Title: APPARATUS AND METHOD FOR LIGHTWEIGHTING APPLICATION THROUGH DUPLICATED APPLICATION MANAGEMENT IN OSGI MIDDLEWARE ENVIRONMENT OF EMBEDDED TERMINAL

Abstract: Provided is an apparatus and method for lightweighting an application through duplicated application management in an Open Service Gateway Initiative (OSGI) middleware environment of an embedded terminal including at least one OSGI service bundle and native module operating based on an OSGI framework. The apparatus includes: a linker providing an interface to the OSGI service bundle for implement duplication of the application when executing the application and exchanging a message, which is required to execute the application, with an integrated manager connecting to the native module; the integrated manager providing the interface to implement duplication of the application and exchanging the message with the linker; and a determiner determining whether a native module related to the OSGI service bundle exists and executes an application according to the OSGI service bundle and removes the application if it is determined that the native module exists or determining application execution through the message exchange between the linker and the integrated manager if it is determined that the native module does not exist. Accordingly, by duplicating application management using a linker and an integrated manager unlike a conventional OSGI middleware, an application can be lightweighted by being separately implemented as modules optimized to each environment.
Description

APPARATUS AND METHOD FOR LIGHTWEIGHTING APPLICATION THROUGH DUPLICATED APPLICATION MANAGEMENT IN OSGi MIDDLEWARE ENVIRONMENT OF EMBEDDED TERMINAL

Technical Field

[1] The present invention relates to an apparatus and method for lightweighting an application through duplicated application management in an Open Service Gateway Initiative (OSGi) middleware environment of an embedded terminal.

Background Art

[2] OSGi stands for Open Service Gateway Initiative and is a noncommercial standardization organization established in March 1999. Over 40 members of home appliances/electronics, motors, communication, and Information & Technology (IT) fields are briskly working. The role of OSGi is to make open standards for transferring a service to a local network or device and operating the transferred service. An OSGi service platform is largely interested in three fields: connection and control between services; connection and control between a service and an OSGi middleware, and connection and control between an OSGi middleware and an external service management system. OSGi is a standard based on a Java virtual machine. The Java virtual machine acts as a buffer on differences between heterogeneous embedded Operating System (OS) and Central Processing Unit (CPU).

[3] An OSGi service is contained in a physical package called a bundle. A plurality of OSGi services may be contained in a single bundle, and the bundle is a basic unit of distribution and management. OSGi middleware manages bundle(s). The OSGi middleware performs registration/inquiry/execution/uninstall on a service with a service registry. The OSGi middleware also searches for and responds to an event. The event is a logical event based on three event creators, such as a service, a bundle, and a framework without association with a physical event created by a device.

[4] Since OSGi is based on Java, OSGi has the easiness in management and development, and thus, an OSGi service platform is one of competitive service platforms suitable for an advanced service environment. However, the biggest problem of OSGi is derived from the fact that OSGi is Java-oriented. That is, the selection of a Java virtual machine causes a performance decrease due to an overhead of a dynamic execution environment when an application is executed in an embedded terminal environment. In addition, if a user interface (UI) is implemented using graphic libraries provided by Java, since too much memory space is required as compared to imple-
mentation of a native type UI, the execution load is too heavy to execute an application in the embedded terminal environment. In addition, restrictions of embedded Java virtual machines limit implementation of applications. For some commercialized embedded Java virtual machines, since a memory address space is physically limited, the number of executable OSGi bundles is limited.


**Disclosure of Invention**

**Technical Problem**

The present invention provides an apparatus and method for lightweighting an application in an embedded terminal by separately implementing the application so as to be optimized to each environment through duplicated application management and interoperating the separated applications through communication in an Open Service Gateway Initiative (OSGi) middleware environment of the embedded terminal.

The present invention also provides an apparatus and method for separately implementing and lightweighting an application as modules optimized to each environment and removing restrictions of an embedded Java environment by performing duplicated application management using an OSGi-to-native linker and a native integrated manager unlike conventional OSGi middleware.

**Technical Solution**

According to an aspect of the present invention, there is provided an apparatus for lightweighting an application through duplicated application management in an OSGi middleware environment of an embedded terminal comprising at least one OSGi service bundle and native module operating based on an OSGi framework, the apparatus comprising: a linker providing an interface to the OSGi service bundle for implement duplication of the application when executing the application and exchanging a message, which is required to execute the application, with an integrated manager connecting to the native module; the integrated manager providing the interface to implement duplication of the application and exchanging the message with the linker; and a determiner determining whether a native module related to the OSGi service bundle exists and executes an application according to the OSGi service bundle.
and removes the application if it is determined that the native module exists or determining application execution through the message exchange between the linker and the integrated manager if it is determined that the native module does not exist.

According to another aspect of the present invention, there is provided a method of lightweighting an application through duplicated application management in an OSGi middleware environment of an embedded terminal, the method comprising: determining whether a native module is included in a received OSGi service bundle; if it is determined that the native module is not included in the received OSGi service bundle, installing, executing, and removing the OSGi service bundle; and if it is determined that the native module is included in the received OSGi service bundle, executing an application in association with the native module.

Advantageous Effects

As described above, according to the present invention, by duplicating application management using a linker and an integrated manager unlike a conventional OSGi middleware, an application can be lightweighted by being separately implemented as modules optimized to each environment. The duplicated application can remove restrictions of a Java environment. The linker and the integrated manager allow the duplication of the application to be easily implemented by providing interfaces for an OSGi bundle and a native module. In addition, the duplicated application has compatibility with applications executed on the conventional OSGi middleware.

Description of Drawings

FIG. 1 is a block diagram of an apparatus for lightweighting an application through duplicated application management in an Open Service Gateway Initiative (OSGi) middleware environment of an embedded terminal according to an embodiment of the present invention;

FIG. 2 is a system configuration for exchanging a message between a linker and an integrated manager illustrated in FIG. 1, according to an embodiment of the present invention;

FIG. 3 is a structure of a message for communication between a linker and an integrated manager illustrated in FIG. 2, according to an embodiment of the present invention; and

FIG. 4 is a flowchart illustrating a process of processing a lifecycle of an application lightweighted through duplicated application management

according to an embodiment of the present invention.

Best Mode

According to an aspect of the present invention, there is provided an apparatus for lightweighting an application through duplicated application management in an OSGi
middleware environment of an embedded terminal comprising at least one OSGi service bundle and native module operating based on an OSGi framework, the apparatus comprising: a linker providing an interface to the OSGI service bundle for implement duplication of the application when executing the application and exchanging a message, which is required to execute the application, with an integrated manager connecting to the native module; the integrated manager providing the interface to implement duplication of the application and exchanging the message with the linker; and a determiner determining whether a native module related to the OSGi service bundle exists and executes an application according to the OSGi service bundle and removes the application if it is determined that the native module exists or determining application execution through the message exchange between the linker and the integrated manager if it is determined that the native module does not exist.

According to another aspect of the present invention, there is provided a method of lightweighting an application through duplicated application management in an OSGi middleware environment of an embedded terminal, the method comprising: determining whether a native module is included in a received OSGi service bundle; if it is determined that the native module is not included in the received OSGi service bundle, installing, executing, and removing the OSGi service bundle; and if it is determined that the native module is included in the received OSGi service bundle, executing an application in association with the native module.

Mode for Invention

Hereinafter, the present invention will be described in detail by explaining preferred embodiments of the invention with reference to the attached drawings.

FIG. 1 is a configuration of an entire system for lightweighting an application through duplicated application management in an Open Service Gateway Initiative (OSGi) middleware environment of an embedded terminal according to an embodiment of the present invention. Referring to FIG. 1, the system includes a basement terminal hardware 100, an Operating System (OS) 101, a Java virtual machine 110, an OSGi framework 111, an OSGi-to-native linker 112, a plurality of OSGi bundles 113, a native integrated manager 120, and a plurality of native modules 121. The basement terminal hardware 100 denotes an embedded terminal of a cellular phone, a Personal Digital Assistant (PDA), a Digital Multimedia Broadcasting (DMB) terminal, or a telematics terminal. The OS 101 provides the least environment for executing a program in an embedded terminal and operates the OSGi framework 111 independent to a frame through the Java virtual machine 110. An application can be implemented in two types: a singular type of an OSGi bundle 113; and a duplicated type of an OSGi bundle 113 and a native module 121. For an application in the duplicated type of an OSGi bundle 113 and a native module 121, the application can be
lightweighted by implementing a portion requiring much memory space, such as a User Interface (UI) using a native module 121 or separately implementing application modules so as to be optimized to each environment, and thereby, restrictions of the Java virtual machine 110 can be overcome. When the lightweighted application is executed, the OSGi bundle 113 and the native module 121 work together, and this process is managed by the OSGi-to-native linker 112 and the native integrated manager 120 through a message exchange. In addition the OSGi-to-native linker 112 and the native integrated manager 120 allows the duplication of the application to be easily implemented by providing interfaces for the OSGi bundle 113 and the native module 121.

FIG. 2 is a system configuration for exchanging a message between application modules using an OSGi-to-native linker 200 and a native integrated manager 210 in order to execute the lightweighted application illustrated in FIG. 1, according to an embodiment of the present invention. Referring to FIG. 2, the system includes the OSGi-to-native linker (hereinafter, linker) 200, the native integrated manager (hereinafter, integrated manager) 210, an OSGi bundle 205, and a native module 215 related to the integrated manager 210, wherein the native module 215 is classified into a stand-alone native module and a passive Dynamic Link Library (DLL) native module. The linker 200 includes a socket server 201, a first message broker 202, a first application manager 203, and a first table 204, and the integrated manager 210 includes a socket client 211, a second message broker 212, a second application manager 213, and a second table 214. The socket server 201 and the socket client 211 take in charge of communication by forming a communication channel between the linker 200 and the integrated manager 210. Each of the first message broker 202 and the second message broker 212 creates a message to be transmitted and transmits the message through a socket, or analyzes a received message and performs an operation according to the analysis result. Each of the first application manager 203 and the second application manager 213 manages a lifecycle of an application and records a state of the application in the first table 204 or the second table 214. The first application manager 203 and the second application manager 213 exchange a message for managing the lifecycle of the application through the first message broker 202 and the second message broker 212. The OSGi bundle 205 and the native module 215 constituting the application interoperate by exchanging a message. That is, when the message is created and transmitted to the first message broker 202 or the second message broker 212, the first message broker 202 or the second message broker 212 transmits the message to the other party through the socket. The first message broker 202 or the second message broker 212, which has received the message, analyzes the message, determines which module the message is transmitted, and transmits the
message to the determined module.

[21] A method of managing a duplicated application in an OSGi middleware environment of an embedded terminal will now be described with reference to FIGS. 3 and 4. FIG. 3 is a structure of a message for communication between the linker 200 and the integrated manager 210 illustrated in FIG. 2, according to an embodiment of the present invention, and FIG. 4 is a flowchart illustrating a process of processing a lifecycle of an application lightweighted through duplicated application management according to an embodiment of the present invention.

[22] Referring to FIG. 3, the message has a binary, extensible Markup Language (XML), or Hypertext Markup Language (HTML) format. A message start field 301 and a message end field 304 are located in both ends of the message, and a head field 302 and a body field 303 exist between the message start field 301 and the message end field 304. The head field 302 includes a target field 311, a message type field 312, and a message related parameters field 313. The body field 303 includes a plurality of target related parameters fields 321. When the message is received, the first message broker 202 or the second message broker 212 analyzes the head field 302. In the target field 311, a name of the OSGi bundle 205 or the native module 215 to which the message is transmitted is recorded. In the message type field 312, a message type, i.e., a type of a message for managing a lifecycle or a message exchanged between application modules, is recorded. In the message related parameters field 313, parameters according to the message type field 312 are recorded. If the first message broker 202 or the second message broker 212 analyzes the head field 302 and determines that a target of the message is an application module, the first message broker 202 or the second message broker 212 transmits the body field 303 to the target application module. Since the body field 303 follows an application-related specification, only the application can analyze the body field 303.

[23] How to process a lifecycle in which an application is installed, started, stopped after the application is completed, and uninstalled through duplicated application management will now be described with reference to FIG. 4. A lifecycle of an application begins when the application is received to a terminal in a bundle type in which a native module is compressed. An OSGi service bundle is received and decompressed in operation 401. It is determined in operation 402 whether a service-related native module is included in the service bundle. If it is determined in operation 402 that the native module is not included, as well as the prior art, the OSGi service bundle is installed in operation 421, started in operation 422, stopped in operation 423, and uninstalled in operation 424.

[24] If it is determined in operation 402 that the native module is included, a name of the native module is detected when the OSGi service bundle is installed in operation 411.
The name of the native module is registered in the first table 204 in operation 412. The native module is loaded in operation 413 and registered in the second table 214 in operation 414. The OSGi service bundle is started in operation 415, and the application is executed through communication between the OSGi service bundle and the native module in operation 416. When the OSGi service bundle is stopped in operation 417, the interoperation between the OSGi service bundle and the native module is terminated, and when the OSGi service bundle is uninstalled in operation 418, the native module is released in operation 419.

The invention can also be embodied as computer readable codes on a computer readable recording medium. The computer readable recording medium is any data storage device that can store data which can be thereafter read by a computer system. Examples of the computer readable recording medium include read-only memory (ROM), random-access memory (RAM), CD-ROMs, magnetic tapes, floppy disks, optical data storage devices, and carrier waves (such as data transmission through the Internet). The computer readable recording medium can also be distributed over network coupled computer systems so that the computer readable code is stored and executed in a distributed fashion.

As described above, according to the present invention, by duplicating application management using a linker and an integrated manager unlike a conventional OSGi middleware, an application can be lightweighted by being separately implemented as modules optimized to each environment. The duplicated application can remove restrictions of a Java environment. The linker and the integrated manager allow the duplication of the application to be easily implemented by providing interfaces for an OSGi bundle and a native module. In addition, the duplicated application has compatibility with applications executed on the conventional OSGi middleware.

While this invention has been particularly shown and described with reference to preferred embodiments thereof, it will be understood by those skilled in the art that various changes in form and details may be made therein without departing from the spirit and scope of the invention as defined by the appended claims. The preferred embodiments should be considered in descriptive sense only and not for purposes of limitation. Therefore, the scope of the invention is defined not by the detailed description of the invention but by the appended claims, and all differences within the scope will be construed as being included in the present invention.

Industrial Applicability

The present invention relates to an apparatus and method for lightweighting an application through duplicated application management in an Open Service Gateway Initiative (OSGi) middleware environment of an embedded terminal.
**Claims**

[1] An apparatus for lightweighting an application through duplicated application management in an Open Service Gateway Initiative (OSGi) middleware environment of an embedded terminal comprising at least one OSGi service bundle and native module operating based on an OSGi framework, the apparatus comprising:

- a linker providing an interface to the OSGI service bundle for implementing duplication of the application when executing the application and exchanging a message, which is required to execute the application, with an integrated manager connecting to the native module;
- the integrated manager providing the interface to implement duplication of the application and exchanging the message with the linker; and
- a determiner determining whether a native module related to the OSGi service bundle exists and executes an application according to the OSGi service bundle and uninstalls the application if it is determined that the native module exists or determining application execution through the message exchange between the linker and the integrated manager if it is determined that the native module does not exist.

[2] The apparatus of claim 1, wherein the linker comprises:

- a socket server forming a communication channel with the integrated manager;
- a first message broker creating and transmitting the message, or analyzing a message received from the integrated manager and transmitting the message to a relevant module;
- a first application manager managing information containing a lifecycle of the application; and
- a first table storing the information.

[3] The apparatus of claim 1, wherein the integrated manager comprises:

- a socket client forming a communication channel with the linker;
- a second message broker creating and transmitting the message, or analyzing a message received from the linker and transmitting the message to a relevant module;
- a second application manager managing information containing a lifecycle of the application; and
- a second table storing the information.

[4] The apparatus of one of claims 1 through 3, wherein the message has a binary, extensible Markup Language (XML), or Hypertext Markup Language (HTML) format.
A method of lightweighting an application through duplicated application management in an Open Service Gateway Initiative (OSGi) middleware environment of an embedded terminal, the method comprising:
determining whether a native module is included in a received OSGi service bundle;
installing, executing, and removing the OSGi service bundle, if it is determined that the native module is not included in the received OSGi service bundle; and executing an application in association with the native module if it is determined that the native module is included in the received OSGi service bundle.

The method of claim 5, wherein executing the application comprises providing an interface for duplication to the OSGi service bundle and the native module and creating and exchanging a message required to execute the application when the application is executed.

The method of claim 6, wherein providing the interface and creating and exchanging the message comprises:
forming a communication channel between the OSGi service bundle and the native module;
installing the OSGi service bundle, detecting a name of the native module, and registering the name of the native module in a first table;
executing the native module and registering the native module in a second table;
analyzing the transmitted and received message and transmitting the message to a relevant module;
when the OSGi service bundle is stopped, terminating the native module and removing the native module from the first and second table; and uninstalling the OSGi service bundle.

The method of claim 6 or 7, wherein the message has a binary, extensible Markup Language (XML), or Hypertext Markup Language (HTML) format.

A computer readable recording medium storing a computer readable program for executing a method of lightweighting an application through duplicated application management in an Open Service Gateway Initiative (OSGi) middleware environment of an embedded terminal, the method comprising:
determining whether a native module is included in a received OSGi service bundle;
if it is determined that the native module is not included in the received OSGi service bundle, installing, executing, and removing the OSGi service bundle; and
if it is determined that the native module is included in the received OSGi service bundle, executing an application in association with the native module.
FIG. 1

113
OSGi
SERVICE
BUNDLE

112
OSGi-to-NATIVE LINKER

111
OSGi

110
JAVA VIRTUAL MACHINE

101
OPERATING SYSTEM

100
HARDWARE

120
NATIVE INTEGRATED MANAGER

121
NATIVE MODULE

NATIVE
MODULE

NATIVE
FIG. 4

START

RECEIVE AND DECOMPRESS SERVICE BUNDLE

NO

IS SERVICE-RELATED NATIVE MODULE INCLUDED?

YES

INSTALL SERVICE BUNDLE

INSTALL SERVICE BUNDLE

REGISTER NATIVE MODULE IN FIRST TABLE

LOAD NATIVE MODULE

REGISTER NATIVE MODULE IN SECOND TABLE

START SERVICE BUNDLE

START SERVICE BUNDLE

COMMUNICATE BETWEEN SERVICE BUNDLE AND NATIVE MODULE

STOP SERVICE BUNDLE

UNINSTALL SERVICE BUNDLE

RELEASE NATIVE MODULE

END
A. CLASSIFICATION OF SUBJECT MATTER

G06F 17/00(2006.01)1

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

IPC G06F 17/00

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Korean Utility models and applications for Utility models since 1975

Japanese Utility models and applications for Utility models since 1975

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

ekipass "OSGi, middleware, embedded terminal, light-weighting, framework, bundle, gateway"

C. DOCUMENTS CONSIDERED TO BE RELEVANT

<table>
<thead>
<tr>
<th>Category</th>
<th>Citation of document, with indication, where appropriate, of the relevant passages</th>
<th>Relevant to claim</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>KR 1020060042412 A (KOREA INFORMATION &amp; COMMUNICATION UNIVERSITY) MAY 2006</td>
<td>1-9</td>
</tr>
<tr>
<td>A</td>
<td>See abstract, figures 1, 6 claims 1,2,3,9</td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>KR 1020050087129 A (SAMSUNG ELECTRONICS CO., LTD) 31 AUG 2005</td>
<td>1-9</td>
</tr>
<tr>
<td>A</td>
<td>See abstract, figure 1 claims 1,5</td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>KR 1020040033172 A (ELECTRONICS AND TELECOMMUNICATIONS RESEARCH INSTITUTE) 21 APR 2004</td>
<td>1-9</td>
</tr>
<tr>
<td>A</td>
<td>See abstract, figures 2,4 claims 1,2,5</td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>KR 102005006066A (ELECTRONICS AND TELECOMMUNICATIONS RESEARCH INSTITUTE) 30 JUN 2005</td>
<td>1-9</td>
</tr>
<tr>
<td>A</td>
<td>See abstract, figures 2,6 claims 1,2,5</td>
<td></td>
</tr>
</tbody>
</table>

See patent family annex

Date of the actual completion of the international search

10 SEPTEMBER 2007 (10.09.2007)

Date of mailing of the international search report

10 SEPTEMBER 2007 (10.09.2007)

Name and mailing address of the ISA/KR

Korean Intellectual Property Office
920 Dunsan-dong, Seo-gu, Daejeon 302-701, Republic of Korea

Facsimile No 82-42-472-7140

Authorized officer

KIM, Jung Jin

Telephone No 82-42-481-5962

Form PCT/ISA/210 (second sheet) (April 2007)
<table>
<thead>
<tr>
<th>Patent document cited in search report</th>
<th>Publication date</th>
<th>Patent family member(s)</th>
<th>Publication date</th>
</tr>
</thead>
<tbody>
<tr>
<td>KR1020060424 12A</td>
<td>12.05.2006</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>KR1020050087 129A</td>
<td>31.08.2005</td>
<td>AU2005215546A1</td>
<td>01.09.2005</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CN1842772A</td>
<td>04.10.2006</td>
</tr>
<tr>
<td></td>
<td></td>
<td>JP2007519117T2</td>
<td>12.07.2007</td>
</tr>
<tr>
<td></td>
<td></td>
<td>US2005195390A1</td>
<td>08.09.2005</td>
</tr>
<tr>
<td></td>
<td></td>
<td>W02005081106A1</td>
<td>01.09.2005</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>KR102004033 172A</td>
<td>21.04.2004</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>KR102005066066A</td>
<td>30.06.2005</td>
<td>US2005144262A1</td>
<td>30.06.2005</td>
</tr>
</tbody>
</table>

Form PCT/ISA/210 (patent family annex) (April 2007)