(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
7 January 2016 (07.01.2016)

WIPO | PCT

(10) International Publication Number
WO 2016/003412 A1

(51) International Patent Classification:
G06F 17/30 (2006.01)

(21) International Application Number:
PCT/US2014/044875

(22) International Filing Date:
30 June 2014 (30.06.2014)

(25) Filing Language: English

(26) Publication Language: English

(71) Applicant: **HEWLETT-PACKARD DEVELOPMENT COMPANY, L.P.** [US/US]; 11445 Compaq Center Drive West, Houston, Texas 77070 (US).

(72) Inventor: **VAQUERO GONZALEZ, Luis Miguel**; Longdown Avenue, Stoke Gifford, Bristol South Gloucestershire BS34 8QZ (GB).

(74) Agents: **BELL, Scott** et al.; Hewlett-Packard Company, Intellectual Property Administration, 3404 E. Harmony Road, Mail Stop 35, Fort Collins, Colorado 80528 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JP, KE, KG, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Declarations under Rule 4.17:

— *as to the identity of the inventor (Rule 4.17(i))*

— *as to applicant's entitlement to apply for and be granted a patent (Rule 4.17(ii))*

Published:

— *with international search report (Art. 21(3))*

WO 2016/003412 A1

(54) Title: CONCURRENT DEAD ACTOR COLLECTION

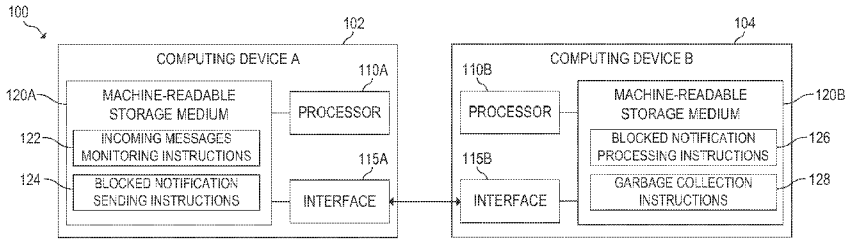

FIG. 1

(57) Abstract: Examples relate to providing concurrent dead actor collection. In some examples, a blocked notification is received from an actor of a number of actors in a distributed system, where the actors are arranged in an actor hierarchy that describes communication links between the actors. In response to receiving the blocked notification, a blocked status is requested from each other actor in a loop subset of the actors, where each of the other actors is connected to the actor in the actor hierarchy by an incoming edge. After using the blocked status of each of the other actors to determine that incoming edges of the actor refer to blocked actors, the actor is designated for garbage collection.

CONCURRENT DEAD ACTOR COLLECTION

BACKGROUND

[0001] Actors are commonly used to facilitate concurrent programming on many-core machines. After an actor has completed its tasks, the actor's associated resources should be collected for reuse. In many cases, actor-able languages and libraries rely on developers to finalize their actors explicitly. Typical automated garbage collection mechanisms in many programming languages can also be used to collect dead actors. In this case, the garbage collection may rely on synchronization mechanisms (e.g., barriers and locks) and cache coherence that do not allow for concurrent or dead actors collection.

BRIEF DESCRIPTION OF THE DRAWINGS

[0002] The following detailed description references the drawings, wherein:

[0003] FIG. 1 is a block diagram of an example computing device for concurrent dead actor collection;

[0004] FIG. 2 is a block diagram of an example distributed system in communication over a network for providing concurrent dead actor collection;

[0005] FIG. 3 is a flowchart of an example method for execution by a computing device for providing concurrent dead actor collection;

[0006] FIG. 4 is a workflow diagram showing an example system for providing concurrent dead actor collection; and

[0007] FIG. 5 is an example actor hierarchy to be used for concurrent dead actor collection.

DETAILED DESCRIPTION

[0008] As discussed above, garbage collection for actor-able languages and actor frameworks (e.g., Simple Actor Language System and Architecture (SALSA), AmbientTalk, Erlang, etc.) is typically unable to provide concurrent or dead actors collection. In examples described herein, dead actors are correctly

identified and designated for garbage collection, where live actors can continue execution while the dead actors are concurrently collected. The actor model states that an actor can send a finite number of asynchronous messages to a buffer addressed to other actors with no delivery guarantees, then the other actors select a proper behavior to handle the next incoming message and may create a finite number of child actors. Once an actor has finished its queue of incoming messages, the actor is considered to be blocked. If a blocked actor is referenced by only blocked actors, then the actor is also considered to be a dead actor.

[0009] Examples described herein use an actor hierarchy to enable concurrent collection of dead actors. Specifically, each actor monitors incoming messages to determine when the actor becomes blocked. The actors in a distributed system may be configured according to the actor hierarchy such that a loop detector manages garbage collection for a subset of actors in the hierarchy. A loop detector is a specialized actor in the distributed system that monitors the blocked status of a subset of actors. Based on the blocked status of the actors in the subset, the loop detector is able to designate actors for garbage collection. Specifically, the blocked actor can send a blocked notification to the loop detector, which then checks the blocked status of actors in the subset. If the incoming edges of the blocked actor are from other blocked actors, the loop detector can designated the blocked actor for garbage collection.

[0010] The actor hierarchy describes the relationship between actors in the distributed system. With respect to an actor, incoming edges in the actor hierarchy correspond to messages received by the actor, and outgoing edges correspond to messages sent by the actor. The reference count for the incoming edges and the outgoing edges can be separately monitored by each actor in the distributed system. For example, as incoming messages are received by the actor, the reference count for incoming edges is incremented. In this example, as the incoming messages are handled, the reference count is decremented. When the reference count becomes zero, the actor is designated as blocked.

[0011] Examples disclosed herein provide concurrent dead actor collection. In some examples, a blocked notification is received from an actor of a number

of actors in a distributed system, where the actors are arranged in an actor hierarchy that describes communication links between the actors. In response to receiving the blocked notification, a blocked status is requested from each other actor in a loop subset of the actors, where each of the other actors is connected to the actor in the actor hierarchy by an incoming edge. After using the blocked status of each of the other actors to determine that incoming edges of the actor refer to blocked actors, the actor is designated for garbage collection.

[0012] Referring now to the drawings, FIG. 1 is a block diagram of an example computing devices 102, 104 for providing concurrent dead actor collection. Each computing devices 102, 104 may be any computing device (e.g., server, distributed node, desktop computer, etc.) that can be configured for a distributed system. In the example of FIG. 1, each computing device 102, 104 includes a processor 110A, 110B, an interface 115A, 115B, and a machine-readable storage medium 120A, 120B.

[0013] Each processor 110A, 110B may be any number of central processing units (CPUs), microprocessors, and/or other hardware devices suitable for retrieval and execution of instructions stored in machine-readable storage medium 120A, 12B. Processor 110A, 110B may fetch, decode, and execute instructions 122, 124 or 126, 128, respectively, to provide concurrent dead actor collection as described below. As an alternative or in addition to retrieving and executing instructions, processor 110A, 110B may include any number of electronic circuits comprising a number of electronic components for performing the functionality of instructions 122, 124 or 126, 128.

[0014] Each interface 115A, 115B may include a number of electronic components for communicating with networked devices. For example, each interface 115A, 115B may be an Ethernet interface, a Universal Serial Bus (USB) interface, an IEEE 1394 (Firewire) interface, an external Serial Advanced Technology Attachment (eSATA) interface, or any other physical connection interface suitable for communication with other nodes in a distributed system. Alternatively, each interface 115A, 115B may be a wireless interface, such as a

wireless local area network (WLAN) interface or a near-field communication (NFC) interface. In operation, as detailed below, each interface 115A, 115B may be used to send and receive data to and from a corresponding interface(s) of distributed nodes.

[0015] Each machine-readable storage medium 120A, 120B may be any electronic, magnetic, optical, or other physical storage device that stores executable instructions. Thus, each machine-readable storage medium 120A, 120B may be, for example, Random Access Memory (RAM), an Electrically-Erasable Programmable Read-Only Memory (EEPROM), a storage drive, an optical disc, and the like. As described in detail below, each machine-readable storage medium 120A, 120B may be encoded with executable instructions for providing concurrent dead actor collection.

[0016] On computing device A 102, incoming messages monitoring instructions 122 monitors the processing of incoming messages to determine a reference count of computing device A 102. The reference count for incoming messages is the number of incoming messages that are queued for processing on computing device A 102. After an incoming message is processed, the reference count is decremented. As incoming messages are received and processed, computing device A 102 updates the reference count accordingly. If the reference count becomes zero, a blocked notification may be sent as described below.

[0017] Blocked notification sending instructions 124 of computing device A 102 sends a blocked notification to computing device B 104. After computing device A 102 has processed all incoming messages and determined that the reference count is zero, the blocked notification is sent to computing device B 104, which in this example may be a loop detector. The blocked notification notifies computing device B 104 that computing device A 102 is blocked and a candidate for garbage collection.

[0018] On computing device B 104, blocked notification processing instructions 126 receive and process the blocked notification from computing device A 102. The blocked notification signals that computing device A 102 is a candidate for garbage collection. In response to the blocked notification, blocked notification

processing instructions 126 may request a blocked status for each actor in a loop subset managed by computing device B 104. The loop subset includes actors (e.g., computing device A 102) that are related via communication links such that computing device A 102 should be designated for garbage collection if all actors in the loop subset are also blocked as described below. In this example, blocked notification processing instructions 126 may receive a blocked or unblocked notification from each of the actors in the loop subset, which allows computing device B 104 to update the portion of the actor hierarchy that corresponds to the loop subset.

[0019] Garbage collection instructions 128 manages garbage collection for actors in the loop subset. If garbage collection instructions 128 determine that all actors in the loop subset are blocked, the actor executing on computing device A 102A can be designated for garbage collection (i.e., the actor is a dead actor that is no longer in use). Computing device B 104 monitors each actor in the loop subset as described above to determine when each actor should be collected.

[0020] FIG. 2 is a block diagram of an example distributed system 200 in communication over a network 245 for providing concurrent dead actor collection. As illustrated, actor computing devices 202A, 202N interact with loop detectors 270A, 270N to manage garbage collection for actors executing on the actor computing devices 202A, 202N.

[0021] Similar to computing device A 102 of FIG. 1, actor computing device A 202A may be a server, distributed node, desktop computer, or any other device suitable for executing the functionality described below. As detailed below, actor computing device A 202A may include a series of modules 204A-208A for providing functionality for distributed system 200. As shown, actor computing device A 202A includes interface module 204A, degree monitor module 206A, and distributed application module 208A.

[0022] Interface module 204A may manage communications with other networked devices (e.g., loop detectors 270A, 270N, special actor 240, actor computing device N 202N, etc.). Specifically, interface module 204A may initiate connections over network 245 with the networked devices and then send blocked

notifications to and receive messages from the networked devices 202N, 240, 270A, 270N.

[0023] Degree monitor module 206A monitors the reference count of incoming and outgoing references of an actor provided by actor computing device A 202A. As incoming messages are received from other actor computing devices (e.g., actor computing device N 202N), the reference count for incoming references is determined by counting incoming messages that have not been handled. Similarly, as outgoing messages are sent to other actor computing devices (e.g., actor computing device N 202N), the reference count for outgoing references is determined by counting outgoing messages that have not been handled. By monitoring the reference counts of incoming and outgoing messages, degree monitor module 206A creates a local graph of the actor hierarchy for the actor provided by actor computing device A 202A.

[0024] The monitoring of reference counts by degree monitor module 206A is a lazy mechanism that may result in inaccurate reference counts. For example, satisfied incoming messages may not be detected in a timely fashion, which results in some dead actors remaining in the distributed system longer than needed. The lazy mechanism is tolerated because it ensures that only truly dead actors are collected.

[0025] In some cases, the actor computing devices (e.g., actor computing device N 202N) monitor each other according to the actor hierarchy. Specifically, the actor computing devices (e.g., actor computing device N 202N) of the actors are configured according to the parent-child relationships in the actor hierarchy. In this case, a child actor may send blocked notifications to its parent actor, which may then (1) determine if the child actor should be designated for garbage collection as described with respect to the loop detectors (e.g., loop detector A 270A, loop detector N 270N) below (i.e., the parent actor may act as a loop detector) or (2) propagate the blocked notification to the parent of the parent actor.

[0026] Further, some actor computing devices (e.g., actor computing device N 202N) may provide actors that exist outside the actor hierarchy. With respect to the actors outside the hierarchy, distributed system 200 behaves in a non-causal

fashion that produces some transient inconsistencies (i.e., dead actors not being timely collected). If a message arrives for a child actor and the child actor no longer exists (e.g., the child actor was removed due to lack of causality or a message timeout), the parent actor may introspect on the message and decide if the child actor should be recreated to process the message or if the child actor should stay deleted and the sender be informed to update the sender's view of the actor hierarchy.

[0027] Distributed application module 208A provides actor functionality for distributed system 200. For example, if the actor of actor computing device A is a router management module, distributed application module 208A may be a router management application. Distributed application module 208A may use services and resources of other actors (e.g., actor computing device N 202N) as shown in the actor hierarchy of distributed system 200.

[0028] In FIG. 2, actor computing device A 202A shows modules 204A-208A, which also exist in actor computing device N 202N but are not shown to simplify the figure. Any number of actor computing devices (e.g., actor computing device A 202A, actor computing device N 202N) can be included in distributed system 200. Each actor computing device 202A, 202N may be associated with a number of loop subsets of distributed system 200, where a loop subset specifies a set of actors provided by actor computing devices 202A, 202N that are related in actor hierarchy because of communication between the set of actors. Further, each actor computing device 202A, 202N may provide any number of actors depending on the configuration of distributed system 200.

[0029] Special actor 240 is configured to provide local variables and referenced provided by the runtime system. In the actor hierarchy, special actor 240 only has outgoing edges for providing its variables and runtimes to other actors. Special actor 240 is typically not considered as a candidate for garbage collection.

[0030] Garbage collector 245 performs garbage collection for distributed system 200. Specifically, garbage collector 245 collects resources of actors that have been designated for garbage collection. Garbage collector 245 is configured to concurrently collect resources as live actors (e.g., actor computing device A

202A, actor computing device N 202N) in distributed system 200 continue to operate.

[0031] Similar to computing device B 104 of FIG. 1, loop detector A 270A may be a server, distributed node, desktop computer, or any other device suitable for executing the functionality described below. As detailed below, loop detector A 270A may include a series of modules 272A-278A for providing concurrent dead actor collection. As shown, loop detector A 270A includes actor interface module 272A, actor hierarchy module 274A, blocked status module 276A, and garbage collection module 278A.

[0032] Actor interface module 272A may manage communications with other networked devices (e.g., loop detector N 270N, special actor 240, actor computing devices 202A, 202N, etc.). Specifically, interface module 272A may initiate connections over network 245 with the networked devices and then send blocked requests to and receive blocked notifications from the networked devices 202A, 202N, 240, 270N.

[0033] Actor hierarchy module 274A may monitor the actor hierarchy of a loop subset associated with loop detector A 270A. The loop subset includes actors (e.g., actor computing device A 202A, actor computing device N 202N) that can be linked by outgoing or incoming messages. In other words, actor hierarchy module 274A may monitor communication activities between actors (e.g., actor computing device A 202A, actor computing device N 202N) to determine incoming and outgoing edges in the actor hierarchy.

[0034] Blocked status module 276A may update the actor hierarchy by collecting blocked status information from actors (e.g., actor computing device A 202A, actor computing device N 202N) in the loop subset. Blocked status information allows the blocked status module 276A to determine whether each actor is blocked or unblocked.

[0035] Garbage collection module 278A may identify dead actors in the loop subset based on the actor hierarchy. Specifically, garbage collection module 278A may identify blocked actors (e.g., actor computing device A 202A, actor computing device N 202N) that are referred to by other blocked actors as dead

actors. After a dead actor is identified, garbage collection module 278A may designate the dead actor for garbage collection. Garbage collection can then collect the resources attributed to the dead actors so that the resources may be allocated to other actors in distributed system 200.

[0036] In some cases, message causality can be assumed in distributed system 200. For example, if distributed system 200 exists within a single server rack, it can be assumed that messages are received by their recipients in the same order that the messages are sent by the senders. In other cases, different latencies among devices (e.g., actor computing device A 202A, actor computing device N 202N, loop detector A 270A, loop detector N 270N, etc.) can cause messages to arrive in a different order than the messages were sent. In this case, the latency of the remote devices is accommodated for by the loop detectors (e.g., loop detector A 270A, loop detector N 270N, etc.) when monitoring the blocked status of the actors provided by actor computing devices (e.g., actor computing device A 202A, actor computing device N 202N). When message causality is applied in the actor hierarchy, messages may be sent in a generational fashion such that the messages transferred from grandparents to grandchildren pass through children as an intermediate steps.

[0037] In FIG. 2, loop detector A 270A shows modules 272A-278A, which also exist in loop detector N 270N but are not shown to simplify the figure. Any number of loop detectors (e.g., loop detector A 270A, loop detector N 270N) can be included in distributed system 200. Each loop detector 270A, 270N may manage a number of loop subsets of distributed system 200.

[0038] FIG. 3 is a flowchart of an example method 300 for execution by a computing device 104 for providing concurrent dead actor collection. Although execution of method 300 is described below with reference to computing device 104 of FIG. 1, other suitable devices for execution of method 300 may be used. Method 300 may be implemented in the form of executable instructions stored on a machine-readable storage medium, such as storage medium 120B, and/or in the form of electronic circuitry.

[0039] Method 300 may start in block 305 and continue to block 310, where computing device 104 receives a blocked notification from an actor in a distributed system. The actor may be provided by another computing device in the distributed system, and the blocked notification signals that the actor is a candidate for garbage collection. In response to the blocked notification, computing device 104 may request a blocked status for each actor in a loop subset managed by computing device 104 in block 315. In this example, computing device 104 may receive a blocked or unblocked notification from each of the actors in the loop subset, which allows computing device 104 to update the portion of the actor hierarchy that corresponds to the loop subset.

[0040] In block 320, if garbage computing device 104 determine that all actors in the loop subset are blocked, the actor can be identified as a dead actor and designated for garbage collection. Computing device 104 monitors each actor in the loop subset as described above to determine when each actor should be collected. Method 300 may then continue to block 325, where method 300 may stop.

[0041] FIG. 4 is a workflow diagram showing an example distributed system for providing concurrent dead actor collection. Workflow diagram shows a loop detector 402, an actor A 404A, an actor N 404N, and a special actor 406, which may each be similar to their corresponding components described above with respect to FIGS. 1 and 2.

[0042] In block 410, actor N 404N sends a message to actor A 404A. The message may request that actor A 404A perform an action in the distributed system. In block 412, the reference count for incoming edges is incremented in response to the incoming message. In block 414, a variable reference is provided by special actor 406 to actor A 404A. Because the variable reference does not request a response, the reference count is not incremented.

[0043] In block 415, actor A 404A provides a response to actor N 404N that notifies that the message of block 410 has been satisfied. In block 416, actor N 404N may close its communication link with actor A 404A. In block 418, actor A 404A decrements the reference count for incoming edges to zero. After the

reference count for incoming edges is set to zero, actor A 404A sends a blocked notification to loop detector 402. The blocked notification notifies loop detector that actor A 404A is a candidate for garbage collection.

[0044] In block 422, loop detector 402 requests the blocked status from actor N 404N. In block 424, actor N 404N sends a blocked notification to loop detector 402. In block 426, loop detector 402 determines that all actors in a loop subset managed by loop detector 402 are blocked and designates actor A 404A for garbage collection.

[0045] FIG. 5 is an example actor hierarchy 500 to be used for concurrent dead actor collection. The actor hierarchy includes actors 502A, 502B, 502C, 502D, loop detector 504, and special actor 506. In this example, actor A 502A, actor B 502B, and actor C 502 are in a first loop subset managed by loop detector 504, and actor C 502C and actor D 502D are in a second loop subset managed by loop detector 504.

[0046] Actor hierarchy 500 shows that actor B 502B can have incoming edges from actor A 502A and actor C 502C. Accordingly, if loop detector 504 receives a blocked notification from actor B 502B, loop detector 504 should determine the blocked status of actor A 502A and actor C 502C before designating actor B 502B for garbage collection. Special actor 506 only has outgoing edges but does not affect the reference count of actor B 502B and actor C 502C.

[0047] The foregoing disclosure describes a number of examples for concurrent dead actor collection. In this manner, the examples disclosed herein identify dead actors for garbage collection by using loop detectors to monitor the blocked status of loop subsets of actors.

CLAIMS

We claim:

1. A system for concurrent dead actor collection, comprising:

a plurality of processors that is configured to execute a distributed application of a distributed system;

a plurality of actors in the distributed system that are arranged in an actor hierarchy that describes a plurality of communication links between the plurality of plurality of actors, wherein an actor of the plurality of actors is configured to:

monitor incoming messages of the plurality of communication links to determine an incoming reference count of the actor, wherein the incoming messages correspond to incoming edges of the actor in the actor hierarchy; and

after determining the incoming reference count is zero, send a blocked notification to a loop detector, the blocked notification confirming that the incoming messages have been processed by the actor; and

the loop detector in the distributed system that is associated with a loop subset of the plurality of actors, the loop detector configured to:

in response to receiving the blocked notification, request a blocked status from each actor in the loop subset; and

after using the blocked status of each of the other actors to determine that incoming edges of the actor refer to blocked actors, designate the actor for garbage collection.

2. The system of claim 1, further comprising:

a special actor comprising local variables and references that can be provided as outgoing messages to the plurality of actors, wherein the special actor cannot be collected by the garbage collection and the outgoing messages correspond to outgoing edges of the special actor in the actor hierarchy.

3. The system of claim 1, wherein the reference count is zero when the actor has responded to the incoming messages.

4. The system of claim 1, wherein the distributed system comprises a plurality of computing devices, wherein each of the plurality of computing devices provides at least one of the plurality of actors.

5. The system of claim 1, wherein the loop detector is a parent actor of the actor in the actor hierarchy.

6. The system of claim 5, further comprising a garbage collector configured to concurrently collect resources associated with the actor while operation of the distributed application continues.

7. A method for concurrent dead actor collection, comprising:

receiving a blocked notification from an actor of a plurality of actors in a distributed system, wherein the plurality of actors are arranged in an actor hierarchy that describes a plurality of communication links between the plurality of plurality of actors;

in response to receiving the blocked notification, requesting a blocked status from each other actor in a loop subset of the plurality of actors, wherein each of the other actors is connected to the actor in the actor hierarchy by an incoming edge; and

after using the blocked status of each of the actors to determine that incoming edges of the actor refer to blocked actors, designating the actor for garbage collection.

8. The method of claim 7, wherein the block notification is sent by the actor when a reference count of incoming edges of the actor is zero.

9. The method of claim 7, wherein the distributed system comprises a plurality of computing devices, wherein each of the plurality of computing devices provides at least one of the plurality of actors.

10. The method of claim 7, wherein the loop detector is a parent actor of the actor in the actor hierarchy.

11. The method of claim 10, wherein the garbage collection concurrently collects resources associated with the actor while operation of the distributed application continues.

12. A non-transitory machine-readable storage medium encoded with instructions executable by a processor for providing concurrent dead actor collection, the machine-readable storage medium comprising instructions to:

receive a blocked notification from an actor of a plurality of actors in a distributed system after a reference count of incoming edges of the actor is determined to be zero, wherein the plurality of actors are arranged in an actor hierarchy that describes a plurality of communication links between the plurality of plurality of actors;

in response to receiving the blocked notification, request a blocked status from each other actor in a loop subset of the plurality of actors, wherein each of the other actors is connected to the actor in the actor hierarchy by an incoming edge; and

after using the blocked status of each of the other actors to determine that incoming edges of the actor refer to blocked actors, designate the actor for garbage collection.

13. The non-transitory machine-readable storage medium of claim 12, wherein the distributed system comprises a plurality of computing devices, wherein each of the plurality of computing devices provides at least one of the plurality of actors.

14. The non-transitory machine-readable storage medium of claim 12, wherein each actor of the plurality of actors provides functionality of a distributed application of the distributed system.

15. The non-transitory machine-readable storage medium of claim 14, wherein the garbage collection concurrently collects resources associated with the actor while operation of the distributed application continues.

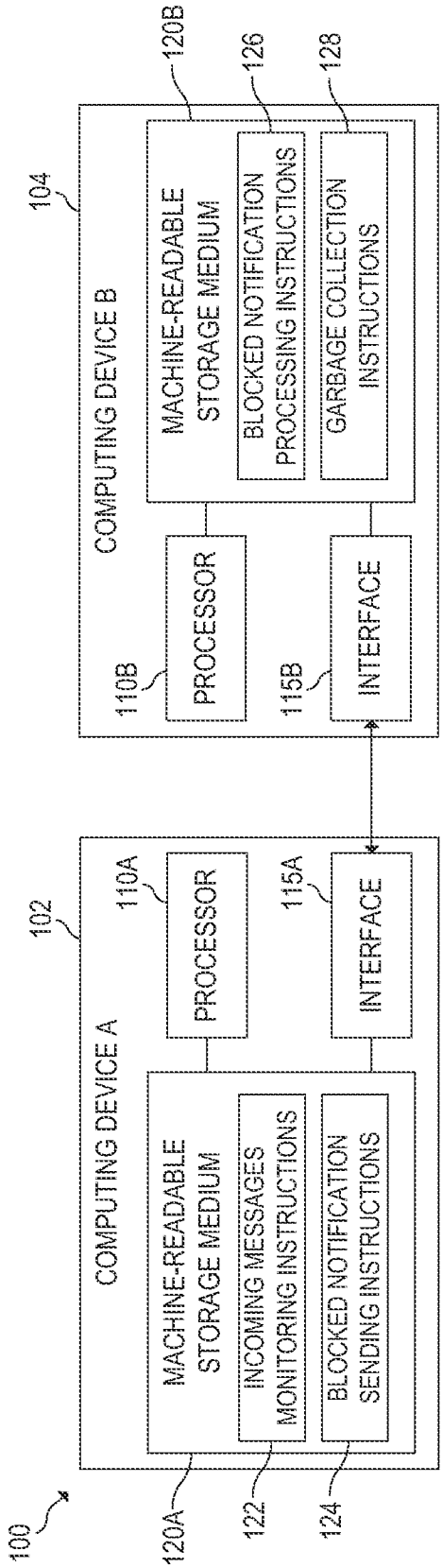
100
102
COMPUTING DEVICE A
110A
PROCESSOR
115A
INTERFACE
MACHINE-READABLE STORAGE MEDIUM
INCOMING MESSAGES MONITORING INSTRUCTIONS
BLOCKED NOTIFICATION SENDING INSTRUCTIONS
120A
122
124
104
COMPUTING DEVICE B
110B
PROCESSOR
115B
INTERFACE
120B
MACHINE-READABLE STORAGE MEDIUM
126
BLOCKED NOTIFICATION PROCESSING INSTRUCTIONS
128
GARBAGE COLLECTION INSTRUCTIONS

FIG. 1

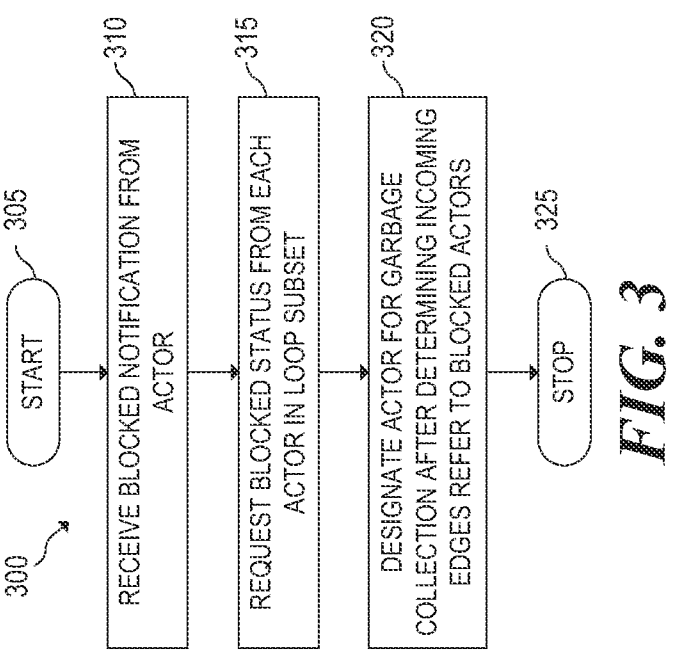
300
START
305
RECEIVE BLOCKED NOTIFICATION FROM ACTOR
310
REQUEST BLOCKED STATUS FROM EACH ACTOR IN LOOP SUBSET
315
DESIGNATE ACTOR FOR GARBAGE COLLECTION AFTER DETERMINING INCOMING EDGES REFER TO BLOCKED ACTORS
320
STOP
325

FIG. 3

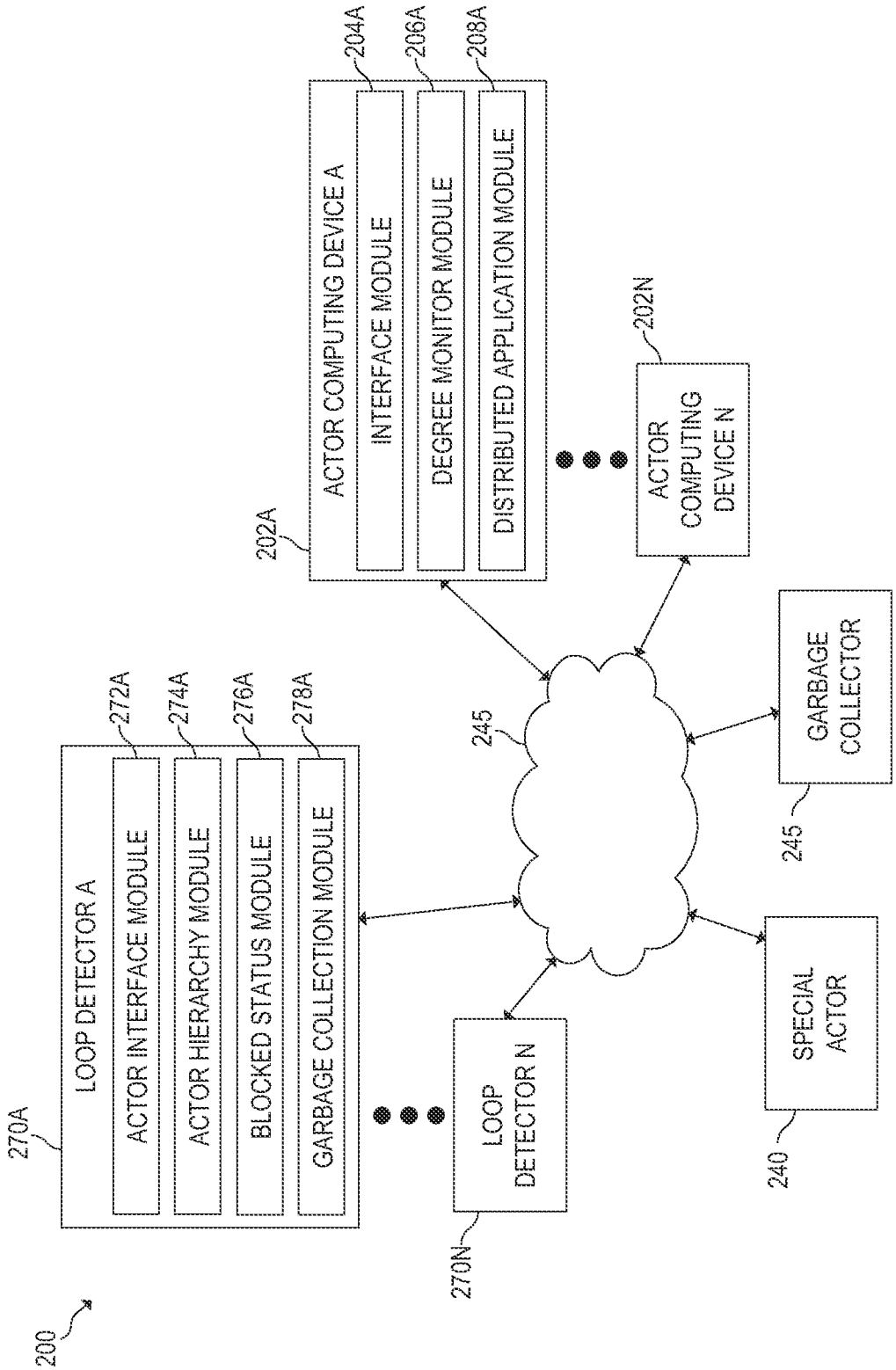
200
270A
LOOP DETECTOR A
ACTOR INTERFACE MODULE
272A
ACTOR HIERARCHY MODULE
274A
BLOCKED STATUS MODULE
276A
GARBAGE COLLECTION MODULE
278A
270N
LOOP DETECTOR N
245
240
SPECIAL ACTOR
245
GARBAGE COLLECTOR
202A
ACTOR COMPUTING DEVICE A
INTERFACE MODULE
204A
DEGREE MONITOR MODULE
206A
DISTRIBUTED APPLICATION MODULE
208A
202N
ACTOR COMPUTING DEVICE N

FIG. 2

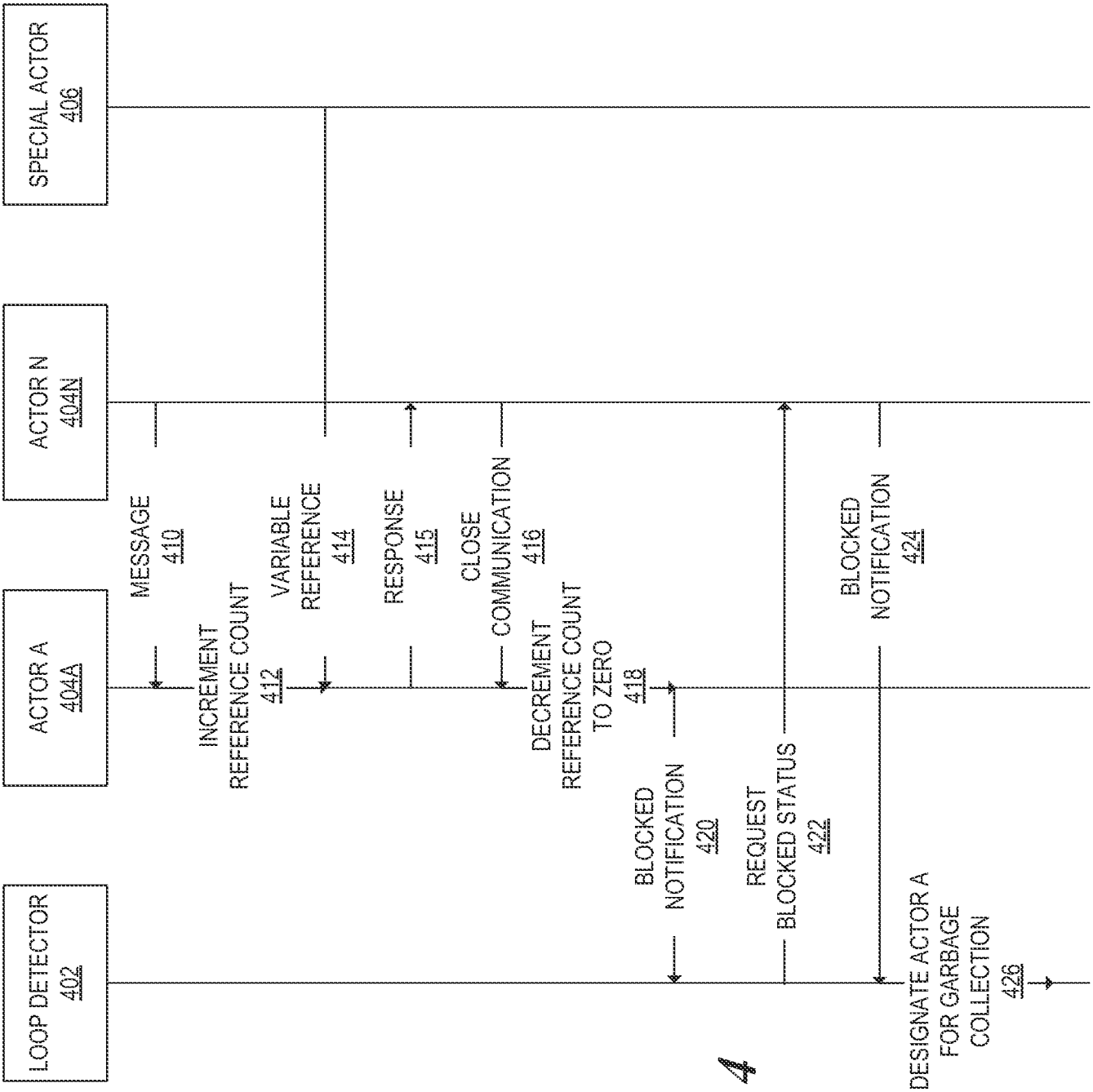
LOOP DETECTOR 402
ACTOR A 404A
ACTOR N 404N
SPECIAL ACTOR 406
MESSAGE 410
INCREMENT REFERENCE COUNT 412
VARIABLE REFERENCE 414
RESPONSE 415
CLOSE COMMUNICATION 416
DECREMENT REFERENCE COUNT TO ZERO 418
BLOCKED NOTIFICATION 420
REQUEST BLOCKED STATUS 422
BLOCKED NOTIFICATION 424
DESIGNATE ACTOR A FOR GARBAGE COLLECTION 426

FIG. 4

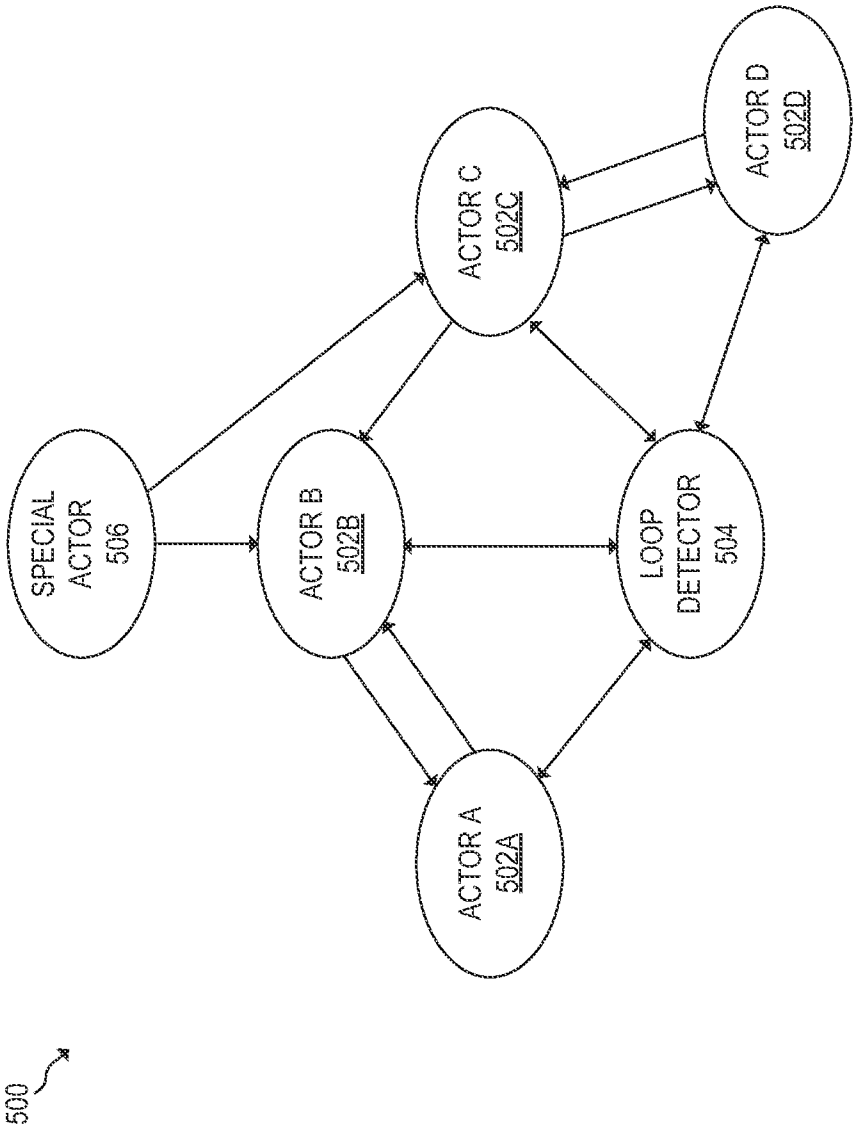
500
SPECIAL ACTOR 506
ACTOR B 502B
ACTOR C 502C
ACTOR D 502D
ACTOR A 502A
LOOP DETECTOR 504

FIG. 5

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2014/044875

A. CLASSIFICATION OF SUBJECT MATTER

G06F 17/30(2006.01)i

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)
G06F 17/30; G06F 15/16; G06F 7/00; G06F 12/00; G06F 9/46; H02H 3/05

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
Japanese utility models and applications for utility models

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
eKOMPASS(KIPO internal) & Keywords: distributed, application, actor, loop, detector, hierarchy, blocked, notification, referece count, garbage collection, and similar terms.

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 2002-0107880 A1 (DAVID FRANCIS BACON) 08 August 2002 See paragraphs [0010]-[0014], [0031]-[0032], [0077]-[0079], and [0109]; and figures 1-3.	1-15
A	US 2010-0333110 A1 (ZHI DA LUO et al.) 30 December 2010 See paragraphs [0009]-[0012], [0036], and [0061]; and figures 3-4a.	1-15
A	US 2011-0004641 A1 (MICHAEL L. ROBERTS) 06 January 2011 See paragraphs [0016] and [0097]; and figures 2-3.	1-15
A	US 6,611,858 B1 (MURALI ARAVAMUDAN et al.) 26 August 2003 See column 9, line 5 - column 10, line 49; and figures 1-3.	1-15
A	US 6,226,761 B1 (VIKTOR BERSTIS) 01 May 2001 See column 2, lines 30-64; column 4, lines 17-49; and figures 1-2.	1-15

☐ Further documents are listed in the continuation of Box C.

☒ See patent family annex.

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"E" earlier application or patent but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed
"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents,such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
26 February 2015 (26.02.2015)	**26 February 2015 (26.02.2015)**

Name and mailing address of the ISA/KR
International Application Division
Korean Intellectual Property Office
189 Cheongsa-ro, Seo-gu, Daejeon Metropolitan City, 302-701,
Republic of Korea
Facsimile No. ++82 42 472 3473

Authorized officer
NHO, Ji Myong
Telephone No. +82-42-481-8528

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.
PCT/US2014/044875

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2002-0107880 A1	08/08/2002	US 6879991 B2	12/04/2005
US 2010-0333110 A1	30/12/2010	CN 101937365 A	05/01/2011
		CN 101937365 B	15/05/2013
		US 2012-0198460 A1	02/08/2012
		US 8661450 B2	25/02/2014
US 2011-0004641 A1	06/01/2011	US 8200718 B2	12/06/2012
US 6611858 B1	26/08/2003	CA 2323371 A1	05/05/2001
		EP 1104897 A2	06/06/2001
		JP 2001-188704A	10/07/2001
US 6226761 B1	01/05/2001	None	