B 课堂软件设计文档

组长：于德强

组员：杨华鹏 袁湘敏 吕顺

一、 技术选型表及理由

项目	手机端	电脑端
1.终端支持	Android	PC
1.1 开发语言框架	Java Android Linux	Java C# C++
1.2 响应式布局框架	-	-
1.3 传感器	加速度传感器	-
2.服务器端支持		
2.1 语言	Python	PHP
2.2 Web 框架	Node.js Tornado	Java Node.js Python
2.3 关系数据库	MySQL	MySQL
2.4 数据缓存(非关系)	-	MangoDB
3. 开发平台与工具		
3.1 IDE	Android Studio	Eclipse
3.2 集成与测试	Android Nuex5	Win8 Win7 Winxp
3.3 源代码管理	GitHub	GitHub

经过对项目涉及的关键技术做一些前期研究，我们决定在客户端（手机）采用 Java 开发，在电脑端采用 Java，服务器端采用 PHP 开发，可能存在的风险和理由如下

- 可能存在消息发送拥塞险象
- 可能会造成屏幕显示混乱
- 服务器安全问题
- 服务器负载问题
- 多版本的手机端适配问题
- 数据库（登陆）存储问题

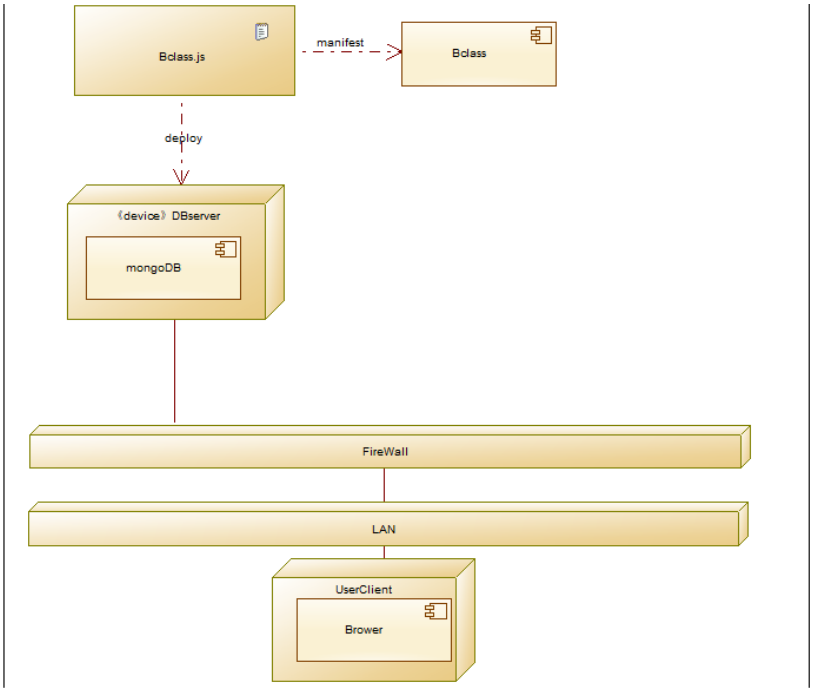
敏捷开发、设计模式对项目变化的贡献

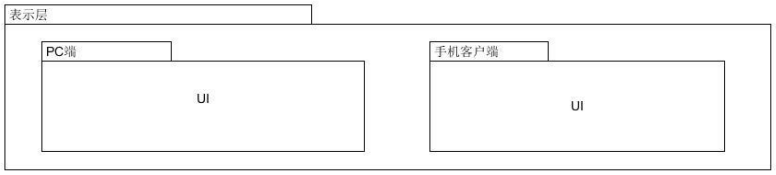
分离了工作，各个成员之间保持良好的沟通和反馈，而且使软件一直处于可用状态

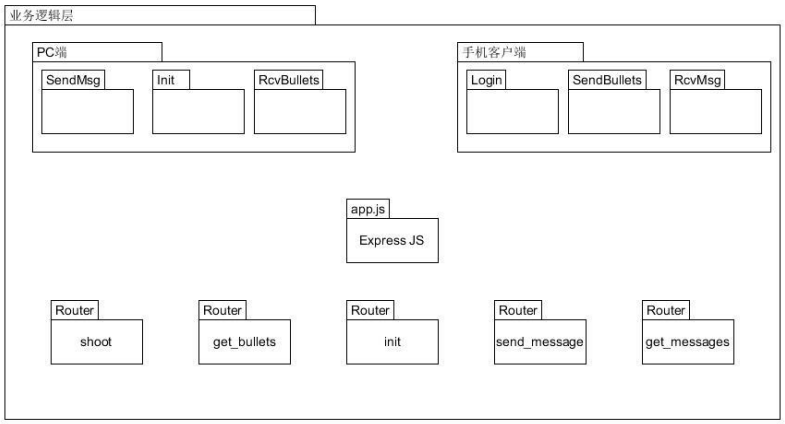
敏捷开发对于变化的适应性很强，有利于功能变更和增加

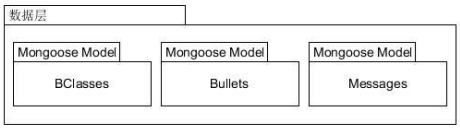
二、 架构设计

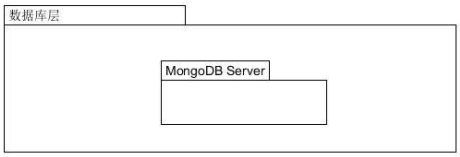
下面是我们设计的部署图和包图的情况，通过手机端发送消息，服务器进行转存，由电脑端进行请求，从而得到相应条数的弹幕。


Bclass.js
manifest
Bclass
deploy
《device》DBserver
mongoDB
FireWall
LAN
UserClient
Brower


表示层
PC端
UI
手机客户端
UI

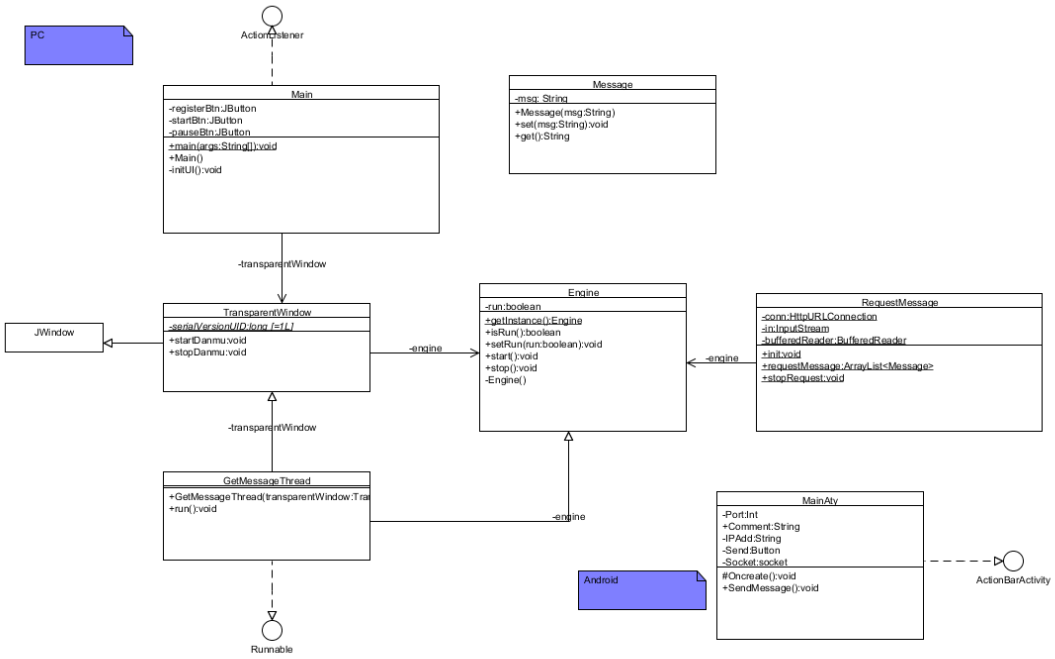

业务逻辑层
PC端
SendMsg
Init
RcvBullets
手机客户端
Login
SendBullets
RcvMsg
app.js
Express JS
Router
shoot
Router
get_bullets
Router
init
Router
send_message
Router
get_messages


数据层
Mongoose Model
BClasses
Mongoose Model
Bullets
Mongoose Model
Messages


数据库层
MongoDB Server

三、 模块划分

下面是手机端和电脑端具体实现的类图



四、 技术细节及对应代码

1. HTTP 连接

对应模块：PCEngine

对应代码：

```
if (engine.isRun()) {
    try {
        URL url = new URL("http://172.18.33.10:3000/init");
        HttpURLConnection conn = (HttpURLConnection) url.openConnection();
        conn.setDoOutput(true);

        conn.setDoInput(true);

        conn.setUseCaches(false);

        conn.setRequestMethod("GET");
        conn.setRequestProperty("Content-type",
                "application/json");

        conn.connect();
```

```
if (engine.isRun()) {
    HttpClient client = new DefaultHttpClient();

    try {
        String url = "http://172.18.33.10:3000/send_message";
        HttpPost post = new HttpPost(url);
        JSONObject obj = new JSONObject();
        obj.put("bcode",code);
        obj.put("message",information);


        StringEntity entity = new StringEntity(obj.toString(), HTTP.UTF_8);

        post.addHeader("Accept","application/json");
        post.addHeader("Content-Type", "application/json");
        post.setEntity(entity);



        HttpParams httpParams = client.getParams();

        HttpResponse response = client.execute(post);
        if(response.getStatusLine().getStatusCode() == 200){

            HttpEntity rentity = response.getEntity();

        }
    } catch (Exception e) {
        //throw new RuntimeException(e);
    }finally {
```

2. 缓存技术

由于应用面对的是用户高并发请求（手机用户频繁发送弹幕）的场景，因此我们对这一用例场景中涉及到的频繁的数据库读写操作（“弹幕”数据）这一部分，采用了 Redis 的解决方案，保证了大量读写的需要。

对应模块：Server

对应代码：

```
var redis = require('redis'),
    client = redis.createClient();

client.on("error", function (err) {
  console.log("Error" + err);
});
```

```
client.multi()
  .lrange(req.query.bcode, 0, parseInt(req.query.limit))
  .exec(function (error, replies) {
    if (error) {
      serverInternalError(res, 'DB FAILURE: QUERY BClass');
    }
    else {
      var resp = new Array();
      for (var i = 0; i < replies[0].length; i++) {
        resp.push(eval('(' + replies[0][i] + ')'));
      }
      res.status(200).json(resp);
    }
    for (var i = 0; i < parseInt(req.query.limit); i++) {
      client.lpop(req.query.bcode);
    }
  });
```

```
var newBullet = {
  bcode: req.body.key,
  //read: false,
  createdAt: (new Date()).toString(),
  important: req.body.important,
  fontSize: req.body.frontsize,
  fontColor: req.body.frontcolor,
  texts: req.body.danmu
};

var FLAG = client.rpush(req.body.key, JSON.stringify(newBullet),
                        redis.print);
```

3. 路由中间件

以路由中间件的形式处理 HTTP 请求

对应模块：Server

对应代码：

```
5  var router = express.Router();
```

```
/**
 * @param  bcode
 * @param  limit
 * @return bullets
 * @coauthor Byron
 */
router.get('/get_bullets', function (req, res, next) {
  console.log(req.query);
  Bclass
    .find({bcode: req.query.bcode})
    .exec(function(error, docs) {
      if (docs == null || docs.length == 0) {
        console.log("Requested B-class not found");
        res.sendStatus(404);
      }
      else {
        Bullet
          .find({bcode: req.query.bcode})
          .where('read').equals(false)
          .limit(req.query.limit)
          .sort('+createdAt')
          .select('createdAt important fontSize fontColor texts')
          .exec(function(error, docs) {
            if (error) {
              serverInternalError(res, 'DB FAILURE: QUERY bullet');
            }
            else {
              for (var i = 0; i < docs.length; i++) {
                Bullet.update(
                  {_id: docs[i]._id},
                  {$set: {read: true}},
                  function (error) {}
                );
              }
              res.status(200).json(docs);
            }
          });
      }
    });
});
```

```
/**
 * @param  bcode
 * @return msgs
 * @coauthor Yu Deqiang
 */
router.get('/get_messages', function (req, res, next) {
  console.log(req.query);
  Bclass
    .find({bcode: req.query.key})
    .exec(function(error, docs) {
      if (docs == null || docs.length == 0) {
        console.log("Requested B-class not found");
        res.sendStatus(404);
      }
      else {
        Message
          .find({bcode: req.query.key})
          .sort('+createdAt')
          .select('createdAt texts')
          .exec(function(error, docs) {
            if (error) {
              serverInternalError(res, 'DB FAILURE: QUERY message');
            }
            else {
              res.status(200).json(docs);
            }
          });
      }
    });
});
```

4. 数据库中间件

对应模块：Server

对应代码：

```
var mongoose = require('mongoose');

var bulletSchema = mongoose.Schema({
  bcode: String,
  read: Boolean,
  createdAt: {type: Date, default: Date.now},
  important: Number,
  fontSize: Number,
  fontColor: Number,
  texts: String
});

var Bullet = mongoose.model('Bullet', bulletSchema);

module.exports = Bullet;
```

4. 其他中间件

这些中间件分别用于处理如 Cookies 解析、HTTP 请求报文解析以及用户会话管理等任务。这些中间件都是 Connect 中间件框架的一部分。在 ExpressJS 开发框架中中，大量使用了中间件。

对应模块：Server

对应代码：

```
    cookieParser = require('cookie-parser'),
    bodyParser = require('body-parser'),
    session = require('express-session'),
```

```
    passport = require('passport');//,
```

```
app.use(bodyParser.urlencoded({ extended: false }));
app.use(cookieParser());
app.use(session({secret: 'supernova', saveUninitialized: true, resave: true}));
app.use(express.static(path.join(__dirname, 'public')));
```

5. 安卓端响应式布局

```
public void setParams() {
    Rect frame = new Rect();
    getWindow().getDecorView().getWindowVisibleDisplayFrame(frame);

    TextView titltText = (TextView)findViewById(R.id.titleText);
    LinearLayout wellayout = (LinearLayout)findViewById(R.id.wellayout);
    LinearLayout buttonLayout = (LinearLayout)findViewById(R.id.buttonLayout);
    TextView infoTitleText = (TextView)findViewById(R.id.infoTitleText);
    TextView infoText = (TextView)findViewById(R.id.infoText);

    RelativeLayout.LayoutParams params = new RelativeLayout.LayoutParams(RelativeLayout.LayoutParams.FILL_PARENT,(int)(frame.height()  * 0.1f + 0.5f));
    params.addRule(RelativeLayout.ALIGN_PARENT_TOP);
    RelativeLayout.LayoutParams params1 = new RelativeLayout.LayoutParams(RelativeLayout.LayoutParams.FILL_PARENT,(int)(frame.height()  * 0.2f + 0.5f));
    params1.addRule(RelativeLayout.BELOW,titltText.getId());
    RelativeLayout.LayoutParams params2 = new RelativeLayout.LayoutParams(RelativeLayout.LayoutParams.FILL_PARENT,(int)(frame.height()  * 0.1f + 0.5f));
    params2.addRule(RelativeLayout.BELOW,wellayout.getId());
    RelativeLayout.LayoutParams params3 = new RelativeLayout.LayoutParams(RelativeLayout.LayoutParams.FILL_PARENT,(int)(frame.height()  * 0.1f + 0.5f));
    params3.addRule(RelativeLayout.BELOW, buttonLayout.getId());
    RelativeLayout.LayoutParams params4 = new RelativeLayout.LayoutParams(RelativeLayout.LayoutParams.FILL_PARENT,(int)(frame.height()  * 0.5f + 0.5f));
    params4.addRule(RelativeLayout.ALIGN_PARENT_BOTTOM);
    titltText.setLayoutParams(params);
    wellayout.setLayoutParams(params1);
    buttonLayout.setLayoutParams(params2);
    infoTitleText.setLayoutParams(params3);
    infoText.setLayoutParams(params4);
}
```