

CLAIMS

1. A method for handling a blockchain service, the method comprising:
receiving, by a first application of a consensus node, a service request sent by a client,
wherein the consensus node comprises the first application, a second application, and a database;
determining to-be-sent service data corresponding to the service request;
storing, by the first application, the to-be-sent service data in the database;
when a predetermined condition is satisfied, fetching, by the second application or the first application, the to-be-sent service data from the database; and
sending the to-be-sent service data to a second consensus node for storage.
2. The method according to claim 1, wherein before storing, by the first application, the to-be-sent service data in the database, the method further comprises:
determining, by the first application, that a state of the to-be-sent service data is a "to be sent" state; and
storing the service data and the "to be sent" state of the service data in the database.
3. The method according to claim 1, wherein satisfying the predetermined condition comprises:
when a specified moment arrives, determining that the predetermined condition is satisfied.
4. The method according to claim 1, wherein fetching the to-be-sent service data from the database comprises:
fetching a specified amount of to-be-sent service data from the database.
5. The method according to claim 1, further comprising:
updating a state of sent service data in the database to a "sent" state.
6. The method according to claim 1, further comprising:

receiving, by the first application, consensus data, and storing the consensus data in the first database;

in response to fetching, by the second application, the consensus data from the first database, performing consensus verification;

adding, by the second application, a verification result to the consensus data;

storing the verification result in the first database; and

sending the verification result to the second consensus node to continue performing the consensus verification.

7. The method according to claim 6, wherein receiving, by the first application, the consensus data comprises:

receiving, by the first application, the consensus data sent by the second consensus node; or

receiving, by the first application, the verification result sent by the second consensus node, and using the verification result as the consensus data.

8. The method according to claim 6, wherein fetching, by the second application, the consensus data from the first database comprises: fetching, by the second application, the consensus data from the first database and service data in a "sent" state in the first database.

9. The method according to claim 8, wherein performing consensus verification comprises:

performing the consensus verification on the consensus data based on the service data in the "sent" state, and determining the verification result.

10. The method according to claim 6, wherein the second consensus node comprises a third application, a fourth application, and a second database.

11. The method according to claim 10, wherein before receiving, by the first application, consensus data, the method further comprises:

determining, by the third application, the consensus data based on service data in a "sent" state in the second database, and storing the consensus data in the second database; and

fetching, by the fourth application, the consensus data from the second database, and sending the consensus data to consensus nodes other than the second consensus node.

12. The method according to claim 6, wherein performing the consensus verification is based on a hash value and a content of the service data.

13. The method according to claim 6, wherein the consensus verification comprises a Practical Byzantine Fault Tolerance algorithm.

14. The method according to claim 1, wherein the to-be-sent service data comprises at least one of a service attribute, a service identifier, a data version, a time stamp, service information, and a state identifier.

15. An apparatus for handling a blockchain service, the apparatus comprising a plurality of modules configured to perform the method of any one of claims 1 to 14.