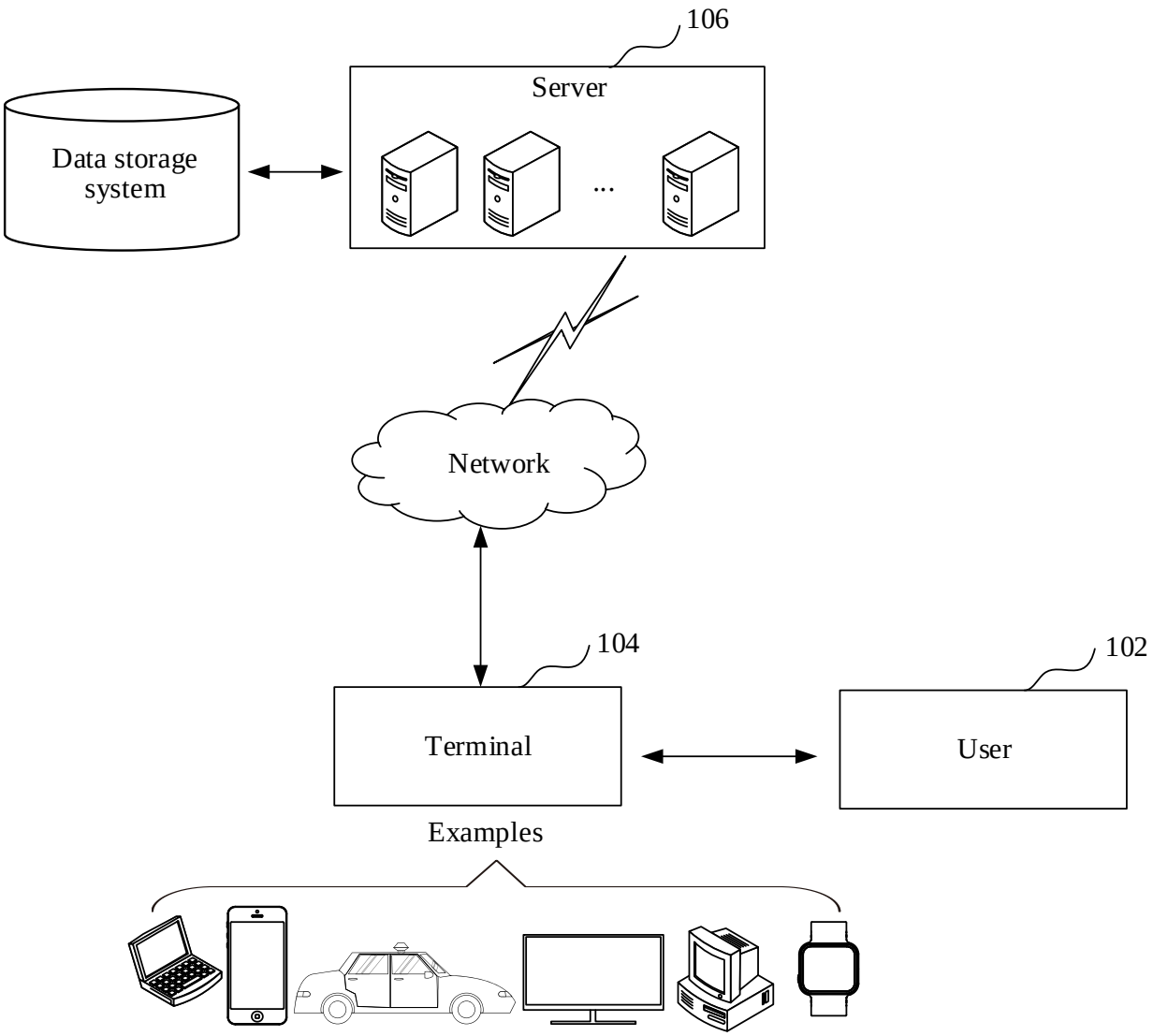

106
Server
Data storage system
...
Network
104
102
Terminal
User
Examples

FIG. 1

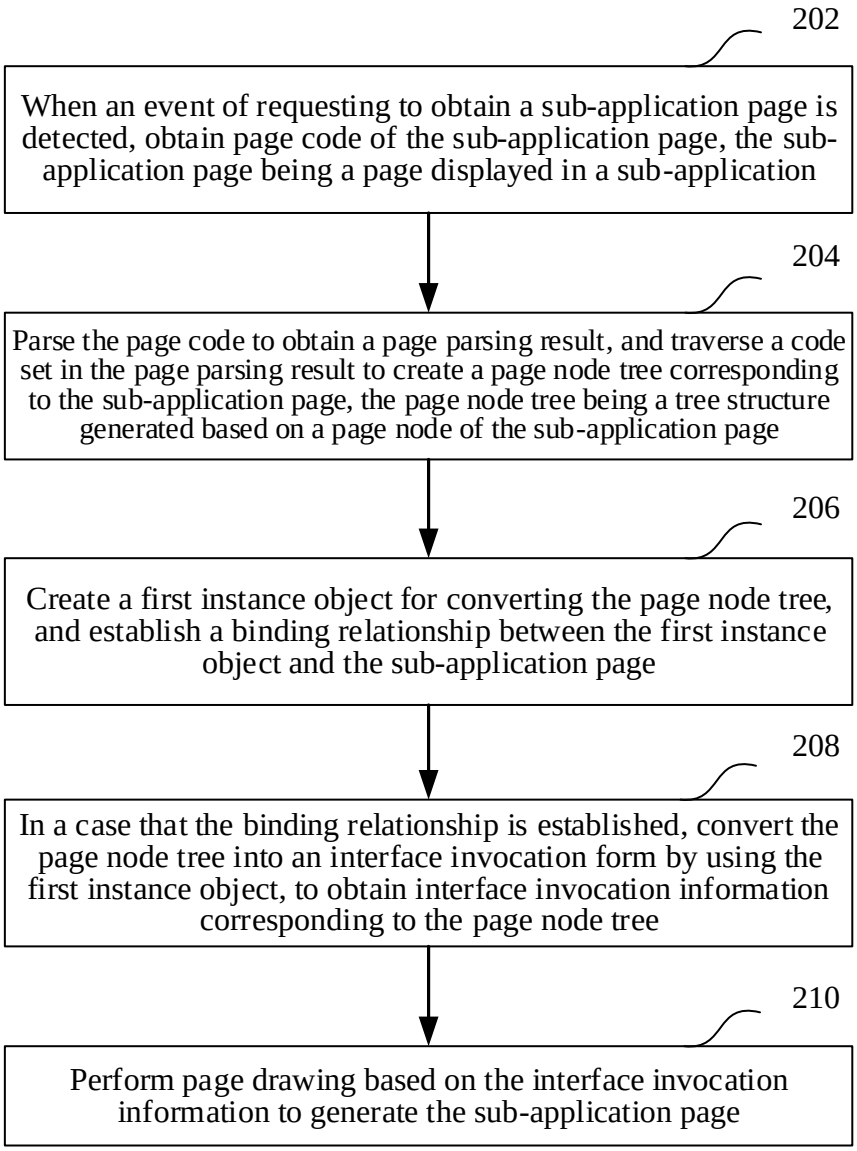

202
When an event of requesting to obtain a sub-application page is detected, obtain page code of the sub-application page, the sub-application page being a page displayed in a sub-application
204
Parse the page code to obtain a page parsing result, and traverse a code set in the page parsing result to create a page node tree corresponding to the sub-application page, the page node tree being a tree structure generated based on a page node of the sub-application page
206
Create a first instance object for converting the page node tree, and establish a binding relationship between the first instance object and the sub-application page
208
In a case that the binding relationship is established, convert the page node tree into an interface invocation form by using the first instance object, to obtain interface invocation information corresponding to the page node tree
210
Perform page drawing based on the interface invocation information to generate the sub-application page

FIG. 2

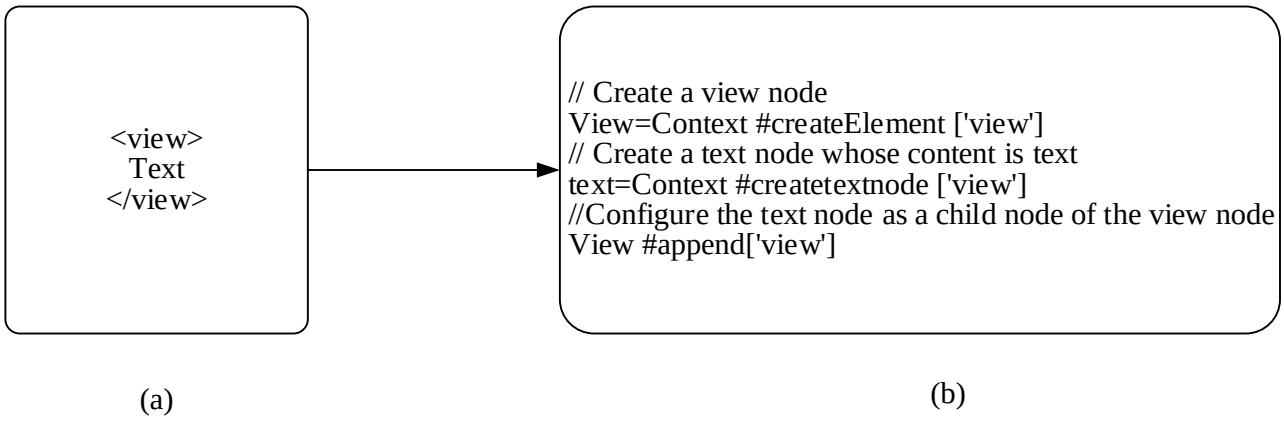

<view>
Text
</view>
// Create a view node
View=Context #createElement ['view']
// Create a text node whose content is text
text=Context #createtextnode ['view']
//Configure the text node as a child node of the view node
View #append['view']
(a)
(b)

FIG. 3

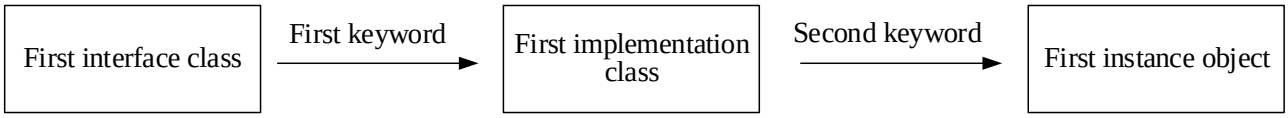


FIG. 4

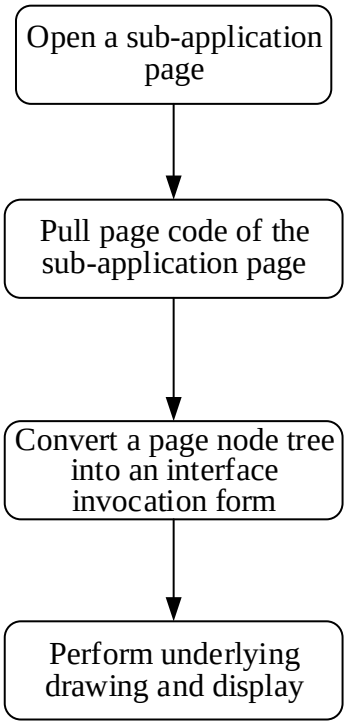


FIG. 5

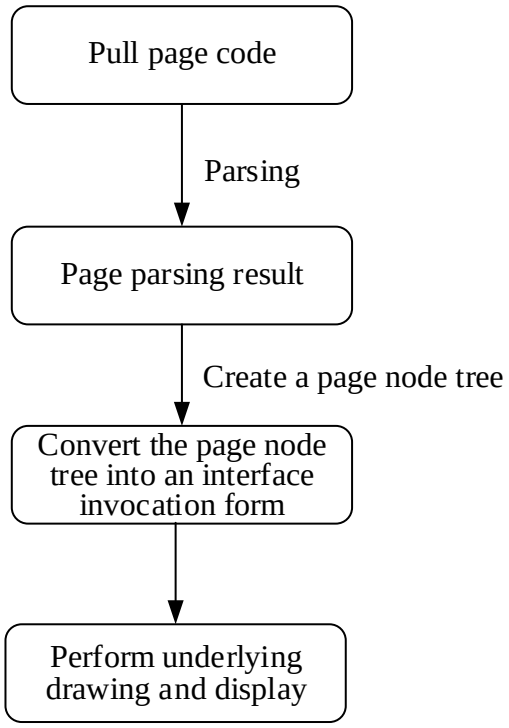


FIG. 6

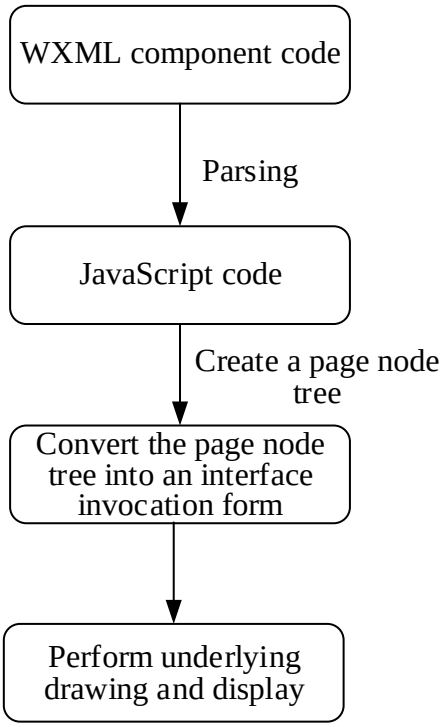

WXML component code
Parsing
JavaScript code
Create a page node tree
Convert the page node tree into an interface invocation form
Perform underlying drawing and display

FIG. 7

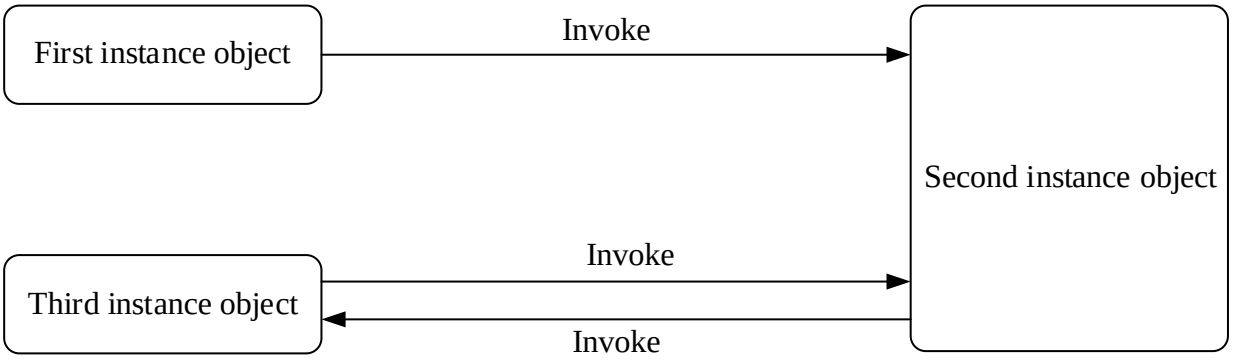

First instance object
Invoke
Second instance object
Third instance object
Invoke
Invoke

FIG. 8

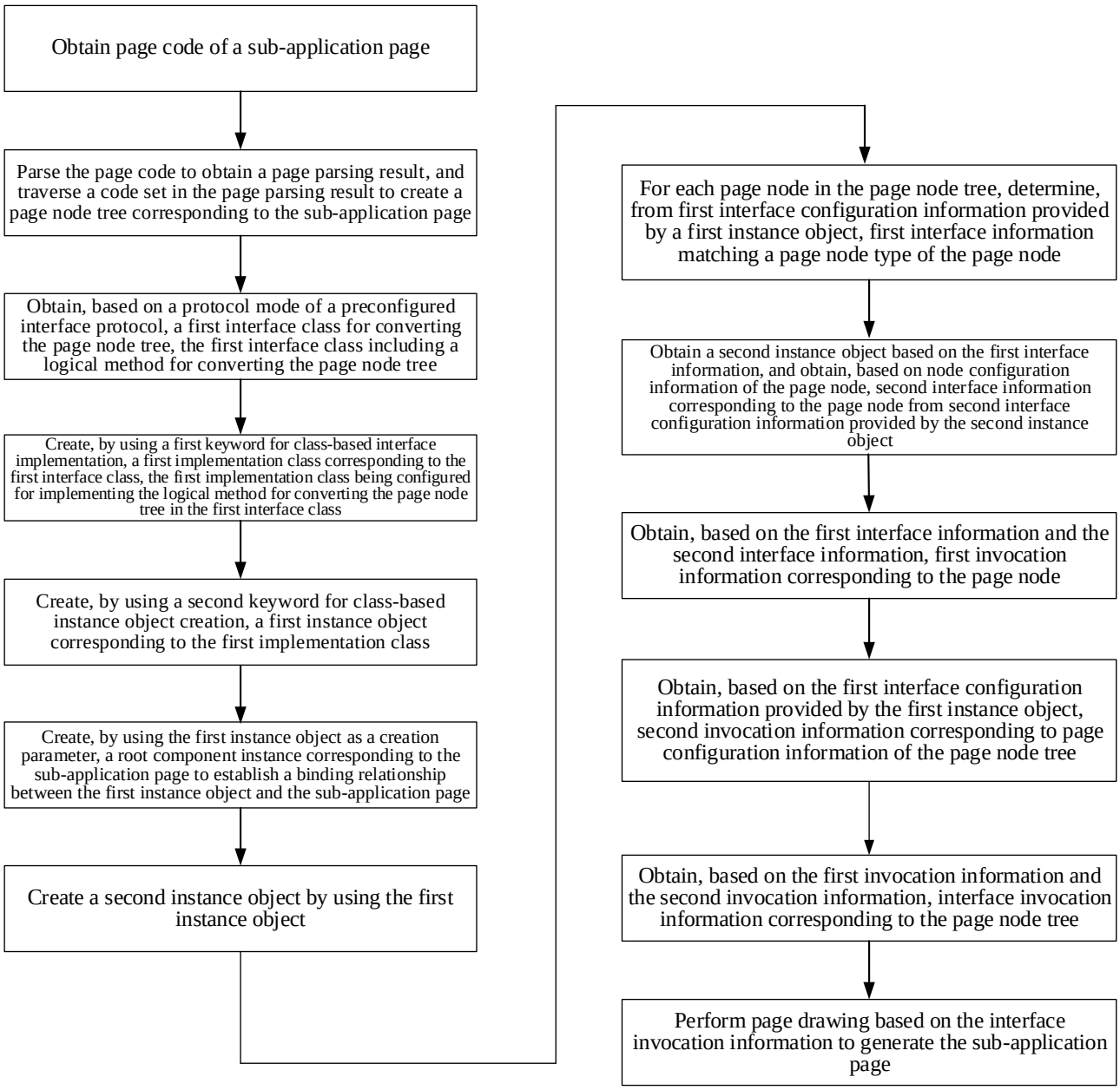

Obtain page code of a sub-application page
Parse the page code to obtain a page parsing result, and traverse a code set in the page parsing result to create a page node tree corresponding to the sub-application page
Obtain, based on a protocol mode of a preconfigured interface protocol, a first interface class for converting the page node tree, the first interface class including a logical method for converting the page node tree
Create, by using a first keyword for class-based interface implementation, a first implementation class corresponding to the first interface class, the first implementation class being configured for implementing the logical method for converting the page node tree in the first interface class
Create, by using a second keyword for class-based instance object creation, a first instance object corresponding to the first implementation class
Create, by using the first instance object as a creation parameter, a root component instance corresponding to the sub-application page to establish a binding relationship between the first instance object and the sub-application page
Create a second instance object by using the first instance object
For each page node in the page node tree, determine, from first interface configuration information provided by a first instance object, first interface information matching a page node type of the page node
Obtain a second instance object based on the first interface information, and obtain, based on node configuration information of the page node, second interface information corresponding to the page node from second interface configuration information provided by the second instance object
Obtain, based on the first interface information and the second interface information, first invocation information corresponding to the page node
Obtain, based on the first interface configuration information provided by the first instance object, second invocation information corresponding to page configuration information of the page node tree
Obtain, based on the first invocation information and the second invocation information, interface invocation information corresponding to the page node tree
Perform page drawing based on the interface invocation information to generate the sub-application page

FIG. 9

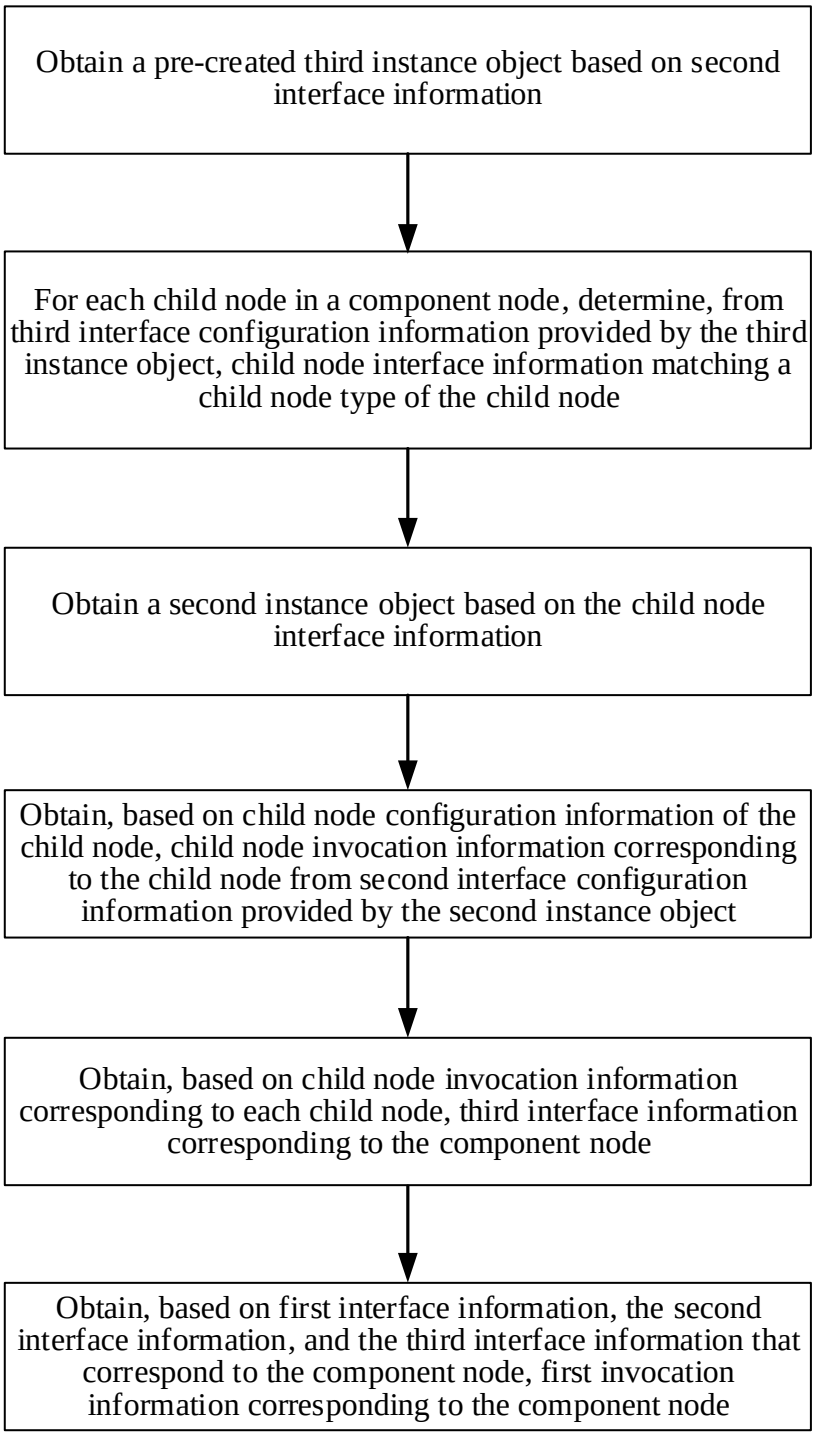

Obtain a pre-created third instance object based on second interface information
For each child node in a component node, determine, from third interface configuration information provided by the third instance object, child node interface information matching a child node type of the child node
Obtain a second instance object based on the child node interface information
Obtain, based on child node configuration information of the child node, child node invocation information corresponding to the child node from second interface configuration information provided by the second instance object
Obtain, based on child node invocation information corresponding to each child node, third interface information corresponding to the component node
Obtain, based on first interface information, the second interface information, and the third interface information that correspond to the component node, first invocation information corresponding to the component node

FIG. 10

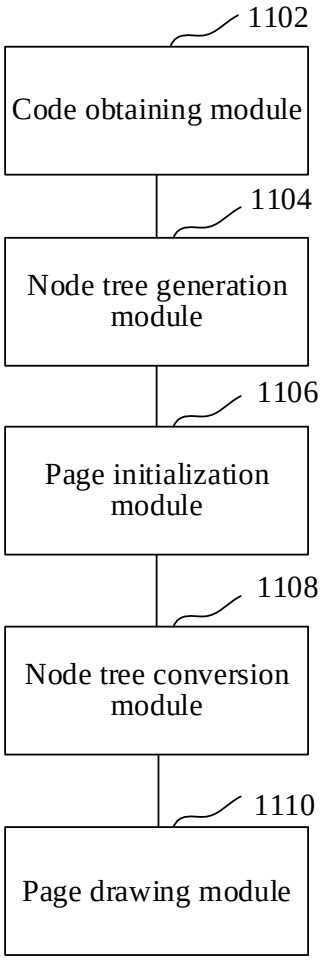

1102
Code obtaining module
1104
Node tree generation module
1106
Page initialization module
1108
Node tree conversion module
1110
Page drawing module

FIG. 11

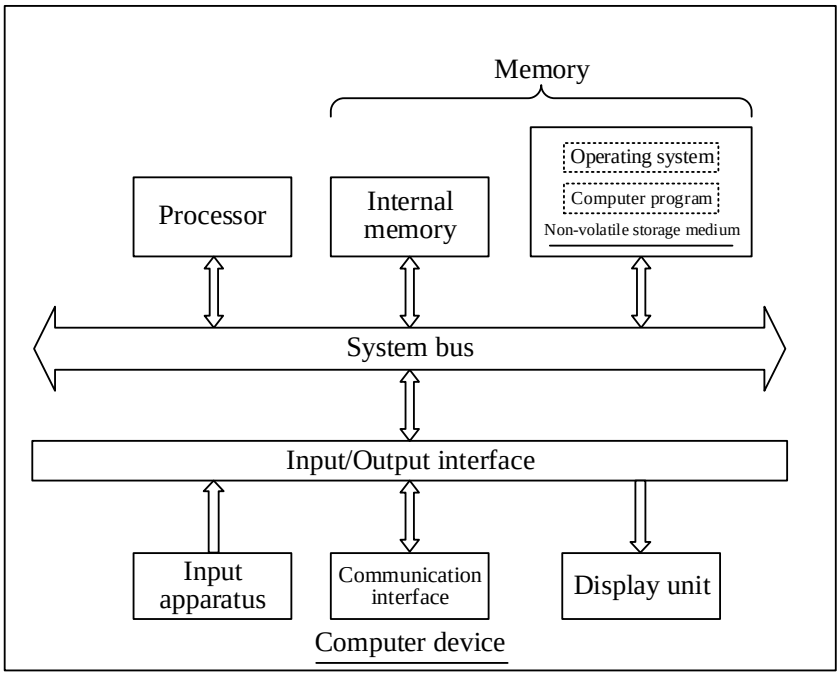

Memory
Operating system
Computer program
Non-volatile storage medium
Processor
Internal memory
System bus
Input/Output interface
Input apparatus
Communication interface
Display unit
Computer device

FIG. 12