Iterative Development: The use of J2EE Best Practices

Prepared By Owen Taylor
of The Middleware Company
Table of Contents

Overview ........................................................................................................................................3
Selecting Appropriate Tools .........................................................................................................3
  Scope ........................................................................................................................................3
  Non-Intrusiveness ....................................................................................................................3
  Ease of use .............................................................................................................................3
  Effectiveness ..........................................................................................................................4
Performance Related Best Practices ............................................................................................4
  Data-caching ...........................................................................................................................4
  Session Façade .......................................................................................................................5
  Service-locator .........................................................................................................................7
Identifying the Need for Best Practices ......................................................................................9
  In Development .....................................................................................................................9
  In Q/A .....................................................................................................................................9
  In Production .......................................................................................................................9
Using Precise/Indepth for the J2EE Platform .............................................................................9
  Instrumenting the code ..........................................................................................................9
  Running SmarTune™ .............................................................................................................10
  Drilling down .......................................................................................................................10
  Identifying the most prominent problem and getting advice ..............................................11
  Aligning the problems with the lack of Best Practices .........................................................12
Conclusion ..................................................................................................................................13
Appendix: Precise Indepth for J2EE In a Nutshell ......................................................................13
About Precise Software Solutions ............................................................................................15

Copyright © 2002. Precise Software Solutions, Ltd. All Rights Reserved. Precise/Interpoint and Precise/Insight are trademarks of Precise Software Solutions, Ltd. The Precise logo, Precise i³, Precise/Savant, Precise/Inform, Precise/Lightpoint, Precise/Sharpoint, Precise/Crosspoint, Precise/Savvy and Performance Warehouse are trademarks of Precise Software Solutions, Ltd. DB2 is a registered trademark and DB2 Universal Database is a trademark of IBM Corporation. Oracle and the Oracle logo are registered trademarks of Oracle Corporation. Java and J2EE are trademarks of Sun Microsystems. All other trademarks used herein are the property of their respective owners.
Overview

Selecting J2EE as a platform for development should increase the chances of success in the enterprise. Successful companies establish the use of best practices, patterns, and tools and spread the awareness of these amongst their J2EE programmers and architects. Successful developers share their knowledge and pass on their wisdom to others. The net result is improved productivity and the implementation of solid applications that are simple to use.

The challenges we all face today are two-fold:

- Obtaining the skills (or those with them) to build effective enterprise systems
- Meeting market-driven time lines while maintaining quality in our implementations

Experienced architects often make trade-offs to meet deadlines. These trade-offs include using re-factoring and other techniques to cut down development time and deal with the inherent (inherited) performance considerations. Application behavior is known to change from development to production and throughout the application lifecycle. As performance and stability issues are discovered, the application must be improved or abandoned. Making the most of these iterations (so that valuable time and money is not wasted) must be our goal. As a result, selecting the right tools and best practices is pivotal to successful application roll-outs.

Selecting Appropriate Tools

Scope

Helpful tools provide analysis of the entire application. One such set of tools is provided by Precise Software Solutions, who offer a complete product line dedicated to providing application performance management from the desktop, across the network, through the Application Server, to the database and the storage device. These products are powerful enough to grow with the companies business needs and provide a basis to achieve a true end-to-end application performance view and meet the evaluation criteria.

Non-Intrusiveness

Ideal tools provide accurate, useable data without altering:

- Source code
- Application Performance

The Precise/Indepth for J2EE method chain instrumentation uses native "C" code for the most time-critical operations and is significantly lighter weight than Java-based instrumentation. When used correctly, the performance overhead associated with using Precise/Indepth for J2EE is between 1% and 4%.

Ease of use

The best tools are easy to learn and use.

Precise/Indepth for J2EE utilizes a browser based User Interface facilitating ease of use, minimal training and ready access by any authorized user via their favorite browser. Important information is shared among multiple users simultaneously, or easily referenced by URL for later examination.
Effectiveness

Powerful tools are those that empower users to make improvements.

Precise/Indepth for J2EE provides SmarTune technology that pinpoints existing application weaknesses and provides reliable “best practices” suggestions for improving them.

Performance Related Best Practices

Data-caching

Much of the interesting data in a system can remain unchanged for minutes, hours or even longer. When accessing interesting information of such a static nature, or in non-crucial use-cases wherein a client does not require real-time data, network traffic and database usage can be greatly reduced through the use of data caching.

Figure 1 - No data caching. 9 invocations, 9 database hits

The clients in Figure 1 are running in the same VM and therefore we are not seeing excessive network traffic, the amount of database hits however, is excessive.
Session Façade

Architects using EJB technologies discovered almost immediately that providing access to Entity Beans from the client layer presented multiple problems:

- Overabundance of network traffic and latency
- Awkward Security management
- Inefficient transactional behavior
- Limits in reusability

Figure 2 - Using Data-caching. 9 invocations, 4 database hits
By interjecting a layer of indirection in the form of a Session Bean, these problems were easily solved (refer to Figure 4):

- The Session Beans become responsible for determining user permissions providing greater flexibility and reuse.
- Session Beans perform collections of calls to the Entity Beans on behalf of the remote clients: reducing network traffic.
- Transactional scope can be applied to methods that call groupings of Entity Beans, thus reducing the transactional overhead.
- Client code can be simplified due to more business logic being executed on the server.
When accessing common services within J2EE such as JMS factories, DataSources, EJBHome objects, etc, the application will use the JNDI (Java Naming and Directory Interface) API. This will require the creation of an initial context object (the root of the naming service), followed by a lookup or search for the desired resource or service. This lookup (if successful) results in the transferring of that resource's serializable representative to the interested party. Some major design problems become evident when the clients are co-located:

- Entire seconds can go by each time these operations are carried out
- Each interested client must be complicated by the inclusion of JNDI-API-specific code
- Unnecessary network traffic is added to the system
The addition of a service-locator object or cache to our system can greatly increase responsiveness of the system as a whole when many co-located clients (Servlets for example) require the same services:

---

**Figure 5 - Clients accessing JNDI services directly**

**Figure 6 - Using the Service-locator to cache references to commonly accessed services**
Identifying the Need for Best Practices

In Development

The question should not be, "is there a best practice?" but rather, "will this best practice improve our application?" The three best practices described in this document are an excellent starting point. Caching of frequently used data and references will benefit any J2EE application however, determining what "frequently used" actually is, can be difficult in development. The use of the Session Façade is a requirement for successful EJB development, but decisions such as transactional behavior and scope still need to be made. By testing early releases of an application, developers can locate and correct inappropriate design decisions.

Precise/Indepth for J2EE offers a feature called SmarTune™ that will recommend the use of particular best practices once it has been directed to examine specific aspects of an applications' behavior, but it is not an early development tool: Its' value is apparent once the application can be run.

In Q/A

In Quality Assurance it is often assumed that best practices and appropriate patterns have already been applied. The Q/A team if different from the development team is unlikely to have the expertise to identify the need for best practices.

The Precise product can offer invaluable assistance to those both familiar and unfamiliar with the application under scrutiny. If the application is deemed to perform poorly or fail it will be thrown back over the wall to the development team. Precise Indepth for J2EE is extremely useful in determining the application hotspots and suggesting the primary candidates for refactoring. Armed with real knowledge, the developers can confidently get to work and improve their applications.

In Production

Applying new design patterns and best practices to a production system is more than difficult: both technically and politically. It can however, be essential to creating a mature and ultimately successful application. Interrupting service to clients in order to re-deploy a production application will commonly happen if the application fails or is unusable. What Precise enables is the uncommon ability to identify when performance is under-par and justify expending additional resources to re-factoring by specifying the nature of the problem.

Using Precise/Indepth for the J2EE Platform

Instrumenting the code

Before you can learn anything about your application from the Precise product you must instrument (see appendix) your code. This demands that you:

1. Install the precise product (~30 minutes)
2. Alter the start script for the application server (~2 minutes)
Running SmarTune™

This feature of the Precise/Indepth product acts as an adviser to the user, applying rules to the gathered data and pointing out the most likely contributors to application slowdown. (SmarTune also offers general advice in how to increase the applications’ performance.) To use SmarTune you:

1. Start Precise/Indepth for J2EE
2. Run the application server and use/load-test the application
3. Open a web browser
4. View the Indepth GUI
5. Select the slice of sample data which holds the highest spike of activity and click the SmarTune link

SmarTune takes a few seconds and then reveals information such as this:

![Figure 7 - SmarTune Discovers Problems](image)

The numbers 70.46 46.41 and 28.18 refer to the percentage of time taken by these method invocations within the 30 second period analyzed.

Drilling down

In order to increase the applications performance, further analysis is necessary. The largest contributor is selected: "Click here to see the details!" and further information is shown:
Identifying the most prominent problem and getting advice

Selecting the provided hyperlink associated with:
shop.PriceListSessionBean_o8g3sf_EOImpl.getPrice provides the user with more specific information regarding that method invocation and SmarTune's educated guess as to how to improve the application:
Aligning the problems with the lack of Best Practices

In the application under scrutiny, two best practices are already in place: JNDI resources are being cached and reused, and a Session Façade is in place. During development, it was not believed that data caching was necessary and indeed the application behaved as expected. Under load, the application behavior changed and the “lightweight” call to the Stateless Session Bean to obtain the prices of items became expensive and overall application performance suffered.

SmarTune recommends applying the best practice of caching data in our application. Modifications are made so that the item list information is cached in memory in the web application layer after the first call to the SessionBean. Instantly, the user experience is improved as the application becomes more responsive. Running the same load test against the application reveals the following analysis of the same method at peak load by SmarTune:

Through caching the data, the ratio of EJB/JSP calls has been reduced from 25 to 1 down to 5.7 to 1. Also, the average response time of the JSP has been cut to 25%.

SmarTune goes on to suggest further tuning of the application, and encouragement is given to develop further ways of increasing responsiveness. Identifying the method getPrice as long-lived encourages the developers to tune the transactional behavior of that method and further responsiveness is achieved. A load test is now performed in series against the original application and the optimized application, and the following graph is generated by Precise Indepth For J2EE:

![Graph showing performance improvement after caching](image-url)
Conclusion

When developing complex distributed systems, there shall always be areas of optimization that are missed. Additionally, there will always be unexpected behavior on the part of the users, the network, and the supporting infrastructure. Recognizing the existence of these missing optimizations, or performance sinkholes, and identifying the paths to correcting them is of utmost importance throughout the lifecycle of an application. Precise/Indepth for J2EE is a powerful tool in the hands of anyone seeking to locate and repair performance problems in J2EE Applications.

Appendix: Precise/Indepth for J2EE In a Nutshell

In January of 2002 Precise released version 2 of their J2EE application performance management solution called, Precise/Indepth for J2EE. An install of the product results in the installation of several components that work together to store, access, and analyze performance data generated from a set of specified JVMs. The components include:
Precise/Indepth for J2EE performs its analysis by "Instrumenting the Java byte code" of your Application Server, the JDBC driver used, and your application. The Instrumentation is really a modification of the Byte code at run-time when it is loaded by the Virtual Machine. This modification enables the gathering of time stamps at the beginning and end of each method as well as the tracing of individual Threads of execution. This gathered data is sent to the Collector Agent (Currently implemented as a C++ component installed on the machine hosting the "Instrumented" VM). The Collector Agent then sends the information on to the Archive Agent that stores it in the Precise database.

Upon the request of a user, the Analysis Agent obtains the data from the Precise database and uses it to create evaluations of the J2EE application's performance behavior and display it textually and graphically in a web browser. The data is gathered in discrete chunks and refreshed according to a configurable polling interval.

**Figure 7 - A deployment of Precise/Indepth for J2EE**
About Precise Software Solutions

Precise Software Solutions is a leading global provider of business Application Performance Management solutions. Precise has an ongoing commitment to deliver proactive performance management solutions that enable companies to significantly improve the performance of their mission-critical applications, and maximize the return on their technology investments.

Headquartered in Westwood, Massachusetts, Precise has over 5,500 clients worldwide. The company serves these customers through offices in the United States, Israel, and United Kingdom, as well as via a global network of distributors worldwide.

>For More information

PRECISE SOFTWARE SOLUTIONS

Corporate Headquarters
Precise Software Solutions, Inc.
690 Canton Street
Westwood, MA 02090
USA
T: +1 781 461 0700
  +1 800 310 4777 (USA & Canada)
E: info@precise.com

Israel
Precise Software Solutions Ltd.
10, Hata’asiya Street
P.O. Box 1066
Or-Yehuda 60408
T: +972 3 735 2222
E: info.il@precise.com

Canada
T: +1 416 229 2086
E: info@precise.com

Benelux
T: +31 (0) 30 60 59 090
E: info.nl@precise.com

France
T: +33 (0) 1 41 41 9895
E: info.fr@precise.com

Germany
T: +49 (0) 221 4855 790
E: info.de@precise.com

UK
T: +44 (0) 1223 716 660
E: info.uk@precise.com

Australia
T: +61 2 9025 3718
E: info.au@precise.com

China
T: +86 21 63906 207
E: info.cn@precise.com

Japan
T: 81 3 5219 1216
E: info.jp@precise.com

www.Precise.com

Please visit our website for a complete list of distributors.