Cloud Programming with GreatFree

Bing Li

bing.li@asu.edu

07/01/2017

References

- Author blog

  http://greatfree.lofter.com

- Author email

  bing.li@asu.edu

- Bing Li. Programming Clouds with GreatFree. Book. To be published. Downloadable at the below URL

  https://github.com/greatfree/Programming-Clouds

References

- QQ group: *469024251*
- Source code & samples

  https://github.com/greatfree/Programming-Clouds
- APIs

  http://greatfree.0fees.us
- Audio (Chinese)

  http://www.ximalaya.com/63922737/album/7023894
- Video (English & Chinese; To be published soon)

Outline

- What is GreatFree?
- How does GreatFree define the concept of Cloud Programming?
- Why do we need to program with GreatFree?
- Is it easy to program with GreatFree?

Outline

- Could you show me a sample?
- What applications can we program with GreatFree?
- What APIs and patterns are proposed by GreatFree?
- Are there any other competitors against GreatFree?

Outline

- How do you get such an idea to design the programming environment?
- Have you made any experiments with GreatFree?
- What is the next step of GreatFree?

What is GreatFree?

- At this time, GreatFree is a series of APIs and patterns for distributed programming
- It is implemented with Java SE, 1.7
- It intends to solve one of the most difficult problems for developers, i.e., how to program a distributed system in any scales for any scenarios
- Fortunately, it has achieved the goal

How does GreatFree define the concept of Cloud Programming?

- It is weird many people talk about Cloud Computing
- Do you remember the terms, CORBA, Web Services, Pervasive Computing, Ubiquitous Computing, Autonomous Computing, Grid Computing or Service Computing?
- GreatFree believes that developers that focus on programming do not care about those terms

How does GreatFree define the concept of Cloud Programming?

- They just desire to know what languages, APIs, patterns or tools can help them implement a computing system conveniently
- For that, GreatFree would rather define the concept of Cloud Programming as the programming techniques to build applications over unlimited scales in any distributed computing environments

How does GreatFree define the concept of Cloud Programming?

- It is necessary to clarify the term of programming with other approaches
- In one case, developers do not need to learn the natures of the computing environment they work on
- Instead, they work with a virtualized development environment, such as object-oriented models or even scripts, which are independent of the physical contexts

How does GreatFree define the concept of Cloud Programming?

- No matter what computing systems to be implemented, developers just need to know one particular script language or the object-oriented modeling
- GreatFree does not believe this approach is called programming
- Such an approach is more appropriate to be called the one of scripting

How does GreatFree define the concept of Cloud Programming?

- Another approach is more simple since developers are required only to configure in order to set up an application based on a mature framework
- Configuring is not programming either
- Such phenomena are popular when implementing complicated applications, like the distributed ones, since it saves much effort

How does GreatFree define the concept of Cloud Programming?

- GreatFree believes that programming specifies the procedure in which developers themselves are able to construct a high quality application with their knowledge about the computing environment as well as programming skills
- Ignoring domain knowledge and even programming skills is neither a proper methodology nor a programming environment for developers

How does GreatFree define the concept of Cloud Programming?

- GreatFree feels confident that neither scripting nor configuring can be regarded as programming in accordance with the definition
- Even though they can construct high quality applications, that is the contributions of the underlying frameworks' designers rather than the developers

Why do we need to program with GreatFree?

- For the perspective, GreatFree has to unveil underlying distributed techniques for developers such that they can employ their knowledge when programming
- On the other hand, GreatFree also aims to speed up the development efficiency such that it proposes rich APIs and patterns

Why do we need to program with GreatFree?

- Thus, when you attempt to establish distributed applications in any scales for any computing environments through programming instead of scripting and configuring, GreatFree is your appropriate choice
- First, such an approach is fast because of the APIs and patterns even though you need to code

Why do we need to program with GreatFree?

- Second, it offers you the flexibility to construct any types of distributed applications with your own algorithms
  - When distributed domain knowledge is unveiled, it represents that developers themselves need to apply them when programming
  - You can implement application level algorithms by GreatFree APIs and patterns as well as inject system level one by modifying GreatFree open source

Why do we need to program with GreatFree?

- Third, programming is the greatest interest of Computer Science
  - Scripting and configuring always rob you of such an interest
  - They degenerate your skills such that you always have to rely on others' frameworks
  - You can never propose new systems, solutions or ideas for a new environment by scripting and configuring
  - You cannot fully control your system with scripting and configuring

Why do we need to program with GreatFree?

- Fourth, if the problems in a specific domain cannot be resolved by programming, this must be the disgrace of computer science scientists
  - Until now, such a technology is not available
  - Traditional languages, such as C, C++ and Java, bring heavy workload to developers
  - Some new languages are proposed, such as Google Go and Scala, but they are far from fully-fledged

Is it easy to program with GreatFree?

- Sure, it is easy because of GreatFree APIs and patterns
- If you are lazy guys, you can even program an application with really simplified behaviors, such as copy-paste-replace (CPR)
- If you are proficient developers, you can either CPR or update from the bottom to the top since GreatFree is completely open source

Could you show me a sample?

- OK, I suggest you to read the book if you are not lazy
- Let me show you an example with as limited pages of slides as possible
- The simplest distributed system is the one based on the pattern of C/S
- GreatFree provides developers with such a pattern

Could you show me a sample?

- To program it to be a specific application, you need to accomplish the below tasks in GreatFree
  - Program your messages transmitted between the client and the server by CPR
  - Program the thread at the server side to process the messages by CPR
  - Program the eventer and reader at the client side to send the message by CPR
- I do not believe this is difficult for anybody

Could you show me a sample?

- What does CPR mean in the context of GreatFree programming?
- Since the open source in GreatFree is highly patterned, you can program distributed applications following existing templates exactly
- In another word, when you program, you always have existing highly patterned code to mimic

Could you show me a sample?

```
1   package com.greatfree.testing.message;
2
3   import com.greatfree.multicast.ServerMessage;
4   import com.greatfree.testing.data.Weather;
5
6   /*
7   * The notification contains the data of weather. It is sent to a server such that the data
8   * on the server can be updated. 02/06/2016, Bing Li
9   */
10
11  // Created: 02/06/2016, Bing Li
12  public class WeatherNotification extends ServerMessage
13  {
14      private static final long serialVersionUID = 3555195575233260451L;
15
16      private Weather weather;
17
18      public WeatherNotification(Weather weather)
19      {
20          super(MessageType.WEATHER_NOTIFICATION);
21          this.weather = weather;
22      }
23
24      public Weather getWeather()
25      {
26          return this.weather;
27      }
28  }
```

This is the existing message!

List 3.1 The code of WeatherNotification.java

Could you show me a sample?

```
1     package com.greatfree.testing.message;
2
3     import com.greatfree.multicast.ServerMessage;
4
5     // Created: 03/10/2017, Bing Li
6     public class TestNotification extends ServerMessage
7     {
8         private static final long serialVersionUID = -6936158947185462689L;
9
10        private String testMessage;
11
12        public TestNotification(String testMessage)
13        {
14            super(MessageType.TEST_NOTIFICATION);
15            this.testMessage = testMessage;
16        }
17
18        public String getTestMessage()
19        {
20            return this.testMessage;
21        }
22
23        public void setTestMessage(String testMessage)
24        {
25            this.testMessage = testMessage;
26        }
27    }
```

This is the message you CPR!

List 3.3 The complete code of TestNotification.java

Could you show me a sample?

```
1   package com.greatfree.testing.server;
2
3   import com.greatfree.concurrency.NotificationQueue;
4   import com.greatfree.testing.data.ServerConfig;
5   import com.greatfree.testing.message.WeatherNotification;
6   import com.greatfree.testing.server.resources.WeatherDB;
7
8   /*
9   * The thread implements following the pattern of notification queue. It receives a notification that contains the
10  * weather information to set the weather instance on the server. 02/11/2016, Bing Li
11  */
12
13  // Created: 02/10/2016, Bing Li
14  public class SetWeatherThread extends NotificationQueue<WeatherNotification>
15  {
16      /*
17      * Initialize the thread. 02/11/2016, Bing Li
18      */
19      public SetWeatherThread(int taskSize)
20      {
21          super(taskSize);
22      }
23
```

This is the existing thread!

Could you show me a sample?

```
24      /*
25      * This is the kernel of the notification pattern that sets the weather instance
26      * concurrently. 02/11/2016, Bing Li
27      */
28      public void run()
29      {
30          // Declare an instance of WeatherNotification. 02/11/2016, Bing Li
31          WeatherNotification notification;
32          // The thread always runs until it is shutdown by the NotificationDispatcher. 02/11/2016, Bing Li
33          while (!this.isShutdown())
34          {
35              // Check whether the notification queue is empty. 02/11/2016, Bing Li
36              while (!this.isEmpty())
37              {
38                  try
39                  {
40                      // Dequeue the notification. 02/11/2016, Bing Li
41                      notification = this.getNotification();
42                      // Set the value of the weather. 02/11/2016, Bing Li
43                      WeatherDB.SERVER().setWeather(notification.getWeather());
44                      // Collect the resource kept by the notification. 02/11/2016, Bing Li
45                      this.disposeMessage(notification);
46                  }
47                  catch (InterruptedException e)
48                  {
49                      e.printStackTrace();
50                  }
```

This is the existing thread!

Could you show me a sample?

```
51                }
52                try
53                {
54                    // Wait for a moment after all of the existing notifications are processed. 01/20/2016, Bing Li
55                    this.holdOn(ServerConfig.NOTIFICATION_THREAD_WAIT_TIME);
56                }
57                catch (InterruptedException e)
58                {
59                    e.printStackTrace();
60                }
61            }
62        }
63    }
```

List 3.4 The code of SetWeatherThread.java

This is the existing thread!

Could you show me a sample?

```
1   package com.greatfree.testing.server;
2
3   import com.greatfree.concurrency.NotificationQueue;
4   import com.greatfree.testing.data.ServerConfig;
5   import com.greatfree.testing.message.TestNotification;
6
7   // Created: 03/15/2017, Bing Li
8   public class TestNotificationThread extends NotificationQueue<TestNotification>
9   {
10      /*
11       * Initialize the thread. 02/11/2016, Bing Li
12       */
13      public TestNotificationThread(int taskSize)
14      {
15          super(taskSize);
16      }
17
```

This is the thread you CPR!

Could you show me a sample?

```
18      /*
19       * This is the kernel of the notification pattern that sets the weather instance
20       * concurrently. 02/11/2016, Bing Li
21       */
22      public void run()
23      {
24          TestNotification notification;
25          // The thread always runs until it is shutdown by the NotificationDispatcher. 02/11/2016, Bing Li
26          while (!this.isShutdown())
27          {
28              // Check whether the notification queue is empty. 02/11/2016, Bing Li
29              while (!this.isEmpty())
30              {
31                  try
32                  {
33                      // Dequeue the notification. 02/11/2016, Bing Li
34                      notification = this.getNotification();
35                      // Do something on your notification. 03/16/2017, Bing Li
36                      System.out.println(notification.getTestMessage());
37                      // Collect the resource kept by the notification. 02/11/2016, Bing Li
38                      this.disposeMessage(notification);
39                  }
40                  catch (InterruptedException e)
41                  {
42                      e.printStackTrace();
43                  }
```

This is the thread you CPR!

Could you show me a sample?

```
44                }
45                try
46                {
47                    // Wait for a moment after all of the existing notifications are processed. 01/20/2016, Bing Li
48                    this.holdOn(ServerConfig.NOTIFICATION_THREAD_WAIT_TIME);
49                }
50                catch (InterruptedException e)
51                {
52                    e.printStackTrace();
53                }
54            }
55        }
56    }
```

List 3.5 The code of TestNotificationThread.java after all of classes related to SetWeatherThread are removed

This is the thread you CPR!

Could you show me a sample?

```
38  // Created: 09/20/2014, Bing Li
39  public class MyServerDispatcher extends ServerMessageDispatcher<ServerMessage>
40  {
41      // Declare a notification dispatcher to process the registration
42      // notification concurrently. 11/04/2014, Bing Li
43      private NotificationDispatcher<RegisterClientNotification, RegisterClientThread,
44          RegisterClientThreadCreator> registerClientNotificationDispatcher;
45      // Declare a request dispatcher to respond users sign-up requests
46      // concurrently. 11/04/2014, Bing Li
47      private RequestDispatcher<SignUpRequest, SignUpStream, SignUpResponse,
48          SignUpThread, SignUpThreadCreator> signUpRequestDispatcher;
49      // Declare a notification dispatcher to set the value of Weather when an instance of
50      // WeatherNotification is received. 02/15/2016, Bing Li
51      private NotificationDispatcher<WeatherNotification, SetWeatherThread,
52          SetWeatherThreadCreator> setWeatherNotificationDispatcher;
53      // Declare a request dispatcher to respond an instance of WeatherResponse to
```

This is the location you place your thread at the server side

Could you show me a sample?

```
46      // concurrently. 11/04/2014, Bing Li
47      private RequestDispatcher<SignUpRequest, SignUpStream, SignUpResponse,
48          SignUpThread, SignUpThreadCreator> signUpRequestDispatcher;
49      // Declare a notification dispatcher to set the value of Weather when an instance of
50      // WeatherNotification is received. 02/15/2016, Bing Li
51      private NotificationDispatcher<WeatherNotification, SetWeatherThread,
52          SetWeatherThreadCreator> setWeatherNotificationDispatcher;
53      private NotificationDispatcher<TestNotification, TestNotificationThread,
54          TestNotificationThreadCreator> testNotificationDispatcher;
55      // Declare a request dispatcher to respond an instance of WeatherResponse to
56      // the relevant remote client when an instance of WeatherReques is received. 02/15/2016, Bing Li
57      private RequestDispatcher<WeatherRequest, WeatherStream, WeatherResponse,
58          WeatherThread, WeatherThreadCreator> weatherRequestDispatcher;
59      // Declare a notification dispatcher to deal with instances of InitReadNotification from
```

Your code is placed at the server side by CPR as well!

Could you show me a sample?

```
20  /*
21   * The class is an example that applies SynchRemoteEventer and AsyncRemoteEventer. 11/05/2014, Bing Li
22   */
23
24  // Created: 11/05/2014, Bing Li
25  public class ClientEventer
26  {
27      // Declare the ip of the remote server. 11/07/2014, Bing Li
28      private String ip;
29      // Declare the port of the remote server. 11/07/2014, Bing Li
30      private int port;
31      // The synchronous eventer to send the online notification. 11/07/2014, Bing Li
32      private SyncRemoteEventer<OnlineNotification> onlineEventer;
33      // The synchronous eventer to send the registering notification. 11/07/2014, Bing Li
34      private SyncRemoteEventer<RegisterClientNotification> registerClientEventer;
35      // The synchronous eventer to send the unregistering notification. 11/07/2014, Bing Li
36      private SyncRemoteEventer<UnregisterClientNotification> unregisterClientEventer;
37      // The asynchronous eventer to send one instance of WeatherNotification to the remote server
38      // to set the value of the weather. 02/15/2016, Bing Li
39      private AsyncRemoteEventer<WeatherNotification> weatherEventer;
40
```

Now the client!

Could you show me a sample?

```
25  // Created: 11/05/2014, Bing Li
26  public class ClientEventer
27  {
28      // Declare the ip of the remote server. 11/07/2014, Bing Li
29      private String ip;
30      // Declare the port of the remote server. 11/07/2014, Bing Li
31      private int port;
32      // The synchronous eventer to send the online notification. 11/07/2014, Bing Li
33      private SyncRemoteEventer<OnlineNotification> onlineEventer;
34      // The synchronous eventer to send the registering notification. 11/07/2014, Bing Li
35      private SyncRemoteEventer<RegisterClientNotification> registerClientEventer;
36      // The synchronous eventer to send the unregistering notification. 11/07/2014, Bing Li
37      private SyncRemoteEventer<UnregisterClientNotification> unregisterClientEventer;
38      // The asynchronous eventer to send one instance of WeatherNotification to the remote server
39      // to set the value of the weather. 02/15/2016, Bing Li
40      private AsyncRemoteEventer<WeatherNotification> weatherEventer;
41
42      private AsyncRemoteEventer<TestNotification> testEventer;
43
```

Now the client is done by CPR!

Could you show me a sample?

- For the limited pages, I cannot show you the details
- Anyway, you can taste that GreatFree must be a distinct and convenient approach
- Whether it will be classic, it depends on whether it gets other competitors
- Fortunately, until now, it is still distinguished

What applications can we program with GreatFree?

- Any distributed applications on any scales
- According to the characters of applications
  - Chatting
  - E-commerce
  - Video
  - Storage
  - Search
  - Social networks

What applications can we program with GreatFree?

- From the protocol's point of view
  - The messaging system
  - The streaming system
- Based on the volume of transmitted data
  - Heavyweight data systems
  - Lightweight data systems

What applications can we program with GreatFree?

- With respect to the distributed models
  - The client/server (C/S) one
  - The peer-to-peer (P2P) one
- In accordance with the distributed topology
  - The centralized one
  - The decentralized one

What applications can we program with GreatFree?

- From the scale's perspective
  - The small scale one
  - The large scale one
- Over the Internet
  - Stable systems
  - Churning systems

What applications can we program with GreatFree?

- According to the difference degree of the computing power or behaviors within a particular system
  - The homogeneous one
  - The heterogeneous one
- Taking into account the social issues
  - The system dominated by social behaviors
  - The system controlled by machine algorithms only

What applications can we program with GreatFree?

- In an extreme case, those properties can be merged together into one particular system
- For example, the one called, the globe-scale heterogeneous socialized distributed system
- I believe GreatFree can handle all of them by programming such that GreatFree is a generic programming tool for the domain of distributed systems

What applications can we program with GreatFree?

- GreatFree believes that the following issues are crucial when programming distributed systems
  - Distributed concurrency
  - Distributed modeling
  - Distributed caching
  - Distributed multicasting
  - Distributed clustering
- So the APIs and the patterns need to cover them

What APIs and patterns are proposed by GreatFree?

- GreatFree APIs
  - Communication
  - Serialization
  - Asynchronous & synchronous programming (concurrency programming)
  - Resource management
  - Distributed modeling

What APIs and patterns are proposed by GreatFree?

- GreatFree APIs (continued)
  - Distributed caching
  - Distributed eventing
  - Distributed requesting/responding
  - Distributed multicasting
  - Distributed clustering

What APIs and patterns are proposed by GreatFree?

- It needs to emphasize that GreatFree patterns are really code-level ones rather than an abstract model developers have to transform them into their real world systems
- Conventionally, code-level patterns are called idioms, which are the code that can be mimic in a straightforward manner, i.e., CPR

What APIs and patterns are proposed by GreatFree?

- The distributed component idioms
  - The client
  - The server
  - The peer
  - The client on a cluster
  - The server on a cluster
  - The peer on a cluster

What APIs and patterns are proposed by GreatFree?

- The distributed model idioms
  - The client/server model
  - The peer-to-peer model
  - The client/server chatting model
  - The peer-to-peer chatting model
  - The cluster model
  - The client/server model on clusters
  - The peer-to-peer model on clusters
  - The client/server chatting model on clusters
  - The peer-to-peer chatting model on clusters

What APIs and patterns are proposed by GreatFree?

- The server side idioms
  - SP – The starting point
  - MS – The main server
  - SD – The server dispatcher
  - RD – The request dispatcher
  - ND – The notification dispatcher
  - BND – The bound notification dispatcher
  - BRD – the bound request dispatcher

What APIs and patterns are proposed by GreatFree?

- The client side idioms
  - RR – The remote reader
  - RE – The remote eventer
  - CR – The cluster root
  - MN – The multicasting notifier
  - MR – The multicasting reader
  - CC – The cluster child
  - CBN – The child broadcasting notifier
  - CBR – The child broadcasting reader

What APIs and patterns are proposed by GreatFree?

- In addition, although GreatFree provides many patterns, through which developers can CPR only if the the infrastructure meets their requirements, it does not cover all of them
- You can see the complexity of distributed application from the previous topic
- Fortunately, developers can program with existing APIs and patterns to generate new APIs and patterns that are compatible with the new environments

Are there any other competitors against GreatFree?

- Until now, some programming languages are proposed to solve the distributed programming issues
  - Google Go
  - Scala
- However, their solutions are far from GreatFree
  - Go's main contribution is the concurrency
  - Scala is scalable to be extended

How do you get such an idea to design the programming environment?

- I had no any plan to do that
- I decided to program the most complicated distributed system as mentioned previously, the globe-scale heterogeneous socialized distributed system years ago
- I believed that that was the alias of the Internet
- I got some fundamental new designs for the system

How do you get such an idea to design the programming environment?

- To program such a system, I had to inject my new solutions such that I could not establish the system with the support of existing frameworks by scripting and configuring
- That means I had to program with generic programming languages to handle everything in the complicated computing environment

How do you get such an idea to design the programming environment?

- After a long term to work on that, I got the system almost done
- Meanwhile, the APIs and patterns were proposed during the long procedure
- So, it is possible to program any distributed systems with such a tool

Have you made any experiments with GreatFree?

- Yes, since I need to program the globe-scale heterogeneous socialized distributed system and the APIs and patterns were summarized during the procedure, the code is robust
- I always program with them since I have no other choices

Have you made any experiments with GreatFree?

- So it is tested
- I need to keep doing that after the globe system is launched
- In addition, I have taught the class in my university for two months
- Nearly seventy students learned it
- Some of them practice it by programming systems, such as file sharing

What is the next step of GreatFree?

- Until now, GreatFree is called a programming environment instead of a programming language
- So I decide to push the evolution such that GreatFree can be turned into a new language
- I notice that Scala is really scalable to be extended
- I might try to embed GreatFree into Scala first