HUAWEI

< HDC.Together >

HUAWEI DEVELOPER CONFERENCE 2021

HarmonyOS编译器和运行时关键技术

HarmonyOS对编译器运行时的需求

支持HarmonyOS生态

- 支持应用开发语言（JS/TS，Java，C/C++）
- 设备无关的应用分发格式，支持不同设备分发部署

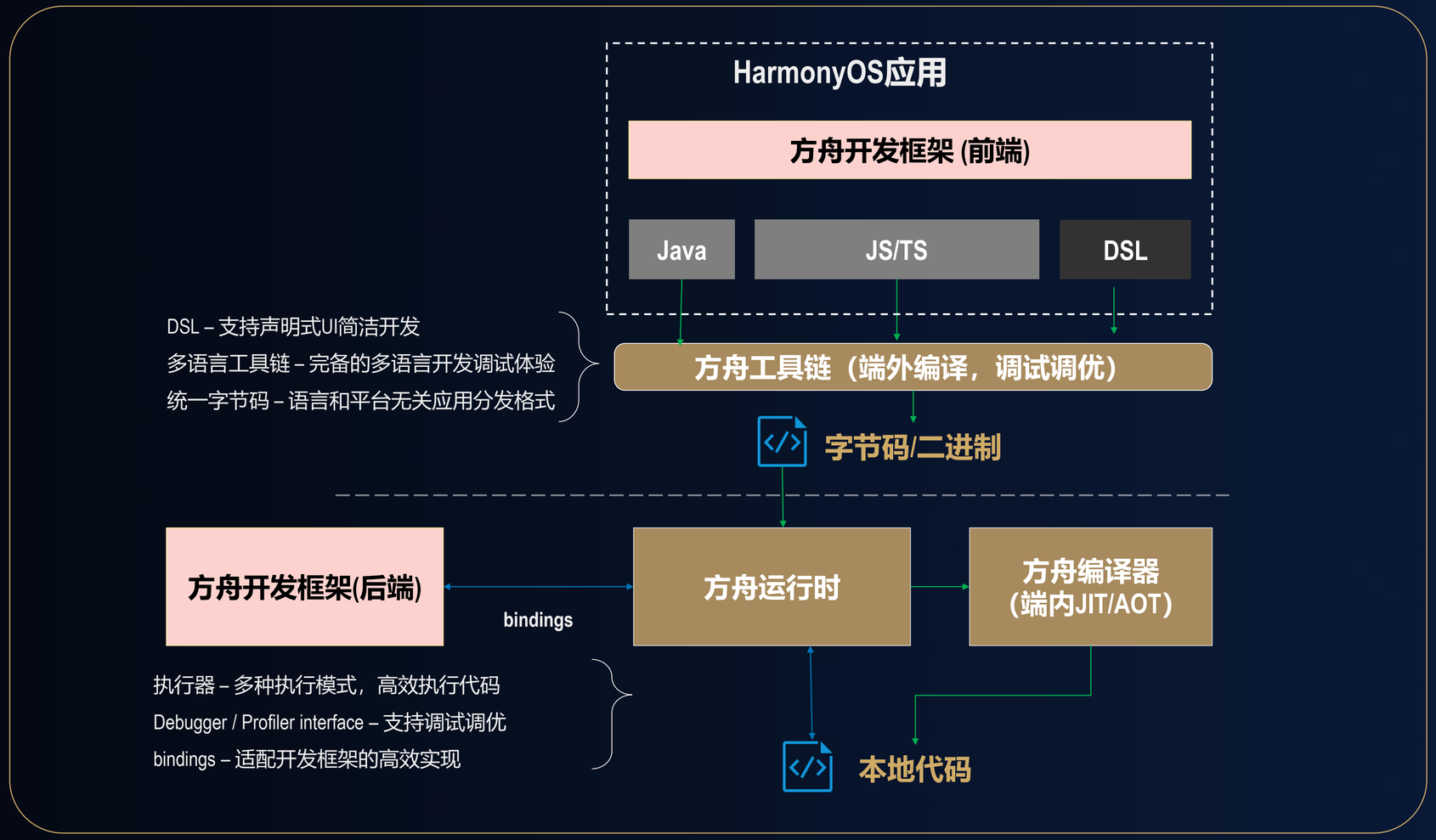
提升应用开发体验

- 语言和范式提升开发效率
- 支持跨设备、分布式应用的开发、编译构建、调试

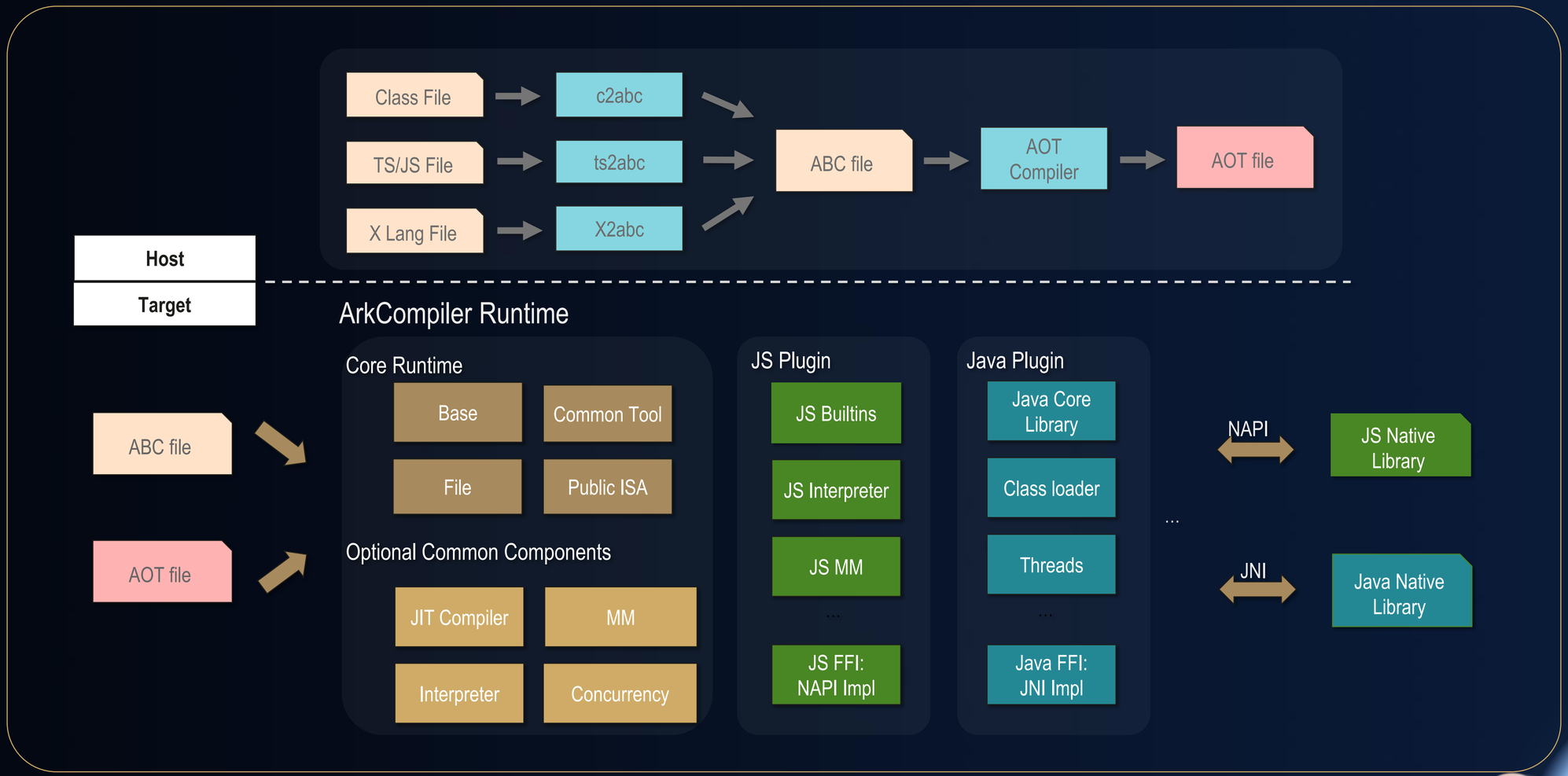
提升用户体验

- 多端适配，优化不同硬件平台上系统和应用的ROM/RAM占用、性能和功耗
- 支持并优化多设备协同全场景业务的分布式应用运行

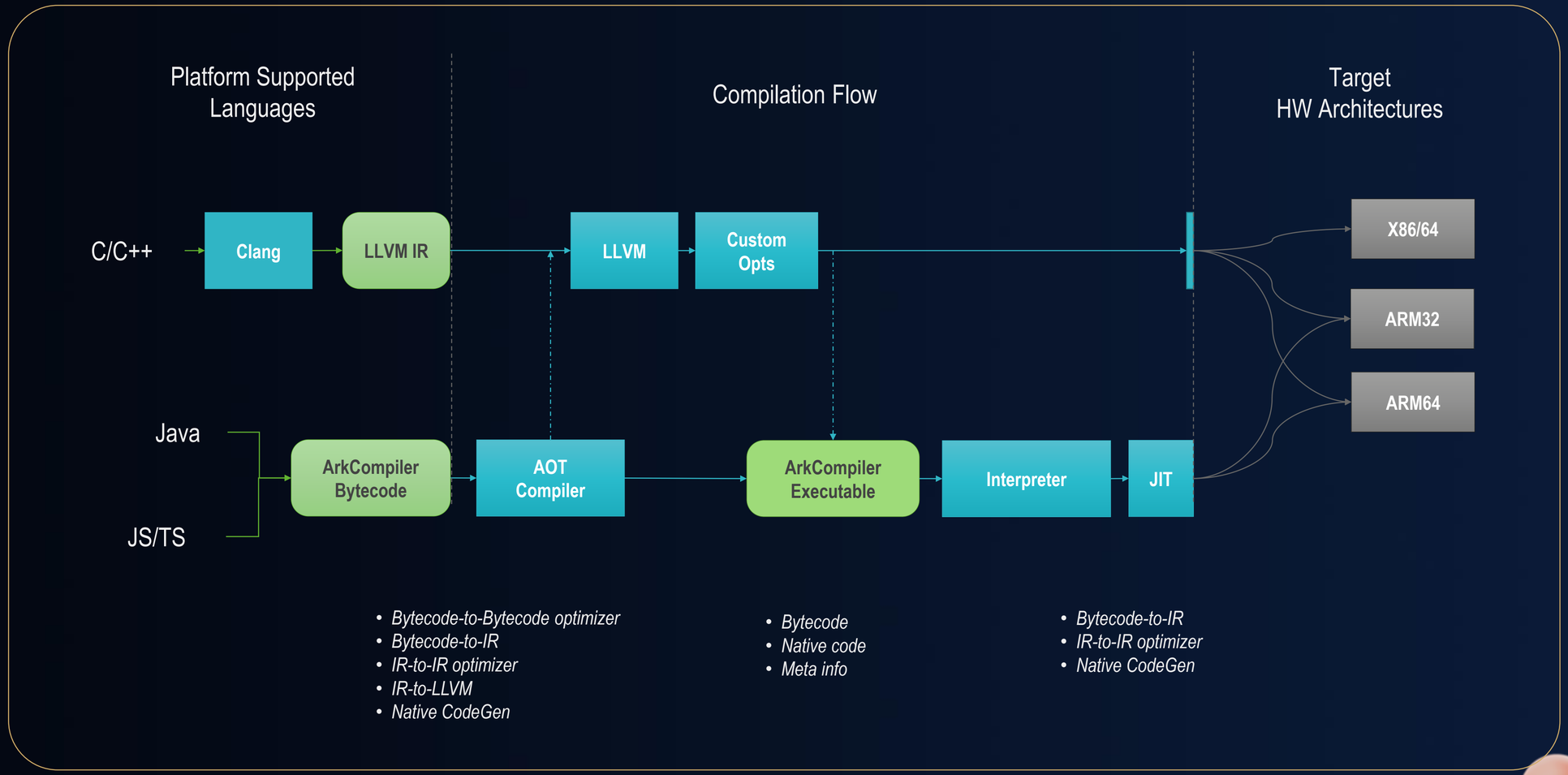
ArkCompiler对HarmonyOS应用的支持



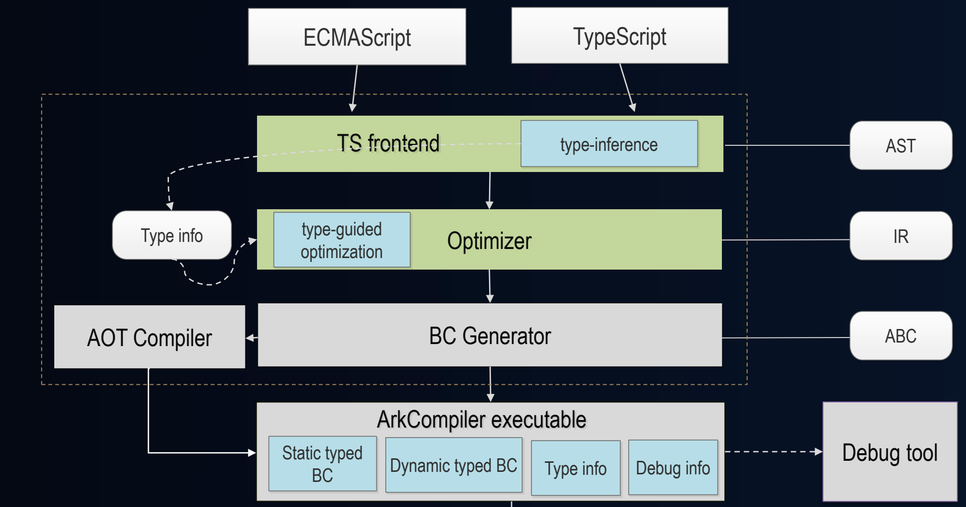
ArkCompiler架构

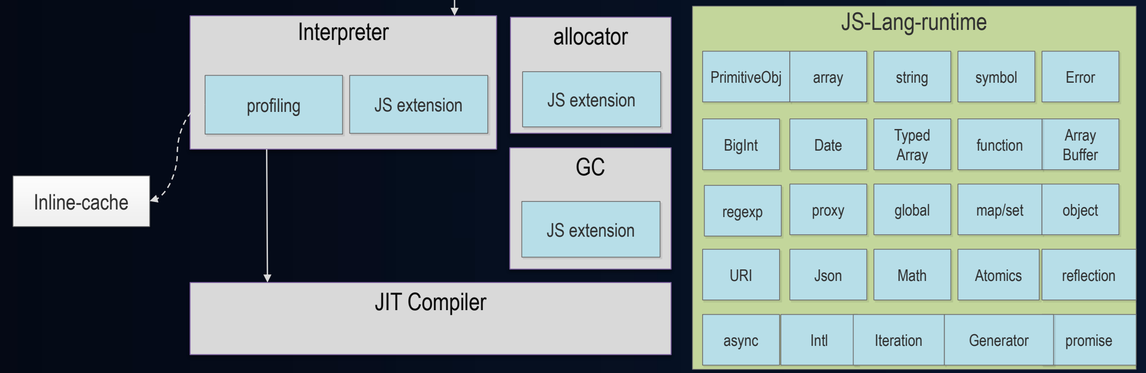


多语言编译流程


Platform Supported Languages
Compilation Flow
Target HW Architectures
C/C++
Clang
LLVM IR
LLVM
Custom Opts
X86/64
ARM32
ARM64
Java
JS/TS
ArkCompiler Bytecode
AOT Compiler
ArkCompiler Executable
Interpreter
JIT
• Bytecode-to-Bytecode optimizer
• Bytecode-to-IR
• IR-to-IR optimizer
• IR-to-LLVM
• Native CodeGen
• Bytecode
• Native code
• Meta info
• Bytecode-to-IR
• IR-to-IR optimizer
• Native CodeGen

基于类型信息的JavaScript编译优化





- 基于类型的编译优化大量移至Host侧
  - 源码预编译，消除端侧源码解析开销
  - 基于TS显式类型声明，应用类型推导进行类型特化优化
  - 类型信息保留至运行时

- 利用静态类型信息的编译和执行优化
  - 辅以基于profiling的投机优化
- JS运行时库fast path汇编/AOT化

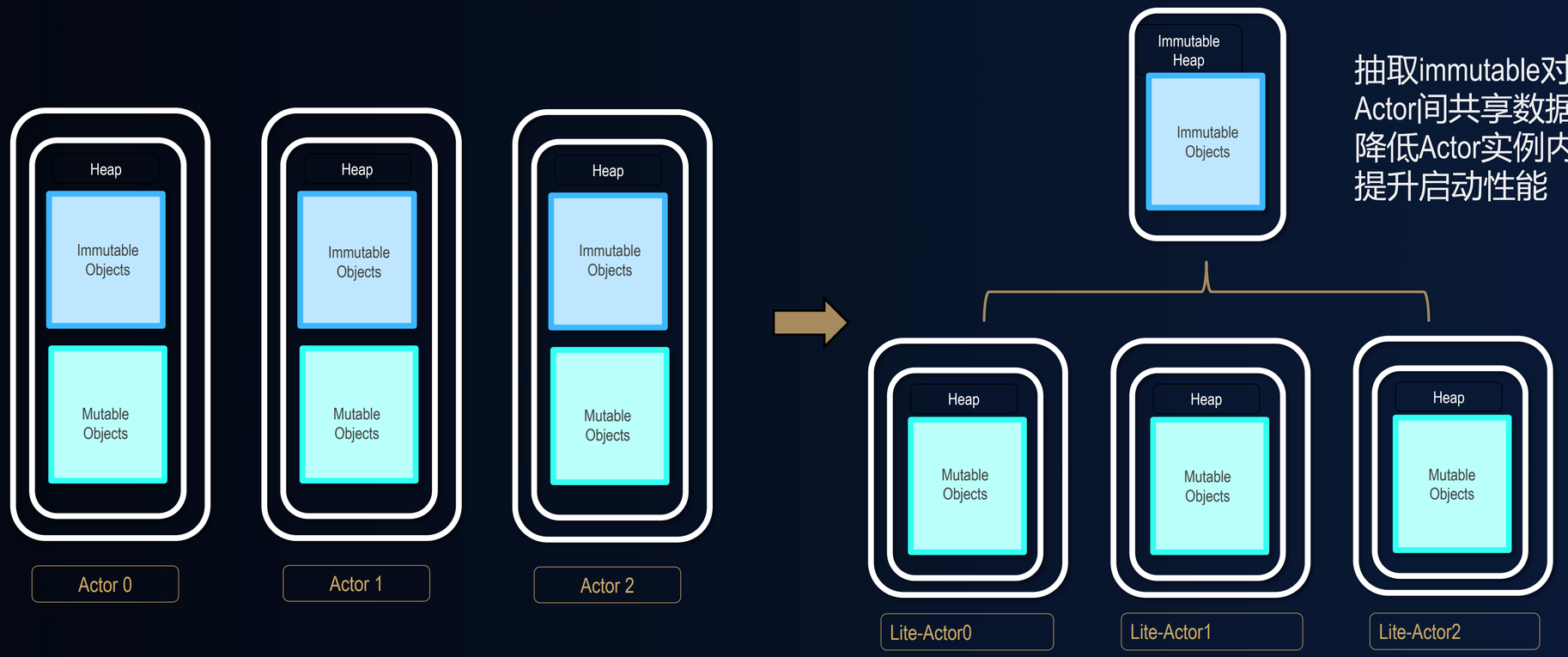
Actor并发编程模型

<JS Worker Sample Code>

```
// Test.js

let count = new SharedArrayBuffer(8);
Image img = [];
function collect(msg) {
   img.push(msg.img);
}

let w1 = new Worker(“load-worker.js”);
w1.postMessage( { path:“xxx.jpg”, count: count } );
w1.onmessage = collect;

let w2 = new Worker(“load-worker.js”);
w2.postMessage( { path:“xxx.bmp”, count: count } );
w2.onmessage = collect;
```

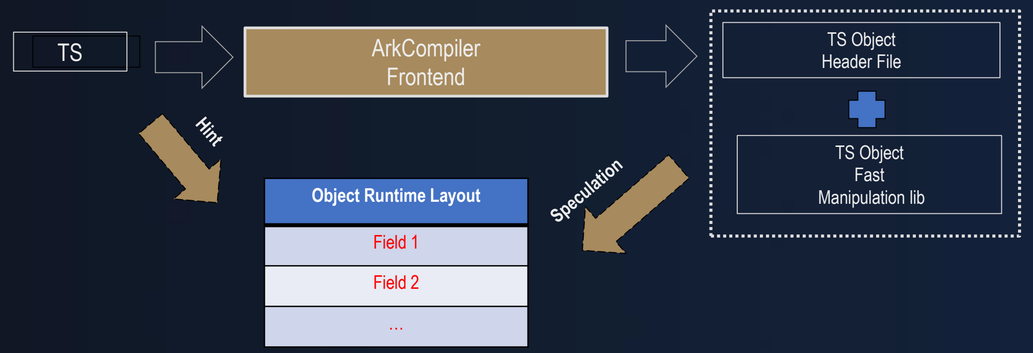
```
// load-worker.js

onmessage = function (msg) {
   Atomics.add(msg.count, 0, 1);
   result= ReadImages(msg.path);
   postMessage( { img: result } );
}
```

轻量级Actor实现



抽取immutable对象，
Actor间共享数据
降低Actor实例内存和
提升启动性能

Fast TS FFI



TS代码编译

根据TS Class声明及运行时约定，生成TS对象布局描述C/C++头文件以及操作TS对象的native库

Native代码编译

include TS对象头文件以及链接对应库，实现C++代码直接操作TS对象

TS对象操作库中插入类型检查，对象布局如在运行时发生变化，回退慢速路径

简化的多语言编程模型

TypeScript

```
import {RequestParam, Action, ResutlType } from "java_request_package";

plus: async function() {
  let actionData: RequestParam = new RequestParam;
  actionData.setFirstNum(1024);
  actionData.setSecondNum(2048);

  let action: Action = new Action;
  action.bundleName = 'com.example.hiaceservice';
  action.abilityName = 'com.example.hiaceservice.ComputeServiceAbility';
  action.messageCode = ACTION_MESSAGE_CODE_PLUS;
  action.data = actionData;
  action.abilityType = ABILITY_TYPE_EXTERNAL;
  action.syncOption = ACTION_SYNC;

  let ret: ResultType = await FeatureAbility.callAbility(action);
  if (ret.code == 0) {
    console.info('plus result is:' + ret.abilityResult);
  } else {
    console.error('plus error code:' + ret.code);
  }
}
```

Java

```
public boolean onRemoteRequest(int code, MessageParcel data, MessageParcel reply, MessageOption option) {
    switch (code) {
        case PLUS: {
            RequestParam p;
            try {
                p = (RequestParam)data.data;
            } catch (RuntimeException e) {
                HiLog.error(LABEL, "convert failed.");
            }
            ResultType ret = new ResultType;
            ret.code = SUCCESS;
            ret.abilityResult = p.getFirstNum() + p.getSecondNum();
            reply.data = ret;
            break;
        }
        default: {
            reply.writeString("service not defined");
            return false;
        }
    }
    return true;
}
```

- 代码兼容 - 已有Java代码无需修改，TS可直接创建、操作和传递Java对象
- 开发便利 - 开发者无需处理交互数据的序列化/反序列化，提高开发效率，降低复杂度
- 性能优化 - 跨语言交互优化, 利用更多静态类型信息的编译优化

ArkCompiler 3.0: 高效的跨端编译运行

- TS原生支持
  - TS代码预编译
  - 静态类型推导分析
- 轻量级并发
  - Actor编程模型
  - 轻量级实现
- 高效跨语言交互
  - 简化FFI* 编程模型
  - 直接数据访问

HarmonyOS应用启动性能提升20%

<HDC.Together>
华为开发者大会2021

扫码参加1024程序员节

<解锁HarmonyOS核心技能，赢取限量好礼>

开发者训练营

Codelabs 挑战赛

HarmonyOS技术征文

HarmonyOS开发者创新大赛

扫码了解1024更多信息

报名参加HarmonyOS开发者创新大赛

谢谢

欢迎访问HarmonyOS开发者官网

欢迎关注HarmonyOS开发者微信公众号