UDC 681.3

A. S. BORANBAYEV

ERROR HANDLING METHODOLOGY WHILE DEVELOPING MODERN INFORMATION SYSTEMS USING SERVICE-ORIENTED ARCHITECTURE

(Represented by academician of NAS of the RK M.O. Otelbayev)

This article is intended to give a description of the Java Error Handler component of the developed application. The project involved the development of an Information System using Service-Oriented Architecture.

INTRODUCTION AND OVERVIEW

This article is intended to give a detailed description of the Java Error Handler component of the developed application. It also documents the discussion decisions, design risks, issues, assumptions and other information related to design.

The developed Information System is online application software. Web-based access to enterprise software applications becomes an important part of many organizations. Intensive technological change in hardware and networking software [1] provides more choices than in the past [2]. It takes care of activities related to the every day functioning of an organization, including the back-end administrative processes of general administration and finance. The administrative processes have been automated to ease the tasks of administration and data keeping, thereby bringing in operational efficiency throughout the organization [3]. The Information System has been developed using Service-Oriented Architecture (SOA) [4].

In the world of component oriented programming, to address the common requirements of Exception handling and logging, developers often use reusable tools like Log4J, etc. [5] Sometimes software developers even try to extend these tools when required. However these tools do not always cater error handling needs in a service oriented environment.

Purpose of the component

Regardless of the caller, for a given error code, the developed Error Handler component fetches corresponding error details from reference tables and returns the same to the caller as a Java Error Object. If the caller invokes the Error Handler with list of values along with the error code, then this components uses the "list of values" to construct a dynamic error message, which is again part of Java Error Object that will be sent back to the caller.

Context

Any software application should properly handle the error that it encounters, so that it can be reliably used by any end-user whether it is either human or other software [6]. Also it is the software application's responsibility to properly identify the reason for error that it encountered and propagate the same to the end user, so that appropriate action can be taken on the error.

In the developed project, all the Java components are exposed as SOA Services and it is the responsibility of services to send back proper error details to the caller [7]. Since the project's Java services are also called by COBOL programs, it would be more meaningful if we can send back an error object in the response instead of simply throwing a Business or Technical exception.

Whenever any Java service in the application encounters an error, whether its Business or Technical error, it calls the Error Handler with appropriate technical error code, to retrieve the error details for the same. After getting the details and depending upon the severity of the error the service either continues processing or aborts it's processing and sends back the error details to its caller that can be a COBOL program or other Java component.

In another context, Service Manager which is a project's service choreographer calls the Error Handler to sort the list of error objects based on the error severity. This is required for Service Manager to take decision on either continues calling next

service or abort processing and send back the error details to the COBOL via MQ Access Service Layer.

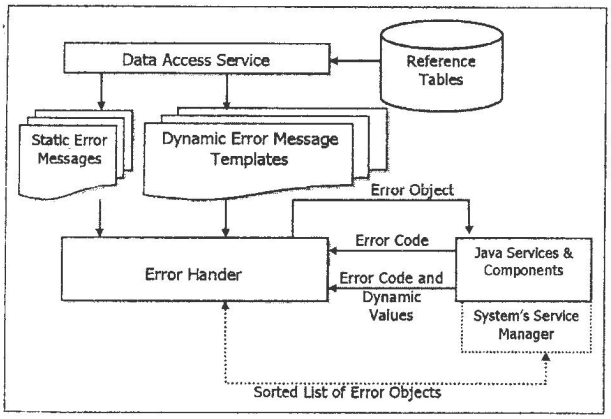

Figure 1: Context diagram

Scope

Following is the scope of Error Handler component:

1. Given a technical error code, Error Handler returns error details to the caller as Java Error Object, which is fetched using system's Data Access Object (DAO) from Reference tables.

2. If the caller invokes Error Handler with values and error code then:

a. Get the error object as mentioned in Point 1.

b. Construct a dynamic technical or business (based on the called API) error message using the incoming values and the dynamic error message template. If the dynamic error message template does not have any place holders for the values, then the dynamic error message template will be treated as error message.

c. Attach the created error message in Point 2.b to the error object before returning the same to the caller.

3. Provide an API in Error Handler that takes a list of Error Objects and returns back the list with Error Objects sorted, based on severity of error.

4. Whenever an error occurs inside Error Handler it creates an Error Object for "Severe Technical Error" and returns the same to the caller. In this scenario the caller will not receive the error details for error code it provided to the Error Handler, instead it will receive an error object with error details that happened inside Error Handler.

REQUIREMENTS AND DESIGN

Error Handler is used by all Services of the system that implement various Business Processes of the system. Hence Error Handler is indirectly associated with almost all the business process.

Functional Requirements

This is a logical component; hence it does not have any use case associated with it. But it is expected to have following capabilities.

• Based on the given error code it should able to fetch error details from the database, construct an Error Object and return the same to the caller.

• In addition to the error code if the caller provides values for dynamic error message construction, then the component should fetch the error details and construct dynamic error message using the provided details before sending the Java Error Object to the caller.

• Given a List of Error Object and attribute name, the "Error Handler" should sort the Error Objects based on the given attribute name.

Dependencies & Design decisions

Internal Dependency	Impact if not met	Desired Outcome
Error DAO: Data access service layer to retrieve data from reference database that contains error details for a given error code. This component's availability is very crucial to Error Handler component.	If Error Handler DAO is not ready on time, it will have impact on how the data will be retrieved from Database.	Error Handler can provide details for a given error code.

External Dependency	Impact if not met	Desired Outcome
Reference Tables	Error Handler will not be able to retrieve error details for a given error code from Reference Tables.	Database available for integration testing.

Decision Description	Impact	Comments
If an error occurs inside "Error handler" then it creates and returns an Error Object for "Severe Technical Error" to the caller.	The caller may not receive the "Error Object" for the "Error Code" it provided.	The caller need to check the severity of returned Error object and based on the same it should either continue or abort processing.

In the message template for dynamic message creation, the variables are in the form of {0}, {1} etc. For e.g. the dynamic message template might look like "{0} members found with first and last name {1}, {2}."

Design Overview

Though the Error Handler does not actually "handle" the error for the calling services/component, it helps the caller to fetch the error details from database using the Error DAO. Also the Error Handler does not make updates using Error DAO.

After retrieving the error details from Error DAO, it clones the Error Object and uses the cloned object to construct the dynamic error message (if required), then returns the same to the caller. Since DAO caches the error objects, we need to make sure that the Error Object we received from Error DAO is not modified. Hence we clone the Error Object i.e. Create a new Error Object and copy all the contents to it.

Apart from the above mentioned activity the Error Handler does not do any functional processing with that object. It is up to the caller to appropriately handle the returned object. Also Error Handler is a Stateless and Java Singleton class [10] (i.e. only one "Error handler" instance will exist per Java Virtual Machine (JVM)).

Error Handler also provides an Application Programming Interface (API) to sort a list of error objects based on the severity of the error. Following are the Sub-components of Error Handler and its functionalities.

ErrorHandler: It is a Java interface, which defines methods to get error details and sort message objects. The implementation of defined methods will be done in the implementation class **ErrorHandlerImpl**.

Brief description of methods defined in ***ErrorHandler*** interface is as follows:

· ***getErrorObject(errorCode : String):*** This method takes an error code, fetches the error details as Error Object from Reference Tables using Error DAO.

· ***getErrorObject(errorCode : String, valueList : String):*** This is an overloaded method, which almost does the same functionality as ***getErrorObject(errorCode : String)***, but in addition, it uses the *"valueList"* as input to construct an dynamic error message. This API internally uses the Java's MessageFormat API to format the error message.

· ***sortErrorObjectsOnSeverity(errorObjects : List):*** This method takes a list of ErrorObjects and sorts it based on Error Object's severity.

· ***getErrorObject(inputError : Error):*** This method takes the errorCode and errorMessageArguments from the given Error object, and returns a new Error object with error messages created accordingly.

· ***getErrorObject(errorCollection : ErrorCollection):*** This method takes the errorCode and errorMessageArguments for each Error object from the given ErrorCollection, and returns a new ErrorCollection with new Error objects having error messages created accordingly.

· ***getErrorObject(errorCode : String, messageArguments : Object[]):*** For a given error code and array of messageArguments, this method fetches the Error object and using the array of messageArguments it fills in the error template for business error message and returns the updated Error object to the caller.

· ***getErrorObject(errorCode : String, businessMessageArguments : Object[], technicalMessageArguments : Object[]):*** Takes the errorCode , businessMessageArguments and technicalMessageArguments as input, and returns a Error object having error messages created accordingly. *getErrorObject(ErrorCode : String), getErrorObject(ErrorCode : String, ValueList : String)* and *getErrorObject(String errorCode, Object[] messageArguments)* in this class indirectly call this method.

ErrorHandlerImpl: Java Class, which implements ErrorHandler interface and the corresponding interface methods.

Frequency of Execution

Type: Interactive or Batch	Execution Schedule Transactional/Daily/Weekly/Monthly/Quarterly/Annual	Execution Frequency
Both	On Demand	In the event of any error that is occurring inside the scope of Java services of the system.

Interfaces

Interface	Input/Output
Error Handler	Input: Error code. Output: Error object, with the following details: • Component name from where the error originated. • Static or Dynamic Error Message given by Business team. • Entity Type (Possible values are Technical, Member). • Error Priority/Severity. • Equivalent Error code assigned by Business team. • Static or Dynamic Error Message assigned by Technical team.
Error DAO	Input: Error code. Output: Error object, with the following details: • Component name from where the error originated. • Static or Template for Dynamic Error Message given by Business team. • Entity Type (Possible values are Technical, Member). • Error Priority/Severity. • Equivalent Error code assigned by Business team. • Static or Dynamic Error Message assigned by Technical team.

Examples of the created Java Program Logic

Program Name	ErrorHandlerImpl
Online / Batch	Online & Batch
Procedure Name	getErrorObject
Input Parameters	errorCode : String
Return Value	Error Object (With the details of Error fetched from database)
Procedure Description	This method internally calls the "getErrorObject(String errorCode, Object[] messageArguments)", which is an overloaded method and simply returns the value whatever it receives.
Pseudo Code Logic	
Calls the overloaded getErrorObject(String errorCode, Object[] messageArguments) method using the incoming ErrorCode and null for the messageArguments.	
Error Handling	*Catches any Exception and returns a Error Object for "Severe Technical Error"*

Program Name	ErrorHandlerImpl
Online / Batch	Online & Batch
Procedure Name	sortErrorObjectsOnSeverity
Input Parameters	List of Error Objects
Return Value	Sorted error object list
Procedure Description	This method returns the sorted list of error objects based on the severity of the Error Object
Pseudo Code Logic	
Step 1 If (error object list = NULL or empty) Return null list Step 2 Construct a comparator interface implementation for sorting Error Objects based on severity Call sort method in Java Collections class and pass the incoming list and the created comparator Step 3 Call the collection class's sort method by passing the incoming error object list and the created comparator Step 4 Return the sorted error object list	
Error Handling	• *No Business or Technical exceptions* • *System Exceptions will be caught and original list will be sent back without sorting.*

Error Handling

Anticipated Error	Error Code	Error Message
SQL Exceptions As application using Spring Framework [9] which throws DataAccessException instead of SQL Exception; we are using EepSQLException which is a customized class for handling SQL exceptions.	1115	Error Message is dynamically generated against the executed query with root cause and error message.
All System Exceptions (All System Exceptions are mapped to Single error code with error severe technical error message)	10002	An internal error is preventing error handler to fetch the error details.
Error DAO returns empty result.	10003	Error code is invalid.

CONCLUSION

This paper described the challenges associated with error handling and logging when we develop Information Systems in a Service oriented world. Error Handler is one of basic and essential components of the developed Information System, which is called by almost all the Java services and components on error condition that are all in the scope of the Information System. Hence this component has been performance tuned the extent, so that the caller does not usually spend considerable amount of time in this component.

Error handling involves identification of error, determining whether the program needs to continue or abort based on the error severity, executing appropriate error handlers, retrying the action, and

clean up [6]. Error logging involves persisting the error information and associated context information to a data store for trouble shooting later [5]. Error handling is often used as the basis of error recovery function in service-oriented architecture. Software developers all over the world are working with web services, implemented by use of such popular development kits as: IBM WebSphere SDK for Web Services (WSDK) [11] and JAX-RPC implementation at Sun Microsystems.

Over the last few years, Service-Oriented Architecture (SOA) has received a lot of attention, because it brought a lot of changes to software development [7]. However, even with SOA we still need to use effective software engineering practices, because poorly managed SOA development can also go wrong, just like any other architectural approach [7] [8]. This article has showed a practical view to developing Information Systems, from both the technology and business perspectives. It presented a case study drawn from real-world experience, and showed the benefits that can be achieved through a successful software implementation.

REFERENCES

1. *Hritonenko N., Yatsenko Yu.* Creative destruction of computing systems: Analysis and modeling, Journal of Supercomputing, 38(2006), 143-154.

2. *Yatsenko Yu., Hritonenko N.* Network economics and optimal replacement of age-structured IT capital, Mathematical Methods of Operations Research, 65(2007), 483-497.

3. *Boranbayev A.S.* Defining methodologies for developing J2EE web-based information systems, Nonlinear Analysis (2009), doi: 10.1016/j.na.2009.02.002

4. Web Services and Service-Oriented Architectures. Retrieved May 11, 2009 from: http://www.service-architecture.com/index.html

5. Apache Logging Services. Apache log4j. Retrieved May 12, 2009 from: http://logging.apache.org/log4j/index.html

6. *Boranbayev A. S.* (April 23-26, 2009). DESIGNING AND DEVELOPING EXCEPTION HANDLING IN THE MODERN CUSTOM WEB APPLICATIONS. The IABPAD Conference Proceedings, THE INTERNATIONAL ACADEMY OF BUSINESS AND PUBLIC ADMINISTRATION DISCIPLINES (IABPAD) CONFERENCE; Dallas, Texas; Volume 6, Number 2, 1217-1226.

7. *Boranbayev A.S.* Optimal Methods for Java Web Services, News of the National Academy of Science of the Republic of Kazakhstan, №5, 2007, 38-43.

8. *Boranbayev A.S.* Reference Architecture for Web Applications, Reports of the National Academy of Science of the Republic of Kazakhstan, 2007, №5, 18-26.

9. The Spring Framework official website. Retrieved April 28, 2009 from http://www.springframework.org/

10. Design Patterns. Singleton Design Pattern. Retrieved May 4, 2009 from http://www.patterns.24bytes.com/2009/04/singleton-design-pattern.html

11. IBM WebSphere SDK for Web Services (WSDK) Version 5.1. Retrieved June 8, 2007 from http://www.ibm.com/developerworks/library/ws-wsdk51intro/

Резюме

Жасалған ақпараттық жүйедегі қателерді және шығаруларды өңдеу үшін арнайы істелінген Java Error Handler компонентінің жұмысы көрсетілген. Жасалған ақпараттық жүйе сервис-ориентир архитектурасына негізделген.

Резюме

Описывается разработанный Java Error Handler компонент для обработки ошибок и исключений в созданной информационной системе. Разработанная информационная система имеет сервис-ориентированную архитектуру.

Казахский университет технологии и бизнеса

Поступила 18.05.09 г.