



基于大数据平台搭建移动端 APM 实践

王思宇

关于七牛云



国内企业级云服务商

- 围绕海量数据提供创新的云服务PaaS平台，
- 加速产品成功，创造数据价值。

500人的服务团队

- 创立于2011年，总部位于上海张江
- 北京、深圳、广州、成都、杭州、厦门、武汉设有研发团队和办事处

支持超过70万家用户

- 文件数超过2000亿，每日新增文件20亿，
- 间接服务全国超过80%网民

不断颠覆和创新是七牛成功的基石

市场创新的速度远超于人脑的预判



议程

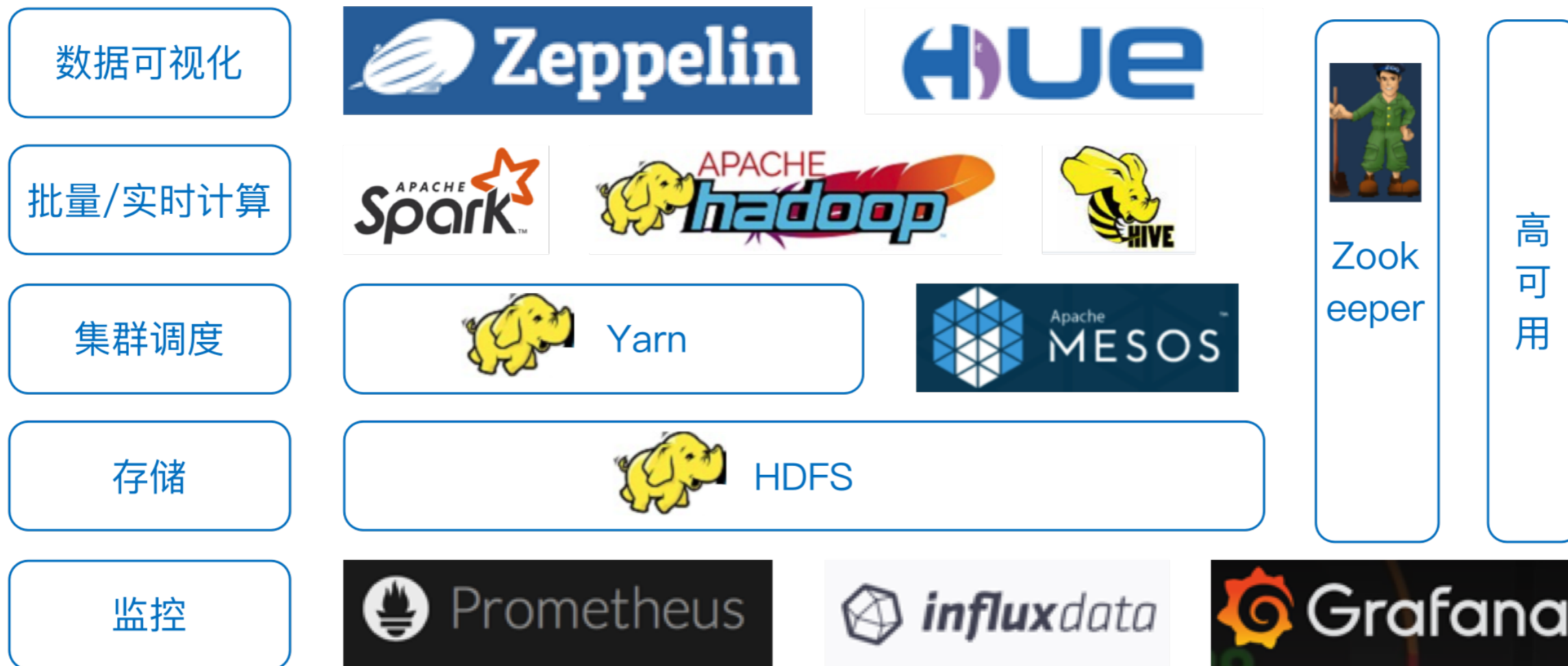
- APM 简介
- 产品实践及遇到的问题
- 展望

APM

Application Performance Management

- **数字化体验监控** — Digital Experience Monitoring
- **应用程序发现、跟踪和诊断** — Application discovery, tracing and diagnostics
- **应用程序分析** — Application Analytics

架构的纠结



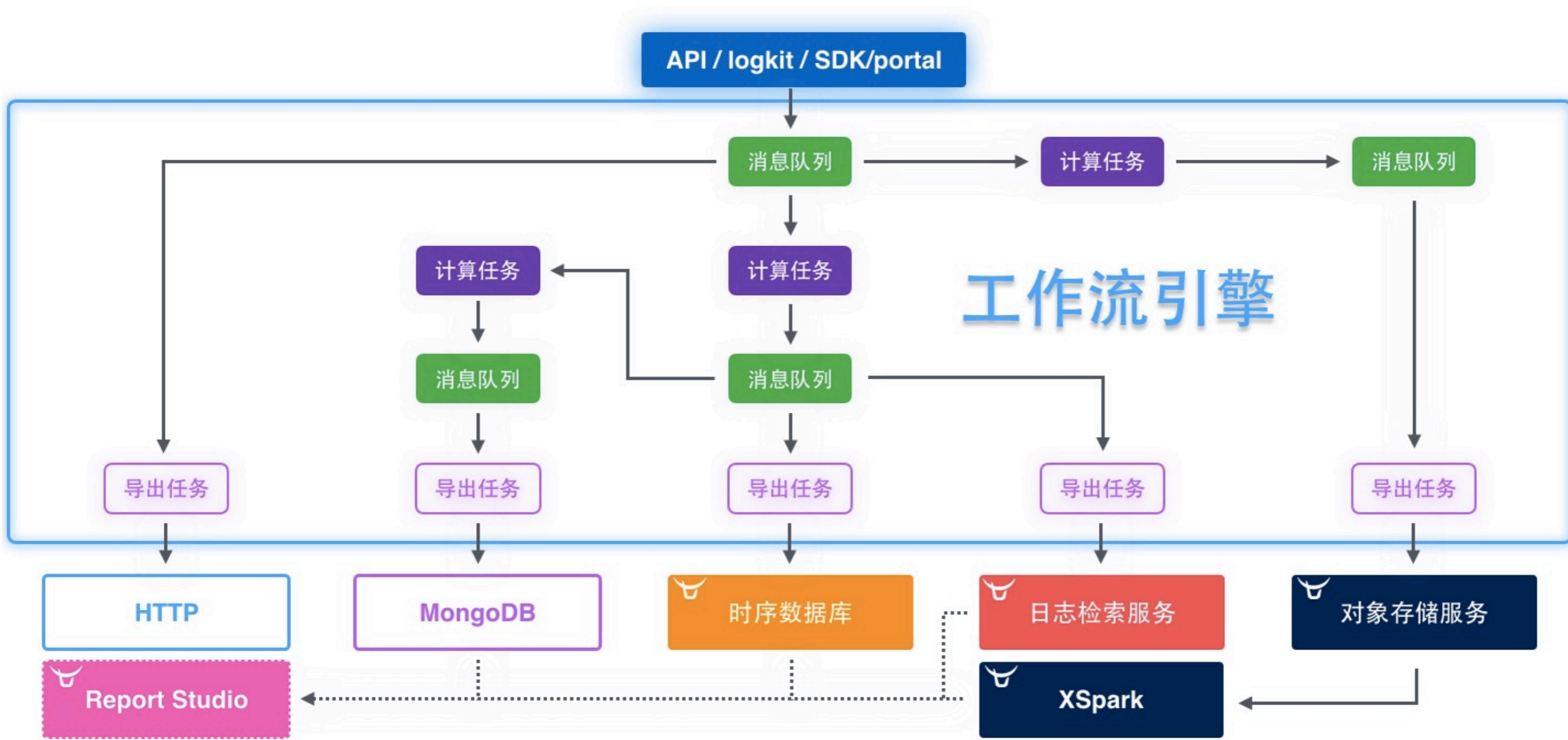
架构的纠结



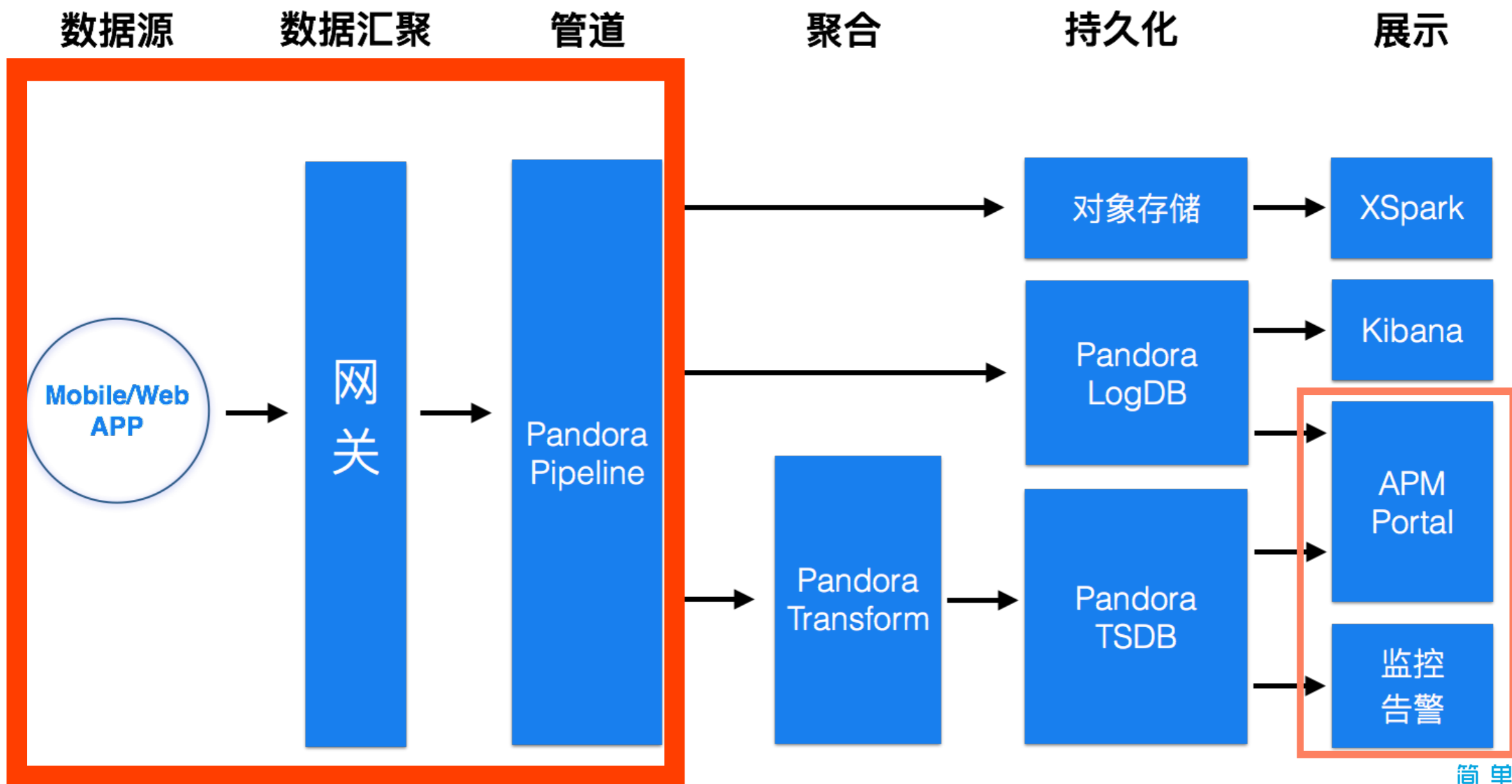
七牛大数据平台 — Pandora

- 从存储到数据可视化，全栈的大数据分析产品
- 用户使用 Workflow 管理自己的数据流，无需大数据背景
- Workflow 实现可视化数据流监控，降低运维成本
- 集成优秀社区组件，优化并做到更好

Pandora架构图



总体架构



监控哪些指标？



移动APP常见痛点



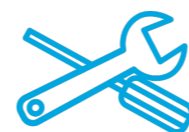
崩溃定位

- 快速暴露、发现、定位、修复
- 统计错误，报警
- 长时间保存
- 机器学习分析



交互响应慢

- ANR 定位
- 界面卡顿
- API 调用慢



网络问题

- 连接超时
- 网络错误
- CDN问题
- 客户端网络信息收集



日志分析

- 客户端上报日志
- 长时间保存
- 机器学习分析

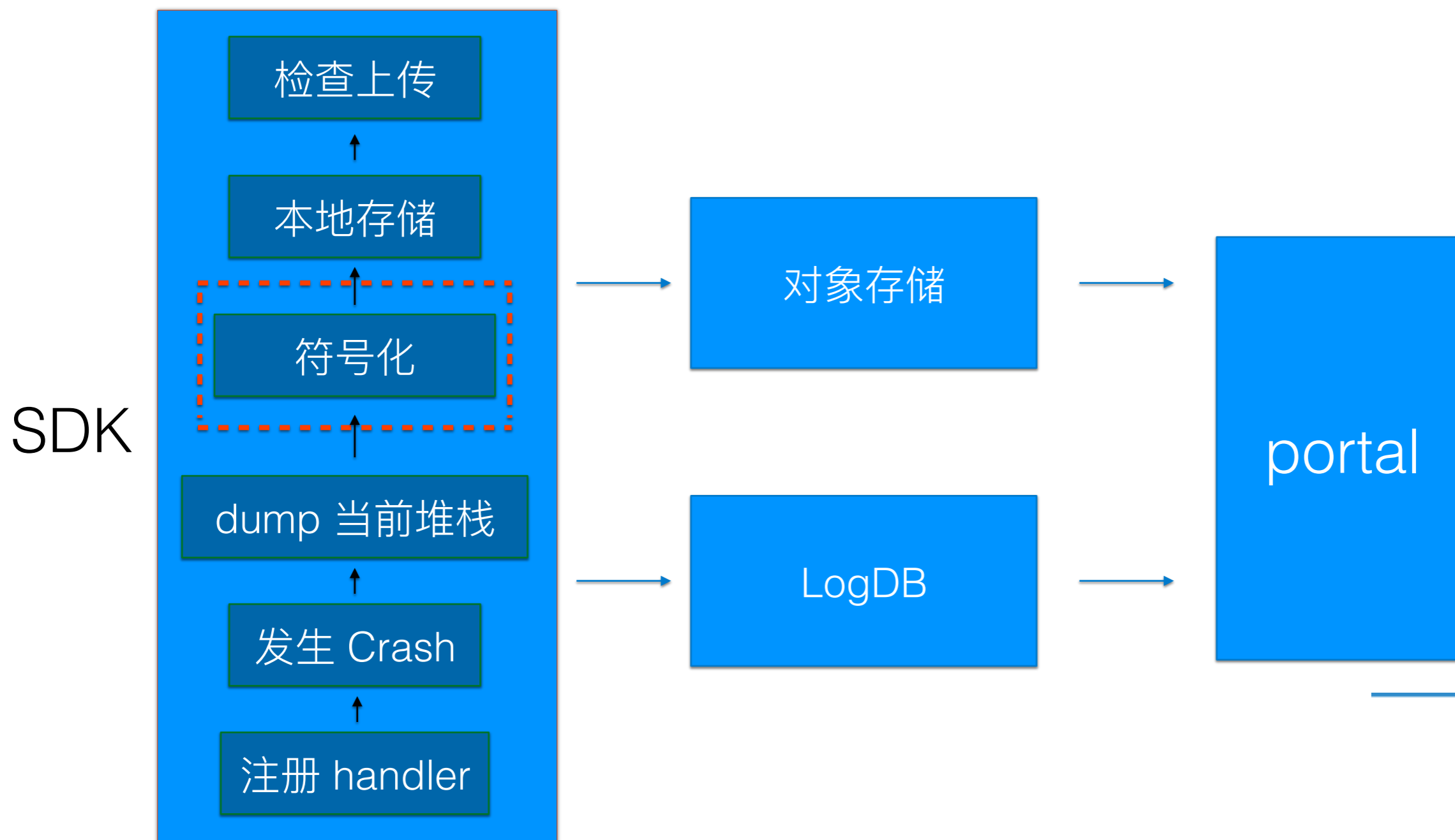
目标

- 与业务解耦，实现无埋点嵌码
- 监控不影响宿主 app 的性能和稳定性
- 不出现数据截断或漏报错报的情况

App 崩溃日志收集

- 支持收集包括 Objective-C, swift, java, c/c++ 在内 crash 信息
- 支持 dump crash 时的调用堆栈以及进行堆栈的符号化
- 本地缓存轮询上报

App 崩溃日志收集



App 崩溃日志收集

```
- (void)startManager {
    [self registerObservers];

    if (!_isSetup) {
        static dispatch_once_t plcrPredicate;
        dispatch_once(&plcrPredicate, ^{
            /* Configure our reporter */

            PLCrashReporterSignalHandlerType signalHandlerType = PLCrashReporterSignalHandlerTypeBSD;

            PLCrashReporterSymbolicationStrategy symbolicationStrategy =
                PLCrashReporterSymbolicationStrategyNone;
            if (self.isOnDeviceSymbolicationEnabled) {
                symbolicationStrategy = PLCrashReporterSymbolicationStrategyAll;
            }

            PREPLCrashReporterConfig *config = [[PREPLCrashReporterConfig alloc] initWithSignalHandlerType:
                signalHandlerType
                symbolicationStrategy:
                symbolicationStrategy];
            self.plCrashReporter = [[PREPLCrashReporter alloc] initWithConfiguration: config];

            // Check if we previously crashed
            if ([self.plCrashReporter hasPendingCrashReport]) {
                _didCrashInLastSession = YES;
                [self handleCrashReport];
            }
        });
    }
}
```


App 崩溃日志收集

```
- (void) handleCrashReport {
    PREDLogVerbose(@"Handling crash report");
    NSError *error = NULL;

    if (!self.plCrashReporter) return;

    // Try loading the crash report
    NSData *crashData = [[NSData alloc] initWithData:[self.plCrashReporter
        loadPendingCrashReportDataAndReturnError: &error]];

    if (crashData == nil) {
        PREDLogError(@"Could not load crash report: %@", error);
    } else {
        NSString *cacheFilename = [NSString stringWithFormat: @"%.0f", [NSDate
            timeIntervalSinceReferenceDate]];
        [crashData writeToFile:[_crashesDir stringByAppendingPathComponent: cacheFilename] atomically:YES];
    }
    [self.plCrashReporter purgePendingCrashReport];
}
```

App 崩溃日志收集

```
- (void)createCrashReportForAppKill {
    NSString *fakeReportUUID = PREDHelper.UUID ?: @"???";
    NSString *fakeReporterKey = PREDHelper.UUID ?: @"???";

    NSString *fakeReportAppBundleIdentifier = PREDHelper.appBundleId;
    NSString *fakeReportDeviceModel = PREDHelper.deviceModel ?: @"Unknown";

    NSString *fakeSignalName = kPREDCrashKillSignal;

    NSMutableString *fakeReportString = [NSMutableString string];

    [fakeReportString appendFormat:@"Incident Identifier: %@\n", fakeReportUUID];
    [fakeReportString appendFormat:@"CrashReporter Key:   %@\n", fakeReporterKey];
    [fakeReportString appendFormat:@"Hardware Model:   %@\n", fakeReportDeviceModel];
    [fakeReportString appendFormat:@"Identifier:      %@\n", fakeReportAppBundleIdentifier];

    NSString *fakeReportAppVersionString = [NSString stringWithFormat:@"%@ (%@)", PREDHelper.appVersion,
                                             PREDHelper.appBuild];

    [fakeReportString appendFormat:@"Version:          %@\n", fakeReportAppVersionString];
    [fakeReportString appendString:@"Code Type:         ARM\n"];
    [fakeReportString appendString:@"\n"];

    NSLocale *enUSPOSIXLocale = [[NSLocale alloc] initWithLocaleIdentifier:@"en_US_POSIX"];
    NSDateFormatter *rfc3339Formatter = [[NSDateFormatter alloc] init];
    [rfc3339Formatter setLocale:enUSPOSIXLocale];
    [rfc3339Formatter setDateFormat:@"yyyy'-'MM'-'dd'T'HH':'mm':'ss'Z'"];
    [rfc3339Formatter setTimeZone:[NSTimeZone timeZoneForSecondsFromGMT:0]];
    NSString *fakeCrashTimestamp = [rfc3339Formatter stringFromDate:[NSDate date]];

    // we use the current date, since we don't know when the kill actually happened
    [fakeReportString appendFormat:@"Date/Time:        %@\n", fakeCrashTimestamp];
    [fakeReportString appendFormat:@"OS Version:      %@\n", PREDHelper.osVersion];
    [fakeReportString appendString:@"Report Version:  104\n"];
    [fakeReportString appendString:@"\n"];
    [fakeReportString appendFormat:@"Exception Type:  %@\n", fakeSignalName];
    [fakeReportString appendString:@"Exception Codes: 00000020 at 0x8badf00d\n"];
    [fakeReportString appendString:@"\n"];
    [fakeReportString appendString:@"Application Specific Information:\n"];
    [fakeReportString appendString:@"The application did not terminate cleanly but no crash occurred."];
    if (self.didReceiveMemoryWarningInLastSession) {
        [fakeReportString appendString:@" The app received at least one Low Memory Warning."];
    }
    [fakeReportString appendString:@"\n\n"];
}
```

App 崩溃日志收集

```
/**
 * Registers the exception handler.
 */
private static void registerHandler(WeakReference<Context> weakContext, boolean ignoreDefaultHandler) {
    if (!TextUtils.isEmpty(AppBean.APP_VERSION) && !TextUtils.isEmpty(AppBean.APP_PACKAGE)) {
        // Get current handler
        Thread.UncaughtExceptionHandler currentHandler = Thread.getDefaultUncaughtExceptionHandler();
        if (currentHandler != null) {
            LogUtils.d("Current handler class = " + currentHandler.getClass().getName());
        }

        Thread.setDefaultUncaughtExceptionHandler(new ExceptionHandler(currentHandler, ignoreDefaultHandler));
    } else {
        LogUtils.d("Exception handler not set because version or package is null.");
    }
}
```

App 崩溃日志收集

Crash log 展

时间区间 2017/07/23 ~ 2017/07/29

AppBundleId	AppId	AppName	AppVersion	CrashLogKey	CrashTime
pre-engineering.PreDemObjcDemo	00000002	PreDemObjcDemo	1.0.0	00000002/crash/3761be760faf402b8cac8332dce5c262	2017-07-13
pre-engineering.PreDemObjcDemo	00000002	PreDemObjcDemo	1.0.0	00000002/crash/50025277520ee62c968	

APP 信息 平台信息 设备信息 返回

appBundled	appName	appVersion	sdkId	sdkVersion
qiniu.predem.example	AndroidPreDem	1.0.0	PreDemSDK	1.0.0

崩溃信息

```
at java.lang.VMThread.sleep(Native Method)
at java.lang.Thread.sleep(Thread.java:1013)
at java.lang.Thread.sleep(Thread.java:995)
at qiniu.predem.example.MainActivity.SleepAMinute(MainActivity.java:55)
at qiniu.predem.example.MainActivity.access$300(MainActivity.java:33)
at qiniu.predem.example.MainActivity$7.onClick(MainActivity.java:182)
at android.view.View.performClick(View.java:4438)
at android.view.View$PerformClick.run(View.java:18422)
at android.os.Handler.handleCallback(Handler.java:733)
at android.os.Handler.dispatchMessage(Handler.java:95)
at android.os.Looper.loop(Looper.java:136)
at android.app.ActivityThread.main(ActivityThread.java:5001)
```

LogDB 检索

Http 及 Webview 请求分析 - iOS

- 利用 iOS runtime 的特性代理 NSURLConnection 和 NSURLSession 的请求
- 采集所代理请求的性能信息并上报 Pandora
- 使用 Pandora 计算任务进行聚合并导出到 tsdb
- 相关信息的展示及分析告警

Http 及 Webview 请求分析 - iOS

```
+ (void)enableHTTPDem {
    if (PREDHTTPMonitorSender.isEnabled) {
        return;
    }
    // 可拦截 [NSURLSession defaultSession] 以及 UIWebView 相关的请求
    [NSURLProtocol registerClass:self.class];

    // 拦截自定义生成的 NSURLSession 的请求
    if (![PREDURLSessionSwizzler isSwizzle]) {
        [PREDURLSessionSwizzler load];
    }

    PREDHTTPMonitorSender.enable = YES;
}

+ (void)disableHTTPDem {
    if (!PREDHTTPMonitorSender.isEnabled) {
        return;
    }
    [NSURLProtocol unregisterClass:self.class];
    if ([PREDURLSessionSwizzler isSwizzle]) {
        [PREDURLSessionSwizzler unload];
    }
    PREDHTTPMonitorSender.enable = NO;
}
```

Http 及 Webview 请求分析 - iOS

```
+ (BOOL)canInitWithRequest:(NSURLRequest *)request {
    if (![request.URL.scheme isEqualToString:@"http"] &&
        ![request.URL.scheme isEqualToString:@"https"]) {
        return NO;
    }

    // SDK主动发送的数据
    if ([NSURLProtocol propertyForKey:@"PREDInternalRequest" inRequest:request] ) {
        return NO;
    }

    return YES;
}
```

Http 及 Webview 请求分析 - iOS

```
+ (NSURLRequest *)canonicalRequestForRequest:(NSURLRequest *)request {
    NSMutableURLRequest *mutableRequest = [request mutableCopy];
    if (mutableRequest.URL.path.length == 0) {
        mutableRequest.URL = [NSURL URLWithString:@"/" relativeToURL:mutableRequest.URL];
    }
    [NSURLProtocol setProperty:@YES
                 forKey:@"PREDInternalRequest"
                 inRequest:mutableRequest];
    [NSURLProtocol setProperty:mutableRequest.URL
                 forKey:@"PREDOriiginalURL"
                 inRequest:mutableRequest];
    if ([request.URL.scheme isEqualToString:@"http"]) {
        NSMutableArray *resolvers = [[NSMutableArray alloc] init];
        [resolvers addObject:[QNResolver systemResolver]];
        [resolvers addObject:[QNResolver alloc] initWithAddress:DNSPodsHost];
        QNDnsManager *dns = [[QNDnsManager alloc] initWithResolvers:resolvers networkInfo:[QNNetworkInfo normal]];
        NSTimeInterval dnsStartTime = [[NSDate date] timeIntervalSince1970];
        NSURL *replacedURL = [dns queryAndReplaceWithIP:mutableRequest.URL];
        NSTimeInterval dnsEndTime = [[NSDate date] timeIntervalSince1970];
        NSError *err;
        NSRegularExpression *regex = [NSRegularExpression regularExpressionWithPattern:@"^[0-9]+\.[0-9]+\.[0-9]+\."
                                     options:0 error:&err];
        NSInteger number = [regex numberOfMatchesInString:replacedURL.host options:0 range:NSMakeRange(0,
        [replacedURL.host length])];
        if (number == 0) {
            return mutableRequest;
        }
        [NSURLProtocol setProperty:@((dnsEndTime - dnsStartTime)*1000)
                         forKey:@"PREDDNSTime"
                         inRequest:mutableRequest];
        [NSURLProtocol setProperty:replacedURL.host
                         forKey:@"PREDHostIP"
                         inRequest:mutableRequest];
        [mutableRequest setValue:request.URL.host forKey:@"Host"];
        mutableRequest.URL = replacedURL;
    }
    return mutableRequest;
}
```


Http 及 Webview 请求分析 - iOS

```
- (void)startLoading {
    NSURLSessionConfiguration *sessionConfig = NSURLSessionConfiguration.ephemeralSessionConfiguration;
    NSURLSession *session = [NSURLSession sessionWithConfiguration:sessionConfig delegate:self
        delegateQueue:[NSOperationQueue new]];
    self.task = [session dataTaskWithRequest:self.request];
    [self.task resume];

    HTTPMonitorModel = [[PREDHTTPMonitorModel alloc] init];
    NSURL *originURL = [NSURLProtocol propertyForKey:@"PREDOriginalURL" inRequest:self.request];
    HTTPMonitorModel.domain = originURL.host;
    HTTPMonitorModel.path = originURL.path;
    HTTPMonitorModel.method = self.request.HTTPMethod;
    HTTPMonitorModel.hostIP = [NSURLProtocol propertyForKey:@"PREDHostIP" inRequest:self.request];
    HTTPMonitorModel.startTimestamp = [[NSDate date] timeIntervalSince1970] * 1000;
    HTTPMonitorModel.endTimestamp = [[NSDate date] timeIntervalSince1970] * 1000;
    HTTPMonitorModel.DNSTime = [NSURLProtocol propertyForKey:@"PREDDNSTime" inRequest:self.request]
        unsignedIntegerValue];
}
```

Http 及 Webview 请求分析 - Android

- 当宿主 app 使用 HttpURLConnection、OkHttp2、Okhttp3 或 Webview 产生 HTTP 请求时，SDK 使用 AOP 方式，在编译的时候无侵入的进行相关信息的采集与上报。
- 使用 Pandora 计算任务进行聚合并导出到 tsdb
- 相关信息的展示及分析告警

Http 及 Webview 请求分析 - Android

```
buildscript {
    repositories {
        jcenter()
    }
    dependencies {
        classpath 'com.android.tools.build:gradle:2.3.3'
        classpath 'com.hujiang.aspectjx:gradle-android-plugin-aspectjx:1.0.10'
        classpath 'org.aspectj:aspectjtools:1.8.9'

        // NOTE: Do not place your application dependencies here; they belong
        // in the individual module build.gradle files
    }
}
```

```
apply plugin: 'com.android.application'
apply plugin: 'android-aspectjx'
```

Http 及 Webview 请求分析 - Android



```
@Around("call(* java.net.URL+.openConnection(..)")")
public Object onHttpURLConnection(ProceedingJoinPoint joinPoint) throws Throwable {
    if (!Configuration.httpMonitorEnable || joinPoint.getArgs().length != 0) {
        return joinPoint.proceed();
    }

    try {
        if (!joinPoint.getTarget().toString().startsWith("http://") && !joinPoint.getTarget().toString().startsWith("https://")) {
            return joinPoint.proceed();
        }

        URL url = (URL) joinPoint.getTarget();

        //判断是否需要收集url信息
        if (GlobalConfig.isExcludeHost(url.getHost()) || ProbeWebClient.isExcludeIPs(url.getHost())) {
            return joinPoint.proceed();
        }

        HttpURLConnection conn;

        LogBean bean = LogBean.obtain();
        bean.setStartTimestamp(System.currentTimeMillis());
        conn = (HttpURLConnection)joinPoint.proceed();
        // 判断host是否是IP
        Matcher matcher = MatcherUtil.IP_Pattern.matcher(url.getHost());
        if (matcher.find()) {
            bean.setHostIP(url.getHost());
            bean.setDnsTime(0);
            bean.setDomain(url.getHost());
            bean.setPath(url.getPath());
        } else {
            String hostIp = null;
            if (Configuration.dnsEnable) {
                long stime = System.currentTimeMillis();
                try {
                    bean.setHostIP(InetAddress.getByName(url.getHost()).getHostAddress());
                } catch (UnknownHostException e) {
```

Http 及 Webview 请求分析 - Android



```
@Around("call(* java.net.URLConnection+.getInputStream(..)")
public Object onHttpURLConnectionInput(ProceedingJoinPoint joinPoint) throws Throwable {
    if (!Configuration.httpMonitorEnable) {
        return joinPoint.proceed();
    }
    try {
        URLConnection conn = (URLConnection) joinPoint.getTarget();

        URL url;
        if (conn instanceof HttpURLConnection) {
            url = ((HttpURLConnection) joinPoint.getTarget()).getURL();
        } else {
            return joinPoint.proceed();
        }

        if (GlobalConfig.isExcludeHost(url.getHost()) || ProbeWebClient.isExcludeIPs(url.getHost())) {
            return joinPoint.proceed();
        }

        try {
            synchronized (reportMap) {
                int key = conn.hashCode();
                if (reportMap.containsKey(key)) {
                    LogBean urlTraceRecord = reportMap.get(key);
                    reportMap.remove(key);
                    if (GlobalConfig.isExcludeHost(urlTraceRecord.getDomain()) || ProbeWebClient.isExcludeIPs(urlTraceRecord.getDomain())) {
                        return joinPoint.proceed();
                    }

                    urlTraceRecord.setResponseTimestamp(System.currentTimeMillis());
                    urlTraceRecord.setStatusCode(((HttpURLConnection) conn).getResponseCode());
                    return ProbeInputStream.obtain((InputStream) joinPoint.proceed(), urlTraceRecord);
                } else {
                    LogBean urlTraceRecord = LogBean.obtain();
                    urlTraceRecord.setDomain(url.getHost());
                    urlTraceRecord.setPath(url.getPath());
                    urlTraceRecord.setMethod(((HttpURLConnection) conn).getRequestMethod());
                    urlTraceRecord.setDnsTime(0);
```

Http 及 Webview 请求分析



UI 卡顿及 ANR 监控

- 在主线程 Looper 或 Runloop 安放探针，监控调度到主线程的单个任务及连续多个任务的执行时间
- 当任务执行事件一定阈值之后会表现为 UI 卡顿，一旦其超过阈值 SDK 将收集系统当前的一些状态以及程序调用的堆栈信息进行上报
- 相关信息的展示、检索及分析告警

UI 卡顿及 ANR 监控

```
- (void)registerObserver {
    CFRunLoopObserverContext context = {0, (__bridge void*)self, NULL, NULL};
    _observer = CFRunLoopObserverCreate(kCFAllocatorDefault,
                                       kCFRunLoopAllActivities,
                                       YES,
                                       0,
                                       &runLoopObserverCallback,
                                       &context);
    CFRunLoopAddObserver(CFRunLoopGetMain(), _observer, kCFRunLoopCommonModes);

    // 创建信号
    _semaphore = dispatch_semaphore_create(0);

    // 在子线程监控时长
    dispatch_async(dispatch_get_global_queue(0, 0), ^{
        while (YES) {
            // 假定连续5次超时50ms认为卡顿(当然也包含了单次超时250ms)
            long st = dispatch_semaphore_wait(_semaphore, dispatch_time(DISPATCH_TIME_NOW, 50*NSEC_PER_MSEC));
            if (st != 0) {
                if (_activity==kCFRunLoopBeforeSources || _activity==kCFRunLoopAfterWaiting) {
                    if (++_countTime < 5)
                        continue;
                    [self sendLagStack];
                }
            }
            _countTime = 0;
        }
    });
}
```


UI 卡顿及 ANR 监控

```
- (void)sendLagStack {
    NSError *err;
    NSData *data = [_reporter generateLiveReportAndReturnError:&err];
    if (err) {
        return;
    }

    PREPLCrashReport *report = [[PREPLCrashReport alloc] initWithData:data error:&err];
    if (err) {
        return;
    }
    if ([PREDCrashReportTextFormatter isReport:report euivalentWith:_lastReport]) {
        return;
    }
    _lastReport = report;
    [self uploadCrashLog:report retryTimes:0];
}
```

UI 卡顿及 ANR 监控

```
public void run() {
    int lastTick;
    while(!stop){
        lastTick = _tick;
        long startTime = System.currentTimeMillis();
        _uiHandler.post(_ticker);
        try {
            Thread.sleep(_timeoutInterval);
        } catch (InterruptedException e) {
            e.printStackTrace();
            return;
        }

        if (_tick == lastTick){
            //产生ANR, 获取stackInfo
            TraceInfo info = new TraceInfo();
            info.mStartTime = startTime;
            info.mEndTime = System.currentTimeMillis();
            info.mLog = stackTraceToString(Looper.getMainLooper().getThread().getStackTrace());
            mList.add(info);
            if (count > 3 && fileName == null){
                fileName = UUID.randomUUID().toString();
                //保存到文件
                FileUtil.getInstance().writeAnrReport(fileName, info);
            }
            count++;
        }else if (_tick != lastTick && fileName != null){
            //恢复 删除文件
            deleteStackTrace(weakContext, fileName);
            fileName = null;
            count = 0;
        }

        //进入后台后停止进度
        if (Functions.isBackground(mContext)){
            stopTrace();
            return;
        }
        if(mList.size()>SIZE){
            mList.remove(0);
        }
    }
}
```

UI 卡顿及 ANR 监控

7NDEM iOS 卡顿监控日志 退出

APP 信息 平台信息 设备信息 返回

appBundled	appName	appVersion	sdkId	sdkVersion	tag
qiniu_predem.example	AndroidPreDem	1.0.0		1.0.0	-

崩溃信息 异常类型

Incident Identifier	72A8D354-C388-47AF-B982-B364ACBA7AA2
CrashReporter Key	PreDemObjcDemo
Hardware Model	iPhone9,2
Process	PreDemObjcDemo [333]
Path	/var/containers/Bundle/Application/DB48868D-4C6D-44A4-AA4F-67ECE014CED8/PreDemObjcDemo.app/PreDemObjcDemo
Identifier	pre-engineering.PreDemObjcDemo
Version	1.0.1_alpha1 (1)
Code Type	ARM-64
Parent Process	??? [1]

堆栈信息

Thread 0:

- 0 libsystem_kernel.dylib 0x000000018a31e314 0x18a2ff000 + 127764
- 1 libsystem_c.dylib 0x000000018a2a6290 0x18a22f000 + 488080
- 2 UIKit 0x00000001911ce7b0 0x191189000 + 284592
- 3 UIKit 0x00000001911ce730 0x191189000 + 284464
- 4 UIKit 0x00000001911b8be4 0x191189000 + 195556
- 5 UIKit 0x00000001911ce01c 0x191189000 + 282652
- 6 UIKit 0x00000001911c8b44 0x191189000 + 281412
- 7 UIKit 0x00000001911c8d8c 0x191189000 + 261516
- 8 UIKit 0x0000000191199858 0x191189000 + 67672
- 9 UIKit 0x0000000191986cb8 0x191189000 + 8379576
- 10 UIKit 0x0000000191980720 0x191189000 + 8353568
- 11 CoreFoundation 0x000000018b2fe278 0x18b221000 + 905848
- 12 CoreFoundation 0x000000018b2fdbc0 0x18b221000 + 904128
- 13 CoreFoundation 0x000000018b2fb7c0 0x18b221000 + 894912
- 14 CoreFoundation 0x000000018b22a048 0x18b221000 + 36936
- 15 GraphicsServices 0x000000018ccad196 0x18cca1000 + 49560
- 16 UIKit 0x0000000191204628 0x191189000 + 505384
- 17 UIKit 0x00000001911f360 0x191189000 + 484192
- 18 PreDemObjcDemo 0x0000000100039600 0x100034000 + 22016

网络诊断信息

- 获取 Ping, Tcp Ping, http request, DNS 上报
- 相关信息的展示、检索及分析告警

网络诊断信息

```
+ (void)diagnose:(NSString *)host
    netClient:(PREDNetworkClient *)client
    complete:(PREDNetDiagCompleteHandler)complete {
    NSString *httpHost;
    if ([host hasPrefix:@"http://"] || [host hasPrefix:@"https://"]) {
        httpHost = host;
        host = [[host componentsSeparatedByString:@"//"] lastObject];
    } else {
        httpHost = [NSString stringWithFormat:@"http://%@", host];
    }
    PREDNetDiagResult *result = [[PREDNetDiagResult alloc] initWithComplete:complete netClient:client];
    [QNNPing start:host size:64 output:nil complete:^(QNNPingResult *r) {
        [result gotPingResult:r];
    }];
    [QNNTcpPing start:host output:nil complete:^(QNNTcpPingResult *r) {
        [result gotTcpResult:r];
    }];
    [QNNTraceRoute start:host output:nil complete:^(QNNTraceRouteResult *r) {
        [result gotTrResult:r];
    }];
    [QNNNslookup start:host output:nil complete:^(NSArray *r) {
        [result gotNsLookupResult:r];
    }];
    [QNNHttp start:httpHost output:nil complete:^(QNNHttpResult *r) {
        [result gotHttpResult:r];
    }];
}
```

网络诊断信息

7NDEM iOS > 网络诊断日志 退出

App 信息 返回

AppBundled	AppName	AppVersion	tag
pre-engineering.PreDemObjcDemo	PreDemObjcDemo	1.0.1_alpha1	userid_release

设备信息

DeviceId	DeviceModel	OsPlatform	OsVersion	SdkVersion	SdkId
	iPhone9,2	IOS	10.0.2	1.0.1_alpha1	B2ED6352-3B6E-438A-84C9-F32A74DF6FD3

HTTP INFO

DnsRecords	HttpBodySize	HttpCode	HttpDuration	HttpIp
www.a.shifen.com\t683\t5n115.239.210.27\t45\t1n115.239.211.112\t45\t1n	111766	200	162.9989743232727	115.239.210.27

Ping INFO

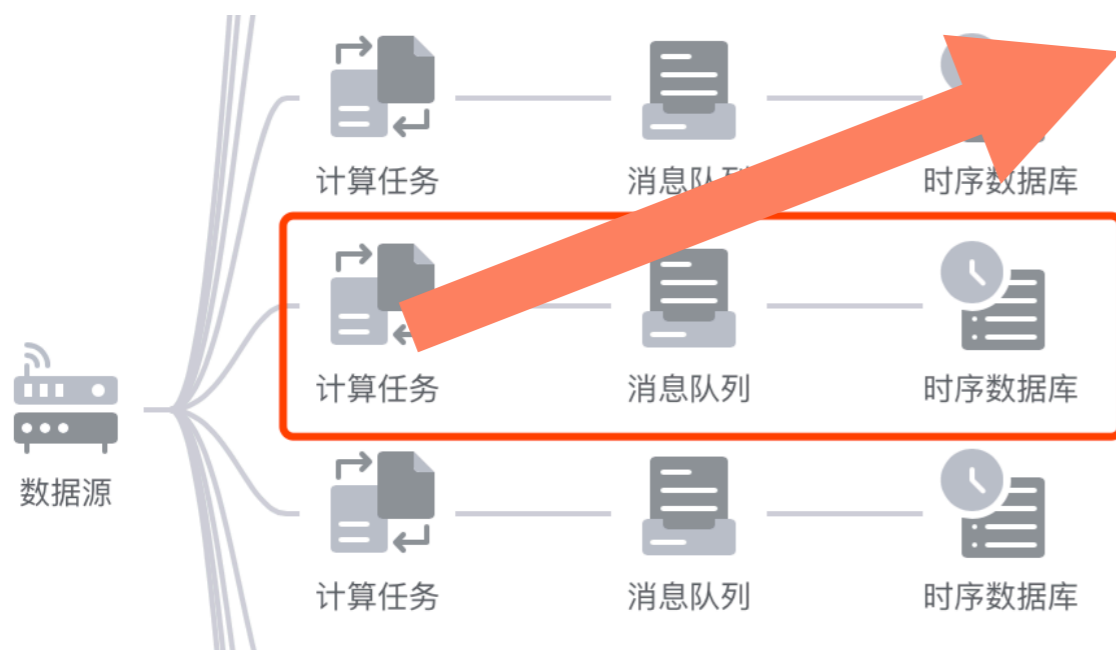
ResultID	04b06c971a166e9c8aa0bb27a7044dfc
PingCode	35
PingIp	115.239.210.27
PingSize	100
PingMaxRtt	0
PingMinRtt	0
PingAvgRtt	0
PingLoss	1
PingCount	1
PingTotalTime	10701.90501213074
PingStddev	0

TCP INFO

TcpCode	0
TcpIp	115.239.210.27
TcpMaxTime	15.60097932815552
TcpMinTime	9.930014610290527
TcpAvgTime	12.94598976771037
TcpLoss	0
TcpCount	3
TcpTotalTime	359.5709800720215
TcpStddev	2.329190583118923

自定义上报

- 接收宿主 app 传入的数据模型直接进行序列化并上报
- 利用 Pandora 强大的分析功能进行自定义分析与展示



```
select
    cast(cast(time as bigint) / 5 as bigint) * 5
as time,
    cdn,
    country,
    province,
    isp,
    platform,
    sum(v4) as total,
    count(v4) as number
from stream
where
    tag = 'monitor'
group by
    cast(cast(time as bigint) / 5 as
    bigint) * 5, cdn, country, province,
    isp, platform
```

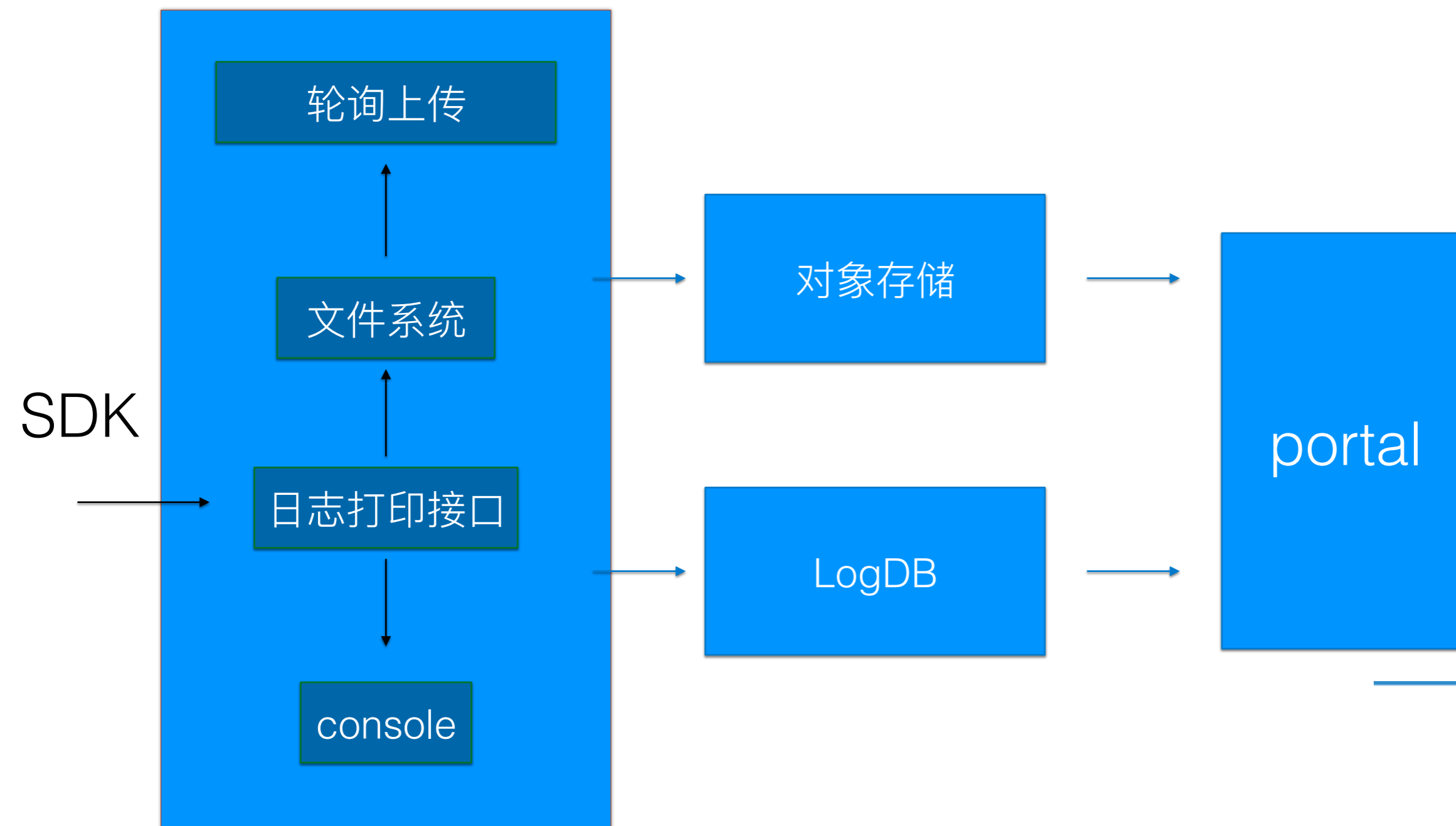

开放分析-数据流

- 树状拓扑
- 每个分支代表一个统计计算或数据导出功能
- 可以导出到Http, Logdb, Tsdb, 存储
- 实时, 离线

日志上报

- 支持区按照级别过滤打印
- 支持自定义将 log 进行本地保存及上传

日志上报



日志上报

```
+ (void)startCaptureLogWithLevel:(PREDLogLevel)logLevel {
    if (_fileLogger && _fileLogLevel == logLevel) {
        return;
    }
    _fileLogLevel = logLevel;
    [self stopCaptureLog];
    PREDLogFileManager *fileManager = [[PREDLogFileManager alloc] init];
    _fileLogger = [[DDFileLogger alloc] initWithLogFileManager:fileManager]; // File Logger
    _fileLogger.rollingFrequency = 0;
    [DDLog addLogger:_fileLogger withLevel:(DDLogLevel)logLevel];
    _logStartTime = [NSDate date];
}

+ (void)stopCaptureLog {
    if (_fileLogger) {
        [DDLog removeLogger:_fileLogger];
        _fileLogger = nil;
    }
}
```

日志上报

```
- (void)didArchiveLogFile:(NSString *)logFilePath {
    NSLog(@"%s: %@", __PRETTY_FUNCTION__, logFilePath);
    [self uploadLog:logFilePath startTime:[_rfc3339Formatter stringFromDate:_logStartTime] endTime:
        [_rfc3339Formatter stringFromDate:[NSDate date]] retryTimes:0];
    _logStartTime = [NSDate date];
}

- (void)didRollAndArchiveLogFile:(NSString *)logFilePath {
    NSLog(@"%s, %@", __PRETTY_FUNCTION__, logFilePath);
    [self uploadLog:logFilePath startTime:[_rfc3339Formatter stringFromDate:_logStartTime] endTime:
        [_rfc3339Formatter stringFromDate:[NSDate date]] retryTimes:0];
    _logStartTime = [NSDate date];
}
```

Web SDK

- 窗口性能分析
- AJAX 分析
- 错误拦截

智能异常报告

- 根据历史推断阈值进行告警，发现异常模式

机器学习

- 利用机器学习对崩溃等信息进行自动分析

更丰富的图表展示

- Http 请求分析多维度聚合
- 崩溃信息多维度聚合

完全开源，和大家一起演进

- 开放的心态，一起成长
- Android 地址：
<https://github.com/pre-dem/pre-dem-android>
- iOS 地址：
<https://github.com/pre-dem/pre-dem-objc>



简单·可信赖

Q&A



网络服务接入优化

梁海龙

数据

1

原因

2

方案讨论

3

CONTENTS

目录



数据

你的失败数据要包括哪些？

用户终止

密码错误

余额不足

无法解析
主机域名

403

404

5xx

网络超时

返回结果
为空

数据方案

口径

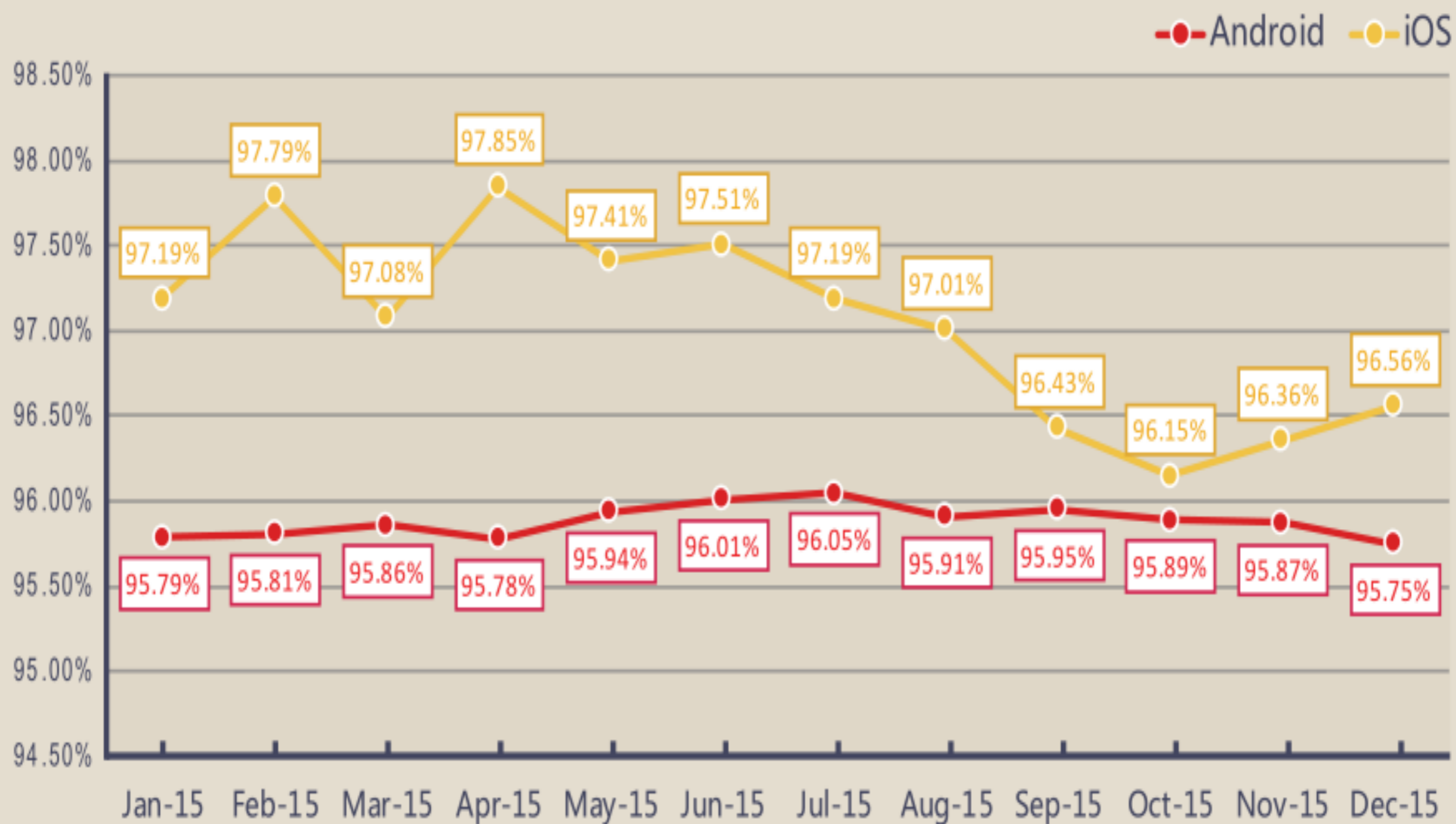
抽样

即时性

网络路径

HTTP响应成功率，听云白皮书数据

HTTP响应成功率





原因

DNS, 某家电商均值2%-3%

good

- 易用
- CDN
- 浏览器

bad

- 不能解析
- 发布慢
- 劫持

内容劫持

发现

暗标记

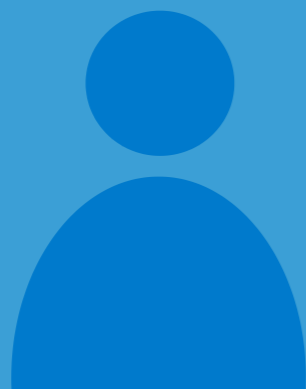
网页检测

解决

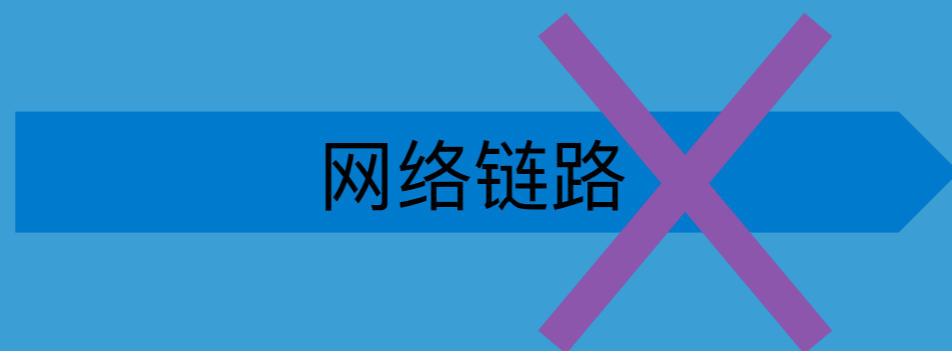
https

私有加密

网络故障



用户



机房

CDN

good

- 快
- 减轻重大故障影响
- 防DDOS

bad

- 不实时
- 故障

2G/3G/4G, 以及某些小运营商

问题

基站踢

连接限制

拥塞丢弃

公共IP

解决

维持

长连接

不要搞特殊

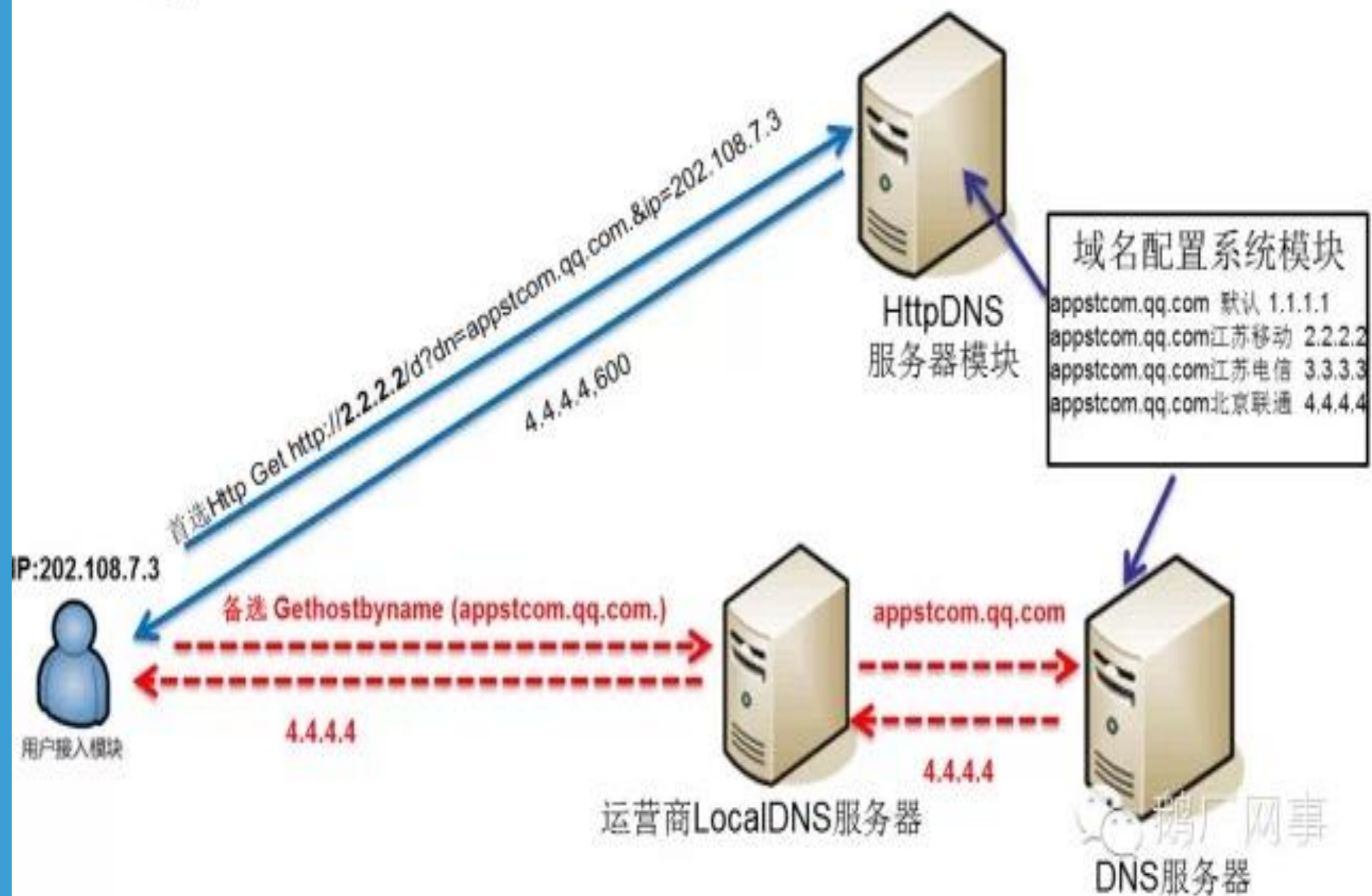
小心IP误杀



方案讨论

HTTP DNS

HttpDNS基本原理



HTTP DNS 酷狗实践



APP配置列表，同步更新

- 安装包默认配置
- 同步服务IP+域名
- 定时同步

不同业务不同策略

- 多域名
- IP优先
- 容灾网关

“这个接口不能重试。”



幂等

所有的接口都可以改造成幂等的接口。

HTTPS 的优化

长连接

握手优化

加密优化

等等

长连接，多路复用

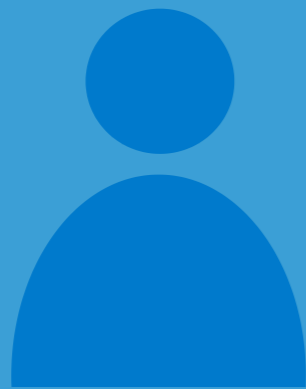
HTTP 1.0

HTTP 1.1

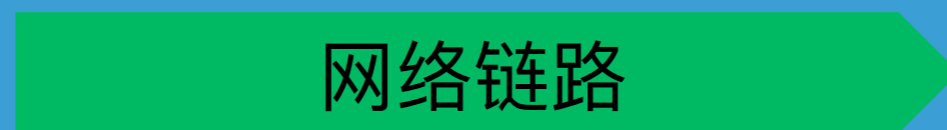
HTTP 2

SPDY

网络故障的应对



用户



机房

给多一条路走

给多一条路走策略



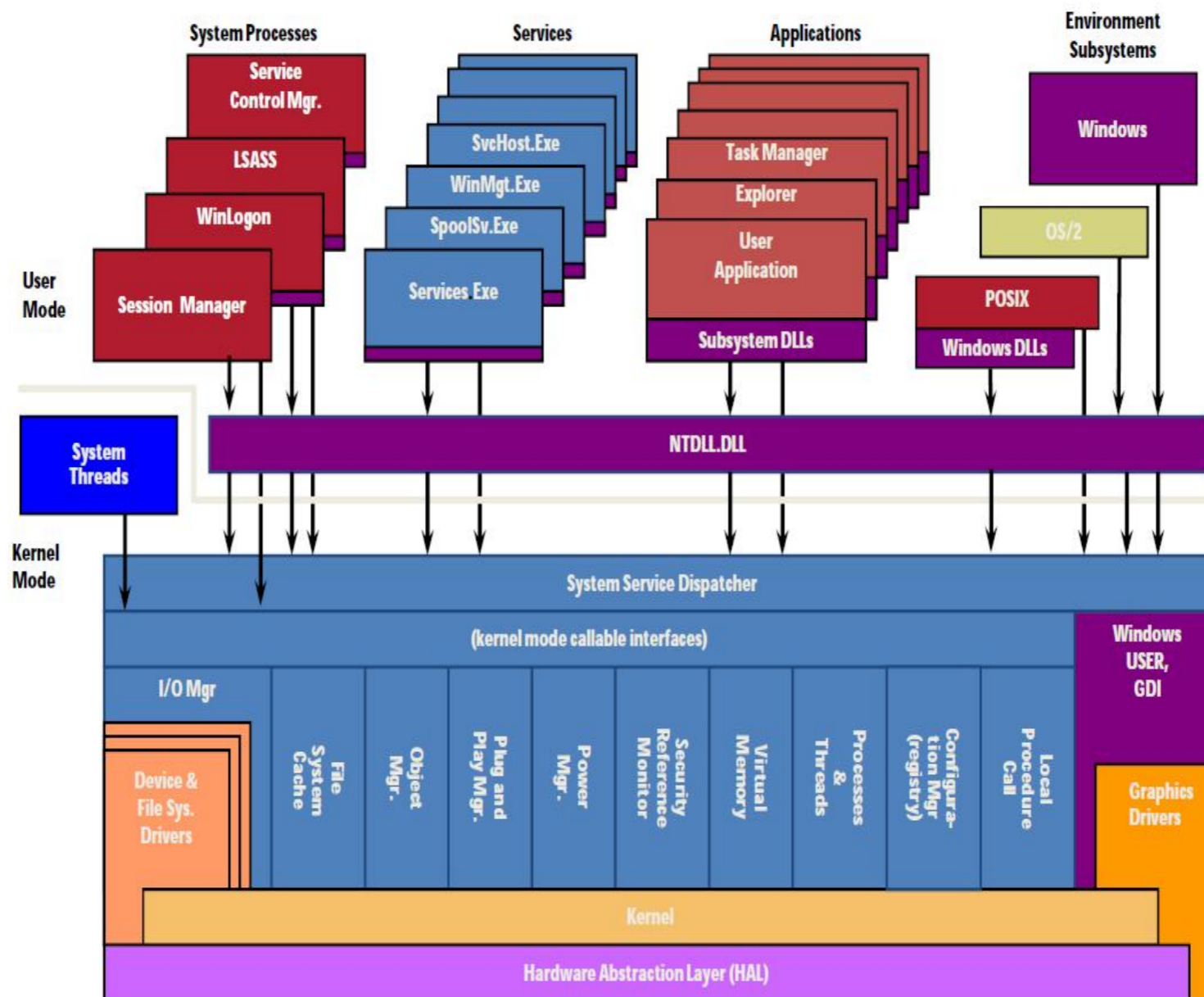
东西、南北机房

CDN、备用CDN、源站

公网、网关+专线

APP架构模式

Windows Architecture



hardware interfaces (buses, I/O devices, interrupts, interval timers, DMA, memory cache control, etc., etc.)

Original copyright by Microsoft Corporation. Used by permission.



最后还有一句话

业务合适的才是真的好







Q & A



THANK YOU