

```
INSERT INTO xxmx_module_messages
...
VALUES
(
    ...
);
```

Commit Transaction

The procedure commits the transaction, thereby saving the new record to the database.

Exception Handling

There are two exception handlers:

- `e_ModuleError`: Handles module-specific errors by logging them and raising an application error with a message referencing the `XXMX_MODULE_MESSAGES` table.
- `OTHERS`: Catches any other exceptions that occur, logs them, and also raises an application error with a reference to the `XXMX_MODULE_MESSAGES` table.

Example Usage

The procedure would typically be used within a PL/SQL block or another stored procedure to log messages about module execution, like so:

```
BEGIN
    log_module_message(
        pt_i_ApplicationSuite => 'HRMS',
        pt_i_Application => 'PAY',
        pt_i_BusinessEntity => 'PAYROLL',
        pt_i_SubEntity => 'EMP',
        pt_i_MigrationSetID => 123,
        pt_i_Phase => 'TRANSFORM',
        pt_i_Severity => 'NOTIFICATION',
        pt_i_PackageName => 'payroll_pkg',
        pt_i_ProcOrFuncName => 'calculate_salary',
        pt_i_ProgressIndicator => '50%',
        pt_i_ModuleMessage => 'Halfway through salary calculation.',
        pt_i_OracleError => NULL
    );
END;
```

Summary

The `log_module_message` procedure is a comprehensive tool for recording detailed messages pertaining to module operations within an Oracle database. It has checks in place to validate the input parameters and will log any errors encountered during its execution, ensuring that there is a robust audit trail available for system administrators and developers to troubleshoot and understand the flow of the system.