鸿蒙生态应用开发白皮书V4.0

HUAWEI

商标声明

以上为华为公司的商标（非详尽清单），未经华为公司书面事先明示许可，任何第三方不得以任何形式使用。

注意

华为会不定期对本文档的内容进行更新。

本文档仅作为使用指导，文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

华为终端有限公司

地址： 广东省东莞市松山湖园区新城路 2 号

网址： https://consumer.huawei.com

CONTENT

Chapter 1

万物互联时代应用开发的机遇、挑战和趋势

经过十多年的发展，传统移动互联网的增长红利已渐见顶。万物互联时代正在开启，应用的设备底座将从几十亿手机扩展到数百亿 IoT 设备。GSMA 预测到 2025 年，全球物联网终端连接数量将达 246 亿个，其中消费物联网终端连接数量将达 110 亿个（注：数据来自于全球移动通信系统协会发布的《2020 年移动经济》报告）。IDC 预计到 2025 年，中国物联网总连接量将达到 102.7 亿个（注：数据来自于 IDC 发布的《中国物联网连接规模预测，2020—2025》报告）。全新的全场景设备体验，正深入改变消费者的使用习惯。 同时应用开发者也面临设备底座从手机单设备到全场景多设备的转变，通过全场景多设备作为全新的底座，为消费者带来万物互联时代更为高效、便捷的体验。

新的场景同时也带来了新的挑战。开发者不仅需要支持更加多样化的设备，还需要支持跨设备的协作。不同设备类型意味着不同的传感器能力、硬件能力、屏幕尺寸、操作系统和开发语言，还意味着差异化的交互方式。同时跨设备协作也让开发者面临分布式开发带来的各种复杂性，例如跨设备的网络通信、数据同步等。若采取传统开发模式，适配和管理工作量将非常巨大。当前移动应用开发中遇到的主要挑战包括：

- 针对不同设备上的不同操作系统，重复开发，维护多套版本。
- 多种语言栈，对人员技能要求高。
- 多种开发框架，不同的编程范式。
- 命令式编程，需关注细节，变更频繁，维护成本高。

与此同时，AI 时代全面来临，在 PC 互联网到移动互联网到智能化终端演进过程中，AI 计算主要在云端数据中心进行，非常依赖网络，具有一定的时延，且数据传输的安全性、私密性不能得到有效保证。随着人们对交互和信息获取的智能化要求越来越高，移动设备的计

算能力越来越强，在设备侧就能提供 AI 的相关能力，例如自然语言交互、环境智能感知、图像识别等。如何快速地使用设备侧的强大 AI 能力，使自己的应用更加智能化，进而更好的服务消费者，也是开发者面临的全新挑战。

移动终端上的应用生态发展到今天也面临着变革。传统厚重的 App，整体体验好，功能齐全，但开发成本高、周期长，且存在搜索，安装，升级，卸载等一系列需要用户主动关注的显性操作，这些显性操作给用户带来了实质性的使用成本。轻量化、可快速达成消费者意图、可独立执行、完成单一功能的程序实体正成为新的趋势，例如小程序、App Clips、快应用等。根据阿拉丁指数的统计，全网小程序已经突破 700 万个（注：数据来自于阿拉丁研究院发布的《2021 年度小程序互联网发展白皮书》），远超 App 数量。大型应用开发者普遍向用户提供轻量化程序实体。在很多特定的使用场景下，小程序等轻量化程序实体的使用占比已超过 App，成为面向用户的主要触达方式。

轻量化的程序实体所具备的“即用即走、无需安装卸载、持续更新”的特征，也推动了 App 基于搜索下载的“人找应用”的传统分发向“服务找人”的智慧分发的演进。App 遵循“搜索、下载、安装、使用”的模式，用户主动发现的成本高，拉新、促活、召回的全生命周期流程相对被动。轻量化的程序实体具有即用即走的体验，可通过各类终端的系统级智慧入口进行分发，甚至可以在生态应用中分发，依托无所不在的入口流量和标签化识别，向用户主动提供精准服务。配合 CPS（Cost Per Sale）等商业模式，可以为开发者带来更高的 ROI（Return of Investment）。

为了更好的抓住机遇，应对万物互联所带来的一系列挑战，新的应用生态应该具备如下特征：

- 单一设备延伸到多设备：应用一次开发就能在多个设备上运行，软件实体能够从单一设备转移到其他设备上，且多个设备间能够协同运行，给消费者提供全新的分布式体验。
- 厚重应用模式到轻量化服务模式：提供轻量化的服务，较低的资源消耗，一步直达，快速完成消费者特定场景的任务。
- 集中化分发到 AI 加持下的智慧分发：为消费者提供智慧场景服务，实现“服务找人”。
- 纯软件到软硬芯协同的 AI 能力：提供软硬芯协同优化的 AI 能力，全面满足应用高性能诉求。

Chapter 2

鸿蒙生态应用开发核心概念

HarmonyOS 应用：使用 HarmonyOS SDK 开发的应用程序，能够在华为终端设备（如：手机、平板等）上运行，其有两种形态：

- 传统方式的需要安装的 App。
- 轻量级，具备免安装，随处可及，服务直达，自由流转等关键特征的元服务。

HarmonyOS 元服务：元服务是 HarmonyOS 面向万物互联时代提供的一种轻量级应用程序形态。它基于 HarmonyOS 平台开放能力开发，打包为 App Pack 形态，运行在 HarmonyOS 操作系统，由 HarmonyOS 应用程序框架管理，具备随处可及、服务直达、跨设备等核心特征。

服务卡片：HarmonyOS 系统定义的一种界面展示形式，是 HarmonyOS 应用和元服务的一个可选组成部分，将重要信息或操作前置到卡片，以达到服务直达，减少操作层级的目的。服务卡片常用于嵌入到其他系统应用（桌面/负一屏）中作为其界面的一部分显示，并支持点击拉起应用或元服务。

HarmonyOS 应用与元服务基于同一个鸿蒙系统技术栈开发，同属一个鸿蒙生态。开发者通过业务解耦将应用分解为若干元服务独立开发，按需根据场景组合成复杂应用。

Chapter 3

鸿蒙生态应用核心技术理念

- 1）一次开发，多端部署
- 2）可分可合，自由流转
- 3）统一生态，系统智能

在万物智联时代重要机遇期，鸿蒙系统结合移动生态发展的趋势，提出了三大技术理念：一次开发，多端部署；可分可合，自由流转；统一生态，系统智能。

图 3-1：核心技术理念

1）一次开发，多端部署

“一次开发，多端部署”指的是一个工程，一次开发上架，多端按需部署。目的是支撑开发者高效地开发多种终端设备上的应用。为了实现这一目的，鸿蒙系统提供了几个核心能力，包括多端开发环境，多端开发能力以及多端分发机制。

图 3-2：一次开发 多端部署

多端开发环境

HUAWEI DevEco Studio 是面向全场景多设备提供的一站式开发平台，支持多端双向预览、分布式调优、分布式调试、多设备模拟、低代码可视化开发等能力，帮助开发者降低成本、提升效率、提高质量。HUAWEI DevEco Studio 提供的核心能力如下图所示：

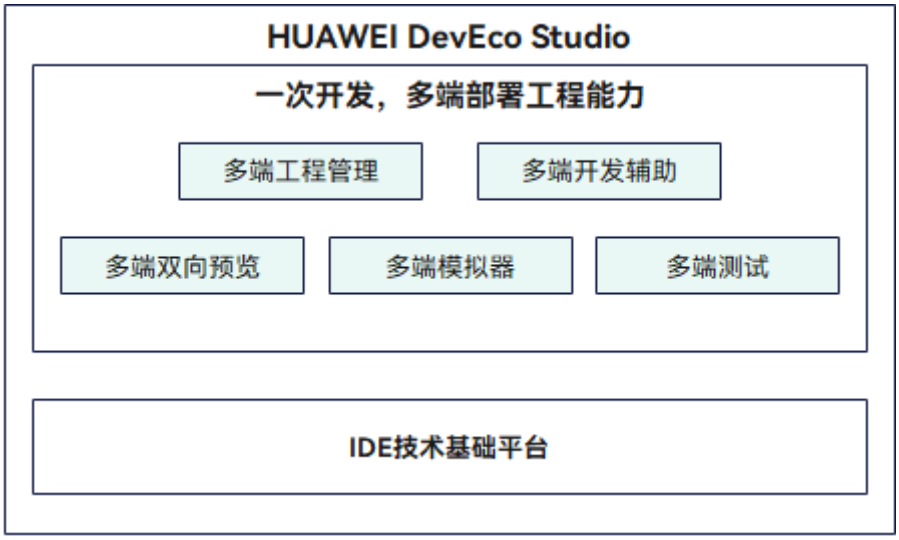


图 3-3：HUAWEI DevEco Studio 核心功能和特征

1. 多端工程管理

创建鸿蒙应用/元服务/模块工程时，支持选择需要适配的设备类型，DevEco Studio 会根据选中的设备类型去匹配及检测对应的功能模块。避免不同的 Entry 模块之间，出现支持的设备类型重叠的情况。推荐典型的多设备工程目录结构如下所示：

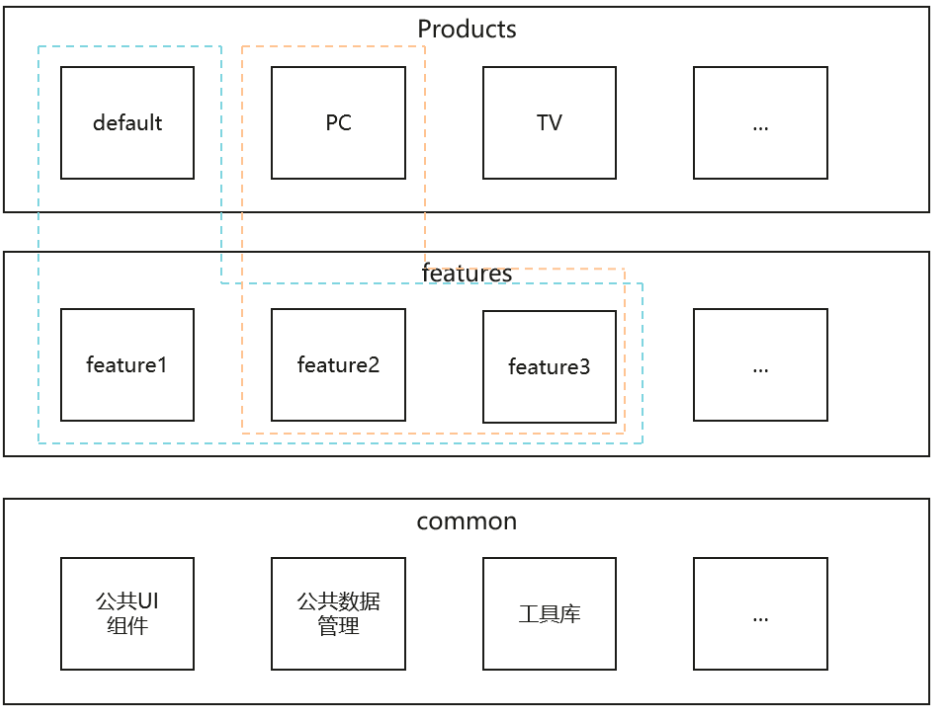


图 3-4：多设备工程目录结构

2. 多端开发辅助

在编码阶段，DevEco Studio 提供了一些实时检测机制，以辅助开发者写出高质量的多设备应用。

- API 检测：检测设备特有 API 是否可用，以避免 API 被误用。例如：在一个平板应用上调用 NFC 接口，编辑区会标红。此外，查看提示信息可以知道，平板设备不具备 NFC 的能力。

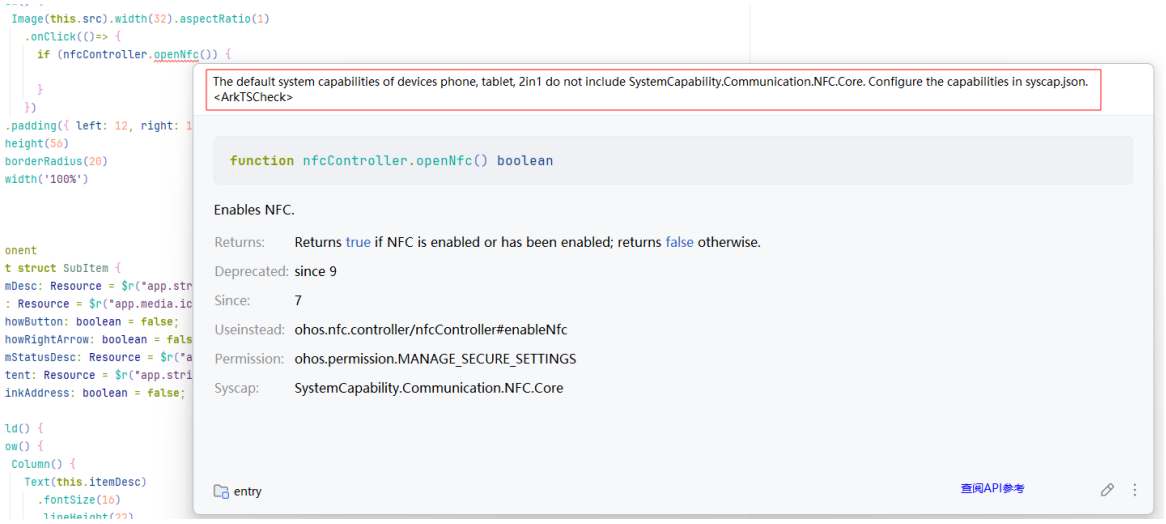


图 3-5：开发辅助检测提示

- 组件参数检测：当组件的宽度、高度、圆角半径等参数使用了固定值，并且和系统 Resource 相同时，DevEco Studio 会给出警告，并提示可使用系统 Resource 进行一键替换。

3. 多端双向预览

在鸿蒙生态应用的开发阶段，因不同设备的屏幕分辨率、形状、大小等差异，开发者需要在不同设备上查看 UI 界面显示，确保实现效果与设计目标一致。传统的开发模式下，开发者需要获取大量不同的真机设备用于测试验证。HUAWEI DevEco Studio 提供了多种设备的双向预览能力，支持同时查看 UI 代码在多个设备上的预览效果，并支持 UI 代码和预览效果的双向定位修改。

4. 多端模拟器

移动应用开发时需要使用本地模拟器来进行应用调试，实现快速开发的目的。鸿蒙生态应用需要运行在多种不同类型的设备上，为此 HUAWEI DevEco Studio 提供了不同类型的设备模拟，支持开发者在多个模拟设备上同时进行开发调试，降低门槛、节约成本。

DevEco Studio 在模拟器中预置了 Mate、Pura 系列机型的模板，同时也支持自定义屏幕参数。另外为支撑多设备并行调测，提供了多屏模拟能力，可以在一个模拟器上同时进行最多 4 种设备的并行调试。

5. 多端测试

DevEco Studio 针对多设备静态编码检测、运行效果检测提供了两种类型检测工具。

- Code Linter 编码检测：支持对应用/元服务的代码文件进行扫描、检测，对于违反多设备开发规范的代码进行告警。典型规范如颜色使用固定值、字体单位未使用 fp 等。
- 应用体检工具：通过体检工具可以检测应用/元服务在设备上的运行效果，检测出人眼难以发觉的 UX 问题，如不同设备下的色差、字体大小、图标清晰度等。

多端开发能力

应用如需在多个设备上运行，需要适配不同的屏幕尺寸和分辨率、不同的交互方式（如触摸和键盘等）、不同的硬件能力（如内存差异和器件差异等），开发成本较高。因此，多端开发能力的核心目标是降低多设备应用的开发成本。为了实现该目标，鸿蒙系统提供了以下几个核心能力，支持多端 UI 适配，交互事件归一，设备能力抽象，帮助开发者降低开发与维护成本，提高代码复用度。

1. 多端 UI 适配

不同设备屏幕尺寸、分辨率等存在差异，鸿蒙系统将对屏幕进行逻辑抽象，包括尺寸和物理像素，并提供丰富的自适应/响应式的布局和视觉能力，方便开发者进行不同屏幕的界面适配。

屏幕逻辑抽象：鸿蒙系统提供虚拟像素 vp（virtual pixel）对分辨率进行抽象，不同设备的系统在底层将物理像素转化成虚拟像素，为应用开发者提供统一单位。不同设备的尺寸存在差异，鸿蒙系统根据设备的屏幕水平宽度，抽象和定义了五种尺寸：超小（xs）、小（sm）、中（md）、大（lg）、超大（xl）；纵向抽象和定义了三种比例：小（Small）、中（Medium）、大（Large） 。这些抽象后的屏幕尺寸与日常使用的设备屏幕类型有一定的对应关系。开发者可面向应用运行的目标设备进行屏幕类型的适配。

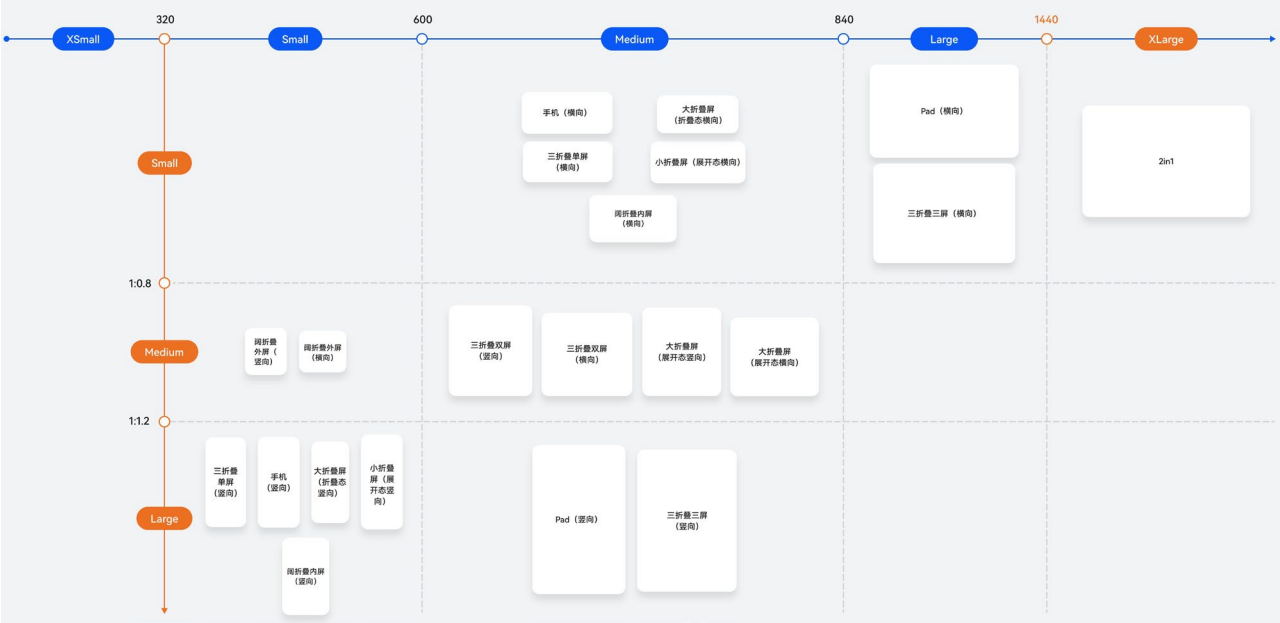


图 3-6：尺寸抽象化

布局：鸿蒙系统提供的布局主要分为自适应布局和响应式布局。自适应布局是当外部容器大小发生变化时，容器内元素可以根据相对关系自动变化以适应外部容器变化的布局能力。

相对关系包含占比、固定宽高比、显示优先级等。当前自适应布局能力主要有 7 种：拉伸能力、均分能力、占比能力、缩放能力、延伸能力、隐藏能力、折行能力。响应式布局是当显示空间大小发生变化时，布局可以根据预设断点、栅格或特定的特征（如屏幕方向、窗口宽高等）进行自动切换。当前响应式布局能力主要有 3 种：断点、媒体查询、栅格布局。鸿蒙系统将响应式布局能力下沉到默认组件的基础属性中，比如网格、列表和轮播组件等，支持自动增加显示列数，以便充分利用屏幕宽度，节省应用开发工作量。

视觉：鸿蒙系统提供的视觉样式能力，包括分层参数、多态组件和主题。

2. 交互事件归一

事件归一抽象：不同设备间的交互方式等存在差异，如触摸、键盘、鼠标、语音、手写笔等，鸿蒙系统将不同设备的输入映射成归一交互事件，从而简化开发者适配逻辑。

以缩放交互为例，通过多指触控的张合来完成缩放动作，在多设备场景下，缩放交互会出现多种不同的操作输入方式，如表 3-1 所示。为了让应用更容易的支持不同的交互方式，鸿蒙系统提供归一的缩放交互事件。

表 3-1：缩放交互的规则

操作方式	触屏双指捏合交互	键盘 Ctrl 键+鼠标滚轮交互	键盘 Ctrl 键+“+/-”键交互	触控板双指捏合交互	表冠旋转交互
上报事件	触屏双指捏合事件	按键+滚轮组合事件	按键组合点击事件	触控板双指捏合事件	表冠旋转事件

组件归一响应：当应用部署在不同设备上供用户使用时，需要支持多种 I/O 设备，界面呈现出相应的状态为用户提供正确的视觉引导。例如触摸时显示按压状态，鼠标特有的悬停状态，键盘走焦状态。鸿蒙系统默认提供多种交互方式的组件实现，方便开发者支持多种输入方式。

3. 设备能力抽象

不同设备间的软、硬件能力等存在差异，如设备是否具备定位能力、是否具备摄像头、是否具备蓝牙功能等，鸿蒙系统需要对设备能力进行逻辑抽象，并提供接口来查询设备是否支持某一能力，方便开发者进行不同软、硬件能力的功能适配。在鸿蒙系统中，使用 SystemCapability（简写为 SysCap）定义每个部件对应用开发者提供的系统软硬件能力。应用开发者基于统一的方式访问不同设备的能力。

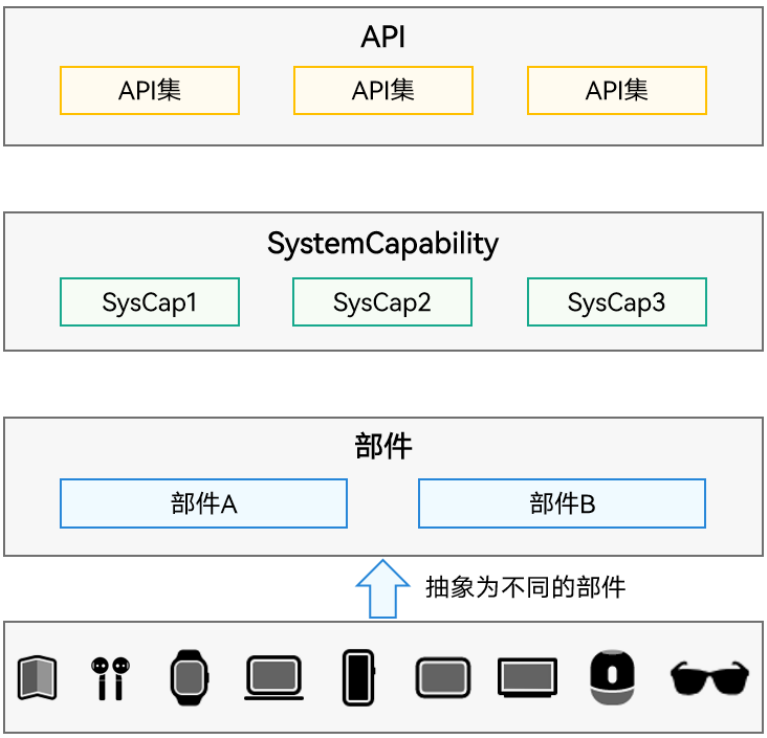


图 3-7：API、SystemCapability、部件和设备的关系

多端分发机制

如果需要开发多设备上运行的应用，一般会针对不同类型的设备多次开发并独立上架。开发和维护的成本大，为了解决这个问题，鸿蒙系统提供了“一次开发，多端部署”的能力，开发者开发多设备应用，只需要一套工程，一次打包出多个 HAP，快速上架，即可根据设备类型按需进行分发。

除了可以开发传统的应用，开发者还可以开发元服务。元服务是一种面向未来的服务提供方式，具有独立入口的、免安装的、可为用户提供一个或多个便捷服务的应用程序形态。鸿蒙系统为元服务提供了更多的分发入口，方便用户获取，同时也增加了元服务露出的机会。

1. 多设备按需分发

鸿蒙系统提供了两种模式帮助开发者基于“一次开发，多端部署”能力分发应用和元服务到不同设备上。

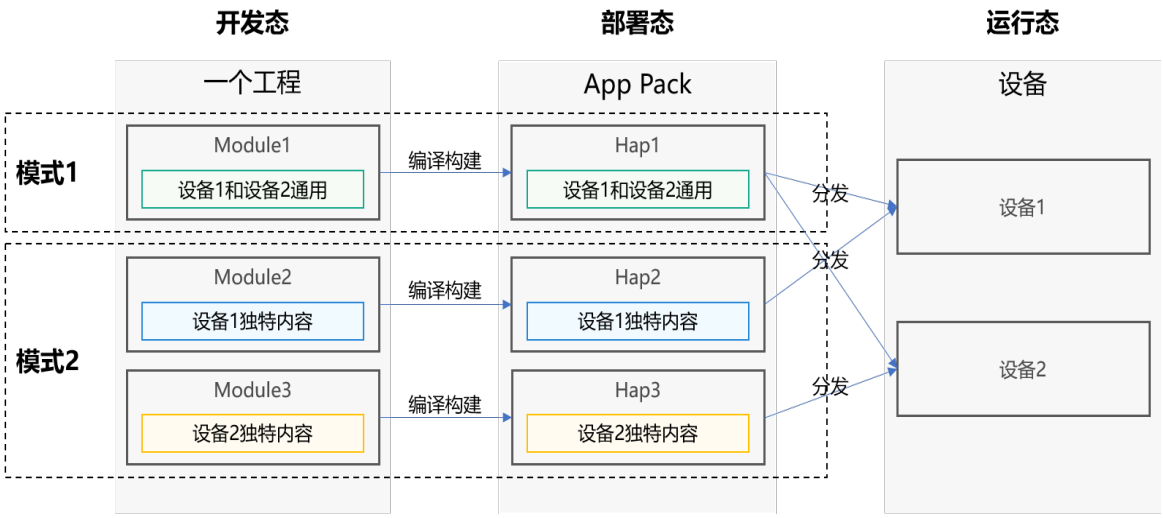


图 3-8：多设备按需分发的两种模式

- 模式 1：应用或服务的 UI 自适应不同尺寸的设备屏幕，并且在不同设备的功能相同，可以实现多设备共享一个 HAP 包。这种场景下建议开发者通过一个模块来开发，并配置该模块支持多设备，然后再编译构建生成一个 HAP，分发到不同类型的设备上运行。
- 模式 2：应用或服务的 UI、功能在不同设备间存在差异，无法实现 HAP 包多设备归一。可根据实际情况设置不同模块适用的设备类型，编译构建多个 HAP 包，一起上架。HUAWEI AppGallery Connect 会自动提取 HAP 中的设备类型的配置信息，为对应的设备自动分发正确的 HAP 包组合。

2. 多入口按需分发

鸿蒙系统为元服务提供了更多的分发入口，基于场景和用户意图拉起元服务，实现“服务直达”。鸿蒙生态提供的丰富入口如下图所示：

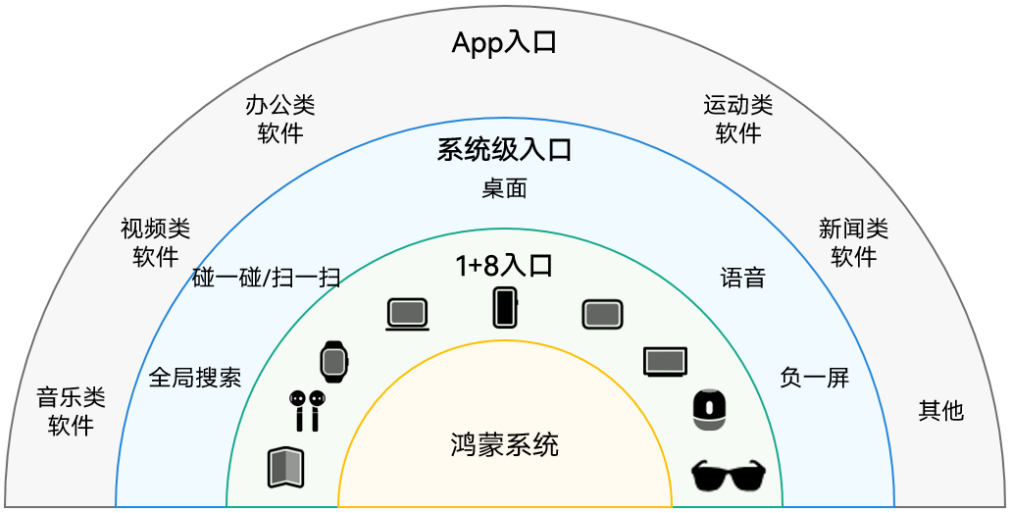


图 3-9：多入口按需分发

2）可分可合，自由流转

元服务是鸿蒙系统提供的一种全新的应用形态，具有独立入口，用户可通过点击、碰一碰、扫一扫等方式直接触发，无需显式安装，由程序框架后台静默安装后即可使用，可为用户提供便捷服务。

传统移动生态下，开发者通常需要开发一个应用版本，如果提供小程序给用户，往往需要开发若干个独立的小程序。鸿蒙生态下，鸿蒙支持元服务开发，开发者无需维护多套版本，通过业务解耦将应用分解为若干元服务独立开发，按需根据场景组合成复杂应用。

元服务基于鸿蒙系统 API 开发，支持运行在 1+8+N 设备上，供用户在合适的场景、合适的设备上便捷使用。元服务是支撑可分可合，自由流转的轻量化程序实体，帮助开发者的服务更快触达用户。具备如下特点：

- 触手可及：元服务可以在服务中心发现并使用，同时也可以基于合适场景被主动推荐给用户使用，例如用户可在服务中心和小艺建议中发现系统推荐的服务。
- 服务直达：元服务无需安装卸载，“秒开体验”，即点即用，即用即走。
- 服务卡片：支持用户无需打开元服务便可获取服务内重要信息的展示和动态变化，如天气、关键事务备忘、热点新闻列表。
- 自由流转：元服务支持运行在多设备上并按需跨端迁移，或者多个设备协同起来给用户提供良好的体验。例如手机上未完成的邮件，迁移到平板继续编辑，手机用作

文档翻页和批注，配合智慧屏完成分布式办公；例如分布式游戏场景，手机可作为手柄，与智慧屏配合玩游戏，获得新奇游戏体验。

可分可合

在开发态，开发者通过业务解耦，把不同的业务拆分为多个模块。

在部署态，开发者可以将一个或多个模块自由组合，打包成不同的 App Pack 独立上架。

在分发运行态，单个 HAP 作为元服务分发满足用户单一使用场景，也可以多个 HAP 组合为应用分发满足用户更加复杂的使用场景。

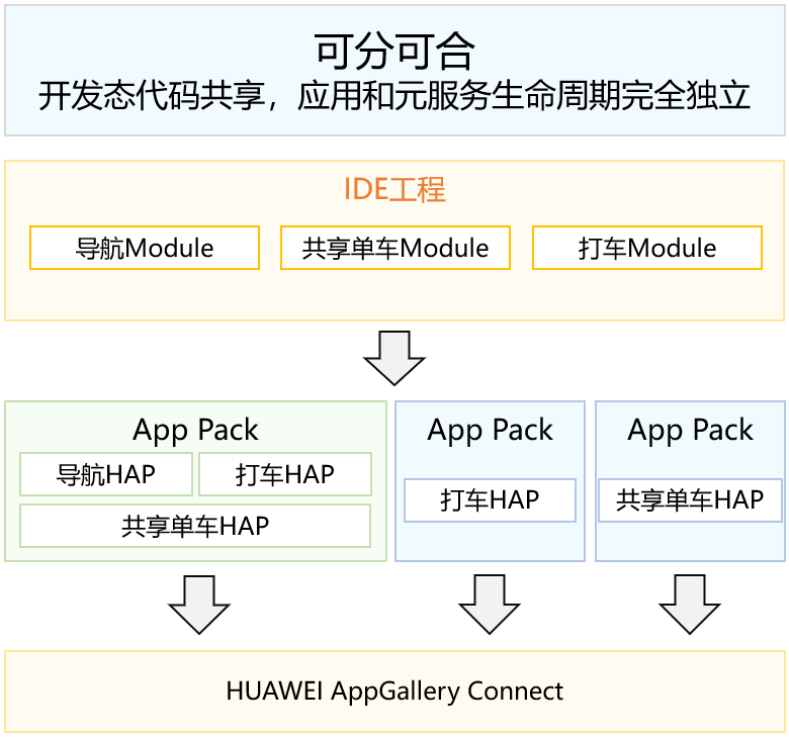


图 3-10：打包上架模式

自由流转

传统操作系统上的应用只能在单个设备内运行，多设备间的应用无法互通，当用户有多个设备且要跨设备完成任务时，体验上存在断裂点，数据不能有效同步。因此应用能够在设备之间流转，不间断给用户提供服务的能力就变得非常重要。

鸿蒙系统提供了自由流转的能力，使得开发者可以方便地开发出跨越多个设备的应用，用户也能够方便地使用这些功能。

自由流转可分为跨端迁移和多端协同两种情况。分别是时间上的串行交互和时间上的并行交互。自由流转不仅带给用户全新的交互体验，也为开发者搭建了一座从单设备时代通往多设备时代的桥梁。关于跨端迁移和多端协同详细说明，会在第七章中详细展开。

3）统一生态，系统智能

统一生态

统一生态具有愿景上的意义，打造智能联接，共建智能世界。

从设备角度来说，基于鸿蒙可以开发多种全场景终端设备；从应用角度来说，可以为鸿蒙开发多种应用，运行在全场景设备上，满足智能家居、智慧办公等全场景使用要求。为此鸿蒙系统提供全套能力来保证。

鸿蒙系统通过提供组件化、统一驱动框架、适配多芯片架构等能力，支持开发标准（如手机、平板等）、轻量（如 TV、手表等）和小型（如智能门锁等）三类设备，可覆盖各种智能终端。

传统设备之间的互联、互通和互操作是在应用层完成的，技术上没有问题，但实际中很难形成生态，不同厂家设备间很难互联互通互操作。鸿蒙系统提供统一的分布式组件、统一的模型、统一的互联业务互操作规范等能力和规范，在操作系统层面实现鸿蒙全场景终端设备的统一互联，彻底解决设备互联的生态难题。

鸿蒙系统通过提供 HarmonyOS SDK、IDE 和开发者服务，以及一次开发、多端部署，应用可分可合、自由流转、分布式服务等开放能力，让开发者实现一个工程、一套代码即可开发出覆盖多种设备的应用，而且通过操作系统的能力即可实现应用间互操作、跨设备流转等，真正做到应用开发出来即可融入整个生态。

系统智能

鸿蒙系统内置强大的 AI 能力，面向鸿蒙生态应用的开发，通过不同层次的 AI 能力开放，满足开发者的不同开发场景下的诉求，降低应用的开发门槛，帮助开发者快速实现应用智能化。

分层提供多样化的AI能力，满足开发者各类AI能力诉求

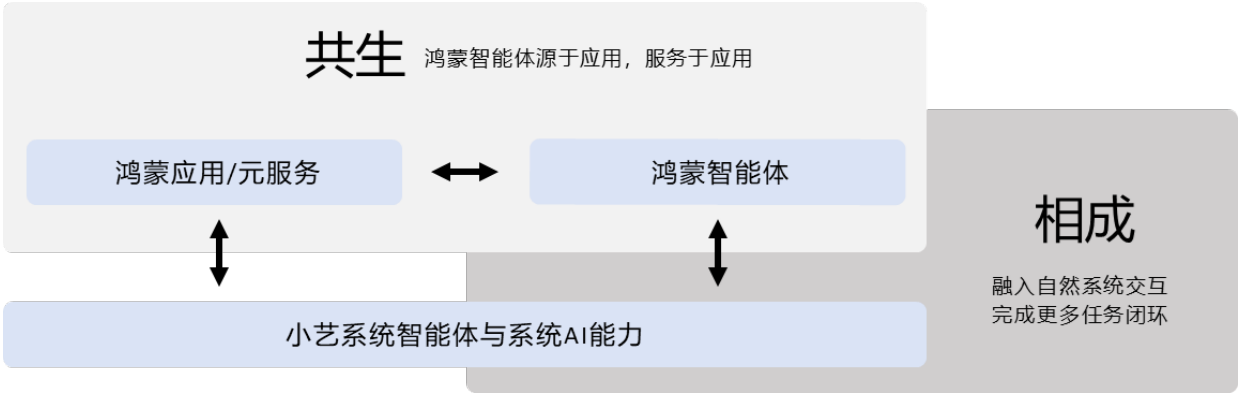
- 场景化 AI 控件：在 Speech Kit 和 Vision Kit 中为开发者提供高阶的、场景化的 AI 解决方案。包括朗读、文档扫描、卡证识别、活体检测、AI 字幕、智能荐图、智能填充等。

- 基础 AI 控件：将 AI 能力与系统基础控件深度融合，使系统控件具有文字识别、图像分割、实体识别等 AI 能力，降低开发成本。
- AI 基础能力：提供 TTS、ASR、OCR 等 AI 基础能力。
- AI 子系统：提供高性能低功耗的端侧推理和端侧学习环境，保证芯片能力高效有序提供。还提供大模型的相关能力。

意图框架提供了HarmonyOS系统级的意图标准体系，通过多维系统感知、大模型等能力构建全局意图范式，实现对用户显性与潜在意图的理解，并及时、准确地将用户需求传递给生态伙伴，匹配合时宜的服务，为用户提供多模态、场景化进阶场景体验。

传统基于LLM的智能体“可以自主地理解意图、规划决策、执行任务、调用工具，并具有记忆能力”，鸿蒙智能体还可以额外提供“与系统、应用和元服务无缝协作实现复杂任务，形成智能体价值网络”。鸿蒙智能体分为两类：

- 系统智能体：鸿蒙系统中有且只有一个系统智能体（即小艺），可以结合OS系统的底层能力，为用户提供体系化、可扩展的智能能力；OS 系统资源中的各项能力（感知能力、记忆能力、工具能力等）向系统智能体开放并由其管理。系统智能体像一位私人助理一样，一直关心、实时在线、适时服务，站在距离用户最近的位置，代表用户超前思考，为用户带来智能的服务体验。
- 领域智能体：类似于终端操作系统（HarmonyOS、Android、iOS）的系统应用和生态应用一样，除了系统智能体之外，下一代 AI 终端中还将存在多个领域智能体。系统智能体和领域智能体相互协作，为用户提供综合性与专业性结合的智能服务体验。由于当前头部的 APP 厂商拥有丰富的数据、强大的算法和充沛的算力，判断大量的领域智能体将首先由 APP 厂商孵化或转型而来；这种与 APP 存在派生与协同的领域智能体，也可以称为“应用智能体”，是鸿蒙生态的重要组成部分。


共生
鸿蒙智能体源于应用，服务于应用
鸿蒙应用/元服务
鸿蒙智能体
小艺系统智能体与系统AI能力
相成
融入自然系统交互
完成更多任务闭环

Chapter 4

鸿蒙生态应用开发能力全景图

- 1）赋能套件
- 2）鸿蒙开发套件
- 3）三方库
- 4）鸿蒙生态伙伴 SDK 市场
- 5）开发者支持平台

围绕开发者旅程，鸿蒙系统为开发者提供了端到端的开发能力支持。如下图所示，鸿蒙系统为开发者提供了赋能套件、鸿蒙开发套件、三方库、开发者支持平台。具体能力全景图如下图所示：

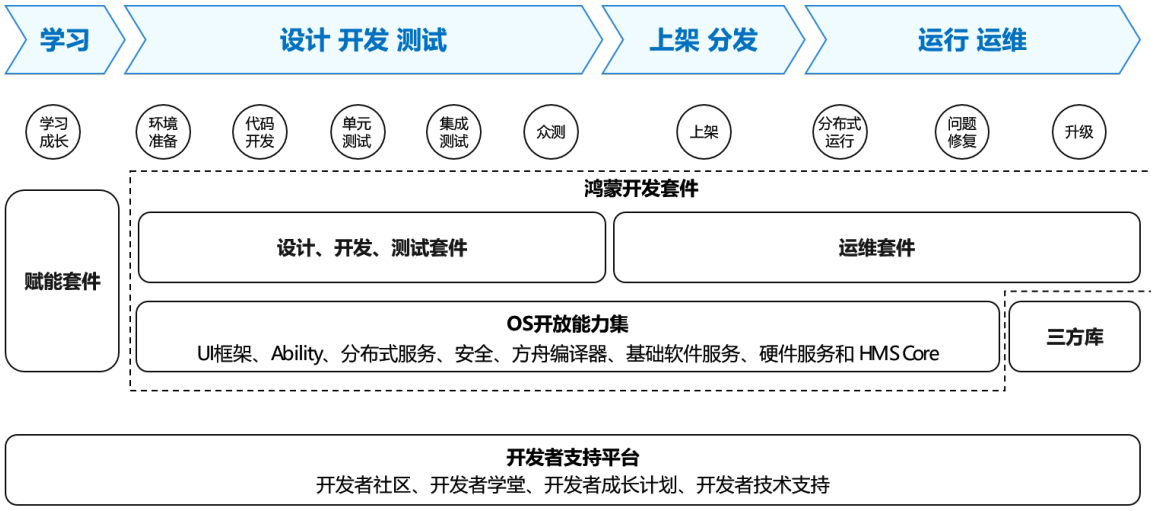


图 4-1：鸿蒙生态应用开发能力全景图

1）赋能套件

开发者了解和学习鸿蒙系统的各类资源，覆盖开发者全旅程，内容包括 HarmonyOS 第一课、开发指南、API 参考、示例代码、大型开源示范应用（HMOS 代码工坊）、AI 智能问答&FAQ 等。

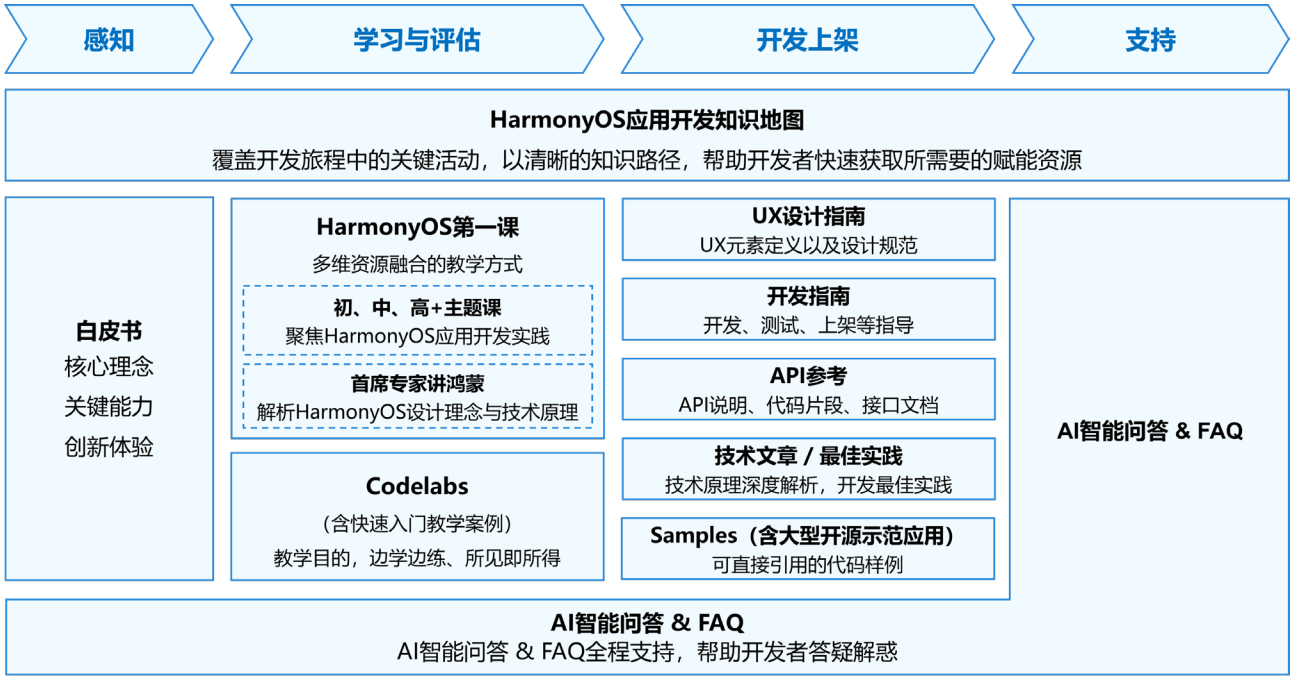


图 4-2：赋能套件全景图

鸿蒙生态白皮书：全面阐释了鸿蒙生态下应用开发核心理念、关键能力以及创新体验，旨在帮助开发者快速、准确、全面的了解鸿蒙开发套件给开发者提供的能力全景和未来的愿景。

HarmonyOS 第一课：基于真实的开发场景，提供向导式学习，即学即练，多维度融合课程等内容，给开发者提供全新的学习体验。包括初、中、高、主题课，以及首席专家讲鸿蒙系列课程。

Codelabs：以教学为目的的代码样例及详细的开发指导，帮助开发者一步一步地完成指定场景的应用开发并掌握相关知识。Codelabs 将鸿蒙生态应用开发技术与典型场景结合，让开发者能够快速地掌握开发高质量应用的方法。同时支持互动式操作，通过文字、代码和效果联动为开发者带来更佳的学习体验。

UX 设计指南：提供开发鸿蒙生态应用所需的 UX 设计规范、指导文档以及推荐的设计资源，满足各种场景的设计要求，可以帮助开发者设计出体验一致的鸿蒙生态应用。

开发、测试及上架指南：提供系统能力概述、快速入门，用于指导开发者进行场景化的开发。指南涉及到的知识点包括必要的背景知识、符合开发者实际开发场景的操作任务流（开发流程、开发步骤、调测验证）以及常见问题等。

API 参考：面向开发者提供鸿蒙系统开放接口的全集，供开发者了解具体接口使用方法。API 参考详细地描述了每个接口的功能、使用限制、参数名、参数类型、参数含义、取值范围、权限、注意事项、错误码、返回值及规范化的示例代码等。

技术文章：针对新发布特性及热点特性提供详细的技术解析和开发最佳实践。

Samples：面向不同类型的开发者提供的鸿蒙生态应用开发优秀实践，每个 Sample 都是一个可运行的工程，为开发者提供实例化的代码参考。

大型开源示范应用（HMOS 代码工坊）：华为官方出品的一款大型开源示范应用，该应用承载鸿蒙应用架构最佳实践，支持 1+8 设备运行，全方位体现鸿蒙应用的精致，流畅、智能、易用、安全、全场景互联等特点，并持续迭代鸿蒙新特性。其中内置集成数百个 Samples 示例代码，覆盖高频的鸿蒙应用开发场景，并支持源码的一键分享，给开发者提供所见即所得的样例代码、最佳实践，支撑开发者高效完成鸿蒙应用的开发。

AI 智能问答&FAQ：鸿蒙 AI 智能问答系统以“精准理解 + 可信生成”为核心设计理念，融合盘古大模型与 DeepSeek 技术能力，打造覆盖“问题解析-知识检索-答案生成”的

全链路智能化体验。开发者常见问题的总结，开发者可以通过 FAQ 更高效地解决常见问题。FAQ 会持续刷新，及时呈现最新的常见问题。

赋能套件旨在为开发者提供全方位的支持，帮助开发者更加轻松地进行 HarmonyOS 应用开发。开发者可以通过华为开发者官网和 IDE 帮助中心一站式获取 HarmonyOS 应用开发文档。

2）鸿蒙开发套件

鸿蒙开发套件包含设计、开发、测试、运维套件以及 OS 开放能力集。通过鸿蒙开发套件，开发者可以高效开发鸿蒙生态应用和元服务。

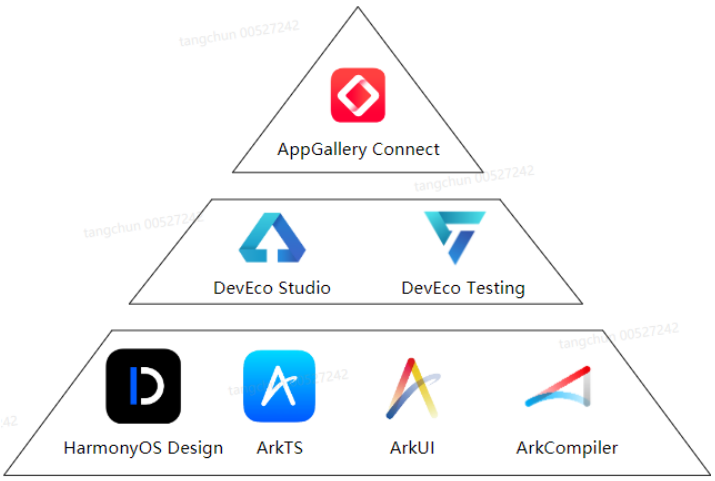


图 4-3：鸿蒙开发套件全景图

HarmonyOS 设计套件

Harmony Design 是面向全场景智能体验的设计系统，致力于构建一个和谐的数字世界，其秉承万物归一、和谐共生、衍生万物的设计理念，为用户带来优秀的交互体验。设计套件涵盖全场景多设备的家族化设计，其全栈式设计规范、丰富的设计资源、高效的设计工

具和插件库，以及垂类模板、体验标准等可以持续帮助开发者提升设计和开发效率，促进鸿蒙生态发展。

全栈式设计规范：包括设计理念、人因研究、应用架构、人机交互、视觉风格、动效、音效、振动、多态控件、界面用语、全球化、无障碍、隐私设计等。

丰富的设计资源：HarmonyOS 字体、HarmonyOS Symbol 和 HarmonyOS 音效库。这些资源旨在简化开发流程，加速设计调用，使开发者能够轻松实现个性化和差异化的设计效果。

高效的设计工具和插件库：在线设计工具、在线组件库、在线样式库、Symbol 插件、主题换肤插件、规范检查插件。

丰富的垂类模版：结合用户在多端设备上的历史交互习惯、各场景下的使用诉求等，进行了一些设计方法的总结，从基础要求、响应式布局、增值体验进行场景化设计。

应用 UX 体验标准：从影响用户体验的各个维度定义了相应测试规范，规定了应用需达到的基础体验要求，用于引导应用的设计与开发，以保证应用良好的使用体验，包含通用体验标准和多设备体验标准。

开发套件

开发者在应用开发过程中使用到的产品集合，包含 HUAWEI DevEco Studio 以及 HUAWEI DevEco Studio 集成的性能调优、设备模拟、命令行工具和 SDK。

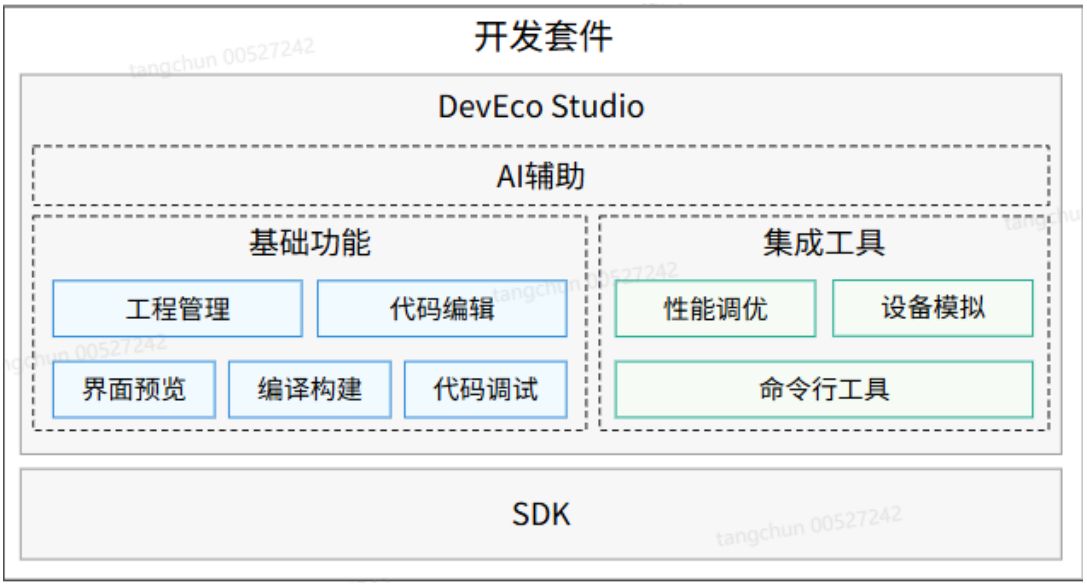


图 4-4：开发套件全景图

HUAWEI DevEco Studio：鸿蒙生态应用、元服务开发配套的集成开发环境（IDE），提供了工程管理、代码编辑、界面预览、编译构建、代码调试等基础功能，同时还集成了性能调优工具、设备模拟工具、命令行工具等帮助开发者解决特定领域的问题，并将智能化贯穿到开发流程中，提供代码续写、代码解释、页面生成、辅助问题定位及修复、辅助调优等AI 能力。

SDK：集成在 HUAWEI DevEco Studio 中，包含开发者可以使用的 API 定义以及调试编译等基础的工具链。

请访问 https://developer.huawei.com/consumer/cn/download/ 获取最新的HUAWEI DevEco Studio。

测试套件

包括测试标准和测试工具两个部分。

1. 测试标准

覆盖鸿蒙生态应用性能、功耗、稳定性、兼容性、UX、安全等测试规范，帮助开发者解决测什么的问题。

表 4-1：测试标准覆盖范围

标准分类	标准看护目的	标准看护关键内容
性能流畅	保障应用极致流畅体验，资源使用无异常。	时延帧率：无响应慢、卡顿等现象。 内容显示：应用启动过程中无黑白闪跳。 资源占用：内存和 CPU 占用峰值不超过规格。
UX 交互	保障应用良好的交互体验，页面精致美观、操作简单易用。	通用应用 UX 体验要求：界面布局合理、清晰、无遮挡、动效一致；导航栏、深色模式、实况通知等系统特性交互适配。 大屏设备应用体验要求：横竖屏、多窗适配体验；折叠、展开、悬停适配体验；键鼠、触控适配体验等。
稳定性	保障长时间稳定运行无故障，内存资源无异常。	运行无故障：应用在使用过程中不触发闪退、卡死等现象。 资源无异常：应用使用过程中无内存泄漏、踩内存、句柄和线程资源过载异常。
功耗	保障应用在前/后台不存在不合理行为，确保续航体验。	后台行为管理：后台蓝牙、网络、音频、GPS 等硬件资源合理使用，无长时任务则主动断链。

		前台行为管理：前台不可见动效减少过度绘制渲染、音视频类应用合理设置类型、CPU 等资源合理使用等。
兼容性&功能治理	保障应用在 OS/设备/升级的兼容性，并满足生态开发规范。	兼容性：应用在不同 OS 版本、不同设备、升级后无兼容问题，功能正常，历史数据可继承。 特性与功能规范（设计规范约束）：应用包体大小、参数设置、账号互通、卡片设置等满足生态规范要求。
纯净安全	保障应用全生命周期的纯净、安全、隐私体验，确保生态健康。	安全性测试：主要包含组件安全、WebView 安全、配置安全、签名安全等。 隐私测试：主要包含隐私政策信息和声明、权限申请、个人信息收集等。 纯净测试：主要包含恶意弹窗、隐藏误导、保活拉活、病毒植入等。

2.测试工具

提供鸿蒙生态应用开发、调试、单元测试、集成测试、上架测试等各开发阶段所需的测试工具集，支持手机、折叠屏、平板、智慧屏、手表、音箱等 1+8+N 设备，帮助开发者全面高效测试。

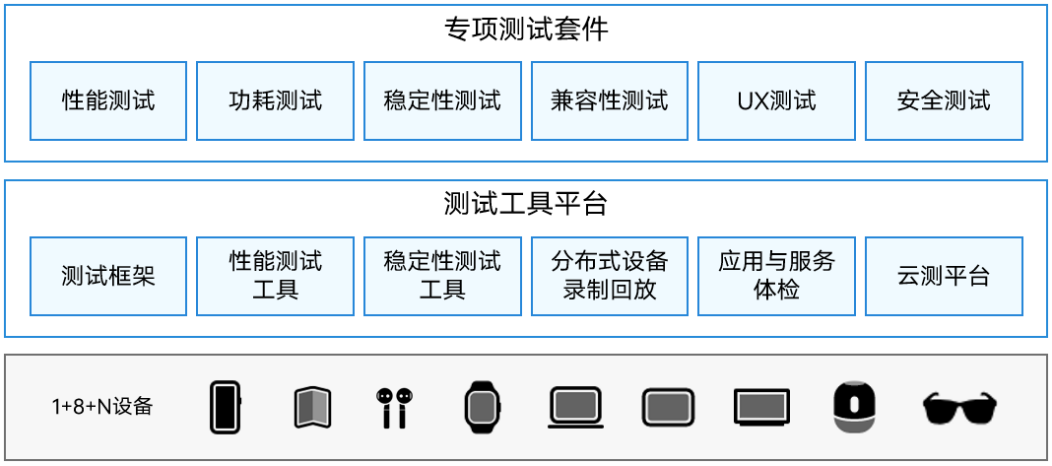


图 4-5：鸿蒙生态应用测试工具概览

典型测试工具能力简介见下表：

表 4-2：典型测试工具能力简介

测试活动	工具服务	主要工具能力
单元测试	DevEco Studio	提供单元测试基础框架，包括脚本开发框架、测试 mock 桩等。
功能场景测试	Hypium	提供应用 UI 集成测试基础框架，包括脚本开发框架、测试 SDK 等。
体验专项测试	DevEco Testing	本地端侧测试服务，主要提供应用性能、功耗、稳定性、UX、安全、兼容性测试服务等，对接本地手机/PAD 等测试设备环境，适合开发阶段在本地环境中使用。

测试套件获取途径如下表：

表 4-3：测试套件获取途径

名称	获取途径
测试套件	DevEco Testing 官网： https://developer.huawei.com/consumer/cn/deveco-testing/ 华为官网访问路径： https://developer.huawei.com/consumer/cn/doc/harmonyos-guides/app-testing-overview 社区官网访问路径： https://gitee.com/openharmony/docs/tree/master/zh-cn/application-dev/application-test

运维套件

主要包括由 HUAWEI AppGallery Connect 提供的上架分发测试和运维分析两大能力。

1. 上架分发测试能力

提供多种上架分发测试能力，满足开发者在不同阶段的上架分发测试诉求。具体如下表介绍：

表 4-4：上架分发测试能力介绍

分发阶段	简介
云测试/调试	快速获取目标机型，便捷远程测试，零脚本、低成本，通过自动化测试快速发现应用的兼容性、性能、稳定性、功耗、安全等问题，出具详细报告，复现与修复应用问题。
邀请测试	可以让开发者的应用在正式发布给所有用户前，面向特定用户群组发布测试版本。参与测试的用户可以向开发者反馈，帮助开发者及时发现技术问题或用户体验问题，以在应用/元服务正式上架前完成改进，从而在此过程中尽可能地降低对用户的影响。
全网上架	开发者在开发测试验证完成后，正式提交应用上架申请，审核人员审核通过后应用就会变为“已上架”状态，用户可在设备上搜索到该应用/服务。
分阶段发布	在当前上架版本为全网发布时，开发者可以采用分阶段发布的方式进行升级。采用分阶段发布，可以先向一定比例的用户发布更新的版本，然后再逐步提升用户比例，最终实现全网发布。通过小范围的版本更新，可以快速获取用户对新版本的反馈意见，降低全网发布后版本出现问题的风险。

2. 运维分析

提供崩溃服务、性能管理、智能分析服务及云服务监控，支撑开发者精准定位问题，同时支持多维度分析，智能诊断问题并给出解决方案。

表 4-5：运维分析能力介绍

能力名称	简介
崩溃服务	帮助开发者快速发现、定位、解决应用崩溃（又称闪退）问题。无需开发任何代码，即可实时查看可视化数据报告并检测到应用在每个设备上的运行状态，及时快速发现或者定位、解决应用崩溃问题，从而确保应用稳定运行，避免崩溃给用户带来糟糕体验。
性能管理	性能管理（APM，App Performance Management）服务提供分钟级应用性能监控能力，检测应用在每个设备上的运行性能数据，帮助开发者快速发现、定位、解决应用性能问题。
云服务监控	云服务监控是面向云函数、云数据库等云服务的质量监控解决方案，帮助开发者快速发现、定位、解决云服务的业务层性能问题。
智能分析服务	智能分析服务帮助开发者高效定位和解决应用运维态问题，通过将专家知识转换为自动化能力，根据日志进行根因分析。例如应用崩溃、应用卡死、卡顿等，并对每类问题进行聚类统计，同时提供故障的详细分析报告，让开发者可以快速地了解各类根因。

3）三方库

鸿蒙生态三方库，是在鸿蒙系统上可重复使用的软件库，可帮助开发者重用技术资产，快速开发鸿蒙生态应用和元服务，提升开发效率。根据不同的开发语言分为两种：

- ArkTS/TS/JS 语言的三方库，可直接导入使用。
- C/C++语言的三方库，在应用开发中通过 NAPI 的方式来使用。

鸿蒙生态三方库发布与使用完整的流程如下图所示：

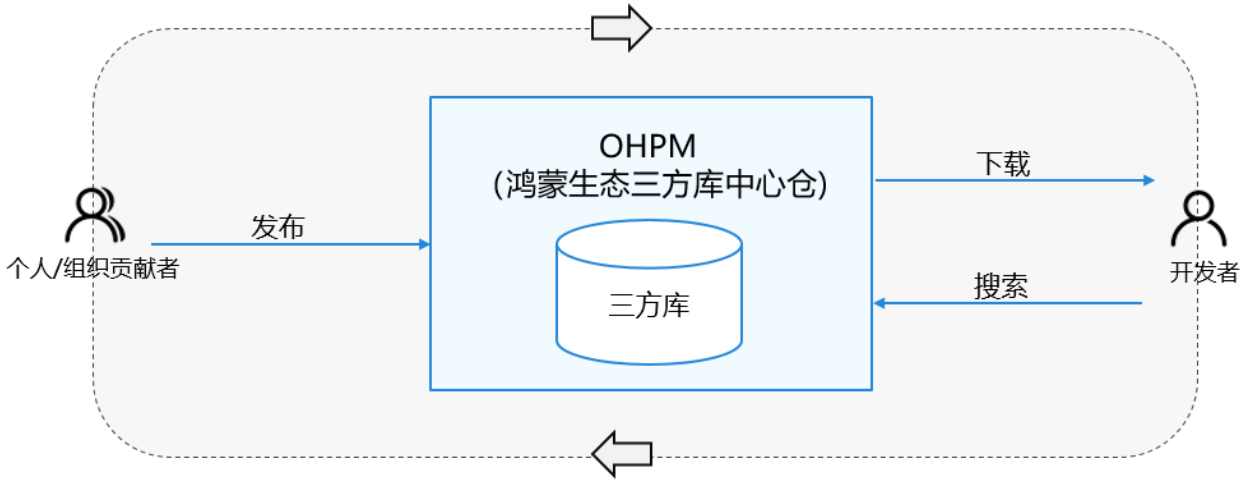


图 4-6：鸿蒙生态三方库管理

鸿蒙生态中心仓聚合了丰富的鸿蒙生态开发三方库，方便开发者一站式获取。个人/组织贡献者将开发好的技术组件库发布到 OHPM 中心仓。

开发者通过如下方式方便快捷的使用：

1. 应用开发者登录鸿蒙生态中心仓，通过分类和关键字搜索需要的三方库信息；

2. 应用开发者在应用开发时，通过 OHPM 包管理工具，将搜索到的三方库引入到应用依赖清单中。

4）鸿蒙生态伙伴 SDK 市场

鸿蒙生态伙伴 SDK 市场帮助开发者获得更优质安全的闭源 SDK，与 SDK 伙伴、开发者共建一站式的 SDK 选用平台，实现开发者、SDK 伙伴和华为共赢。鸿蒙生态伙伴 SDK 市场汇聚热门 SDK 助力开发者构建高品质鸿蒙应用。同时伙伴 SDK 市场通过 SDK 签名认证、安全检测/审核、SDK 上架发布等机制保障 SDK 的安全、纯净、可控。

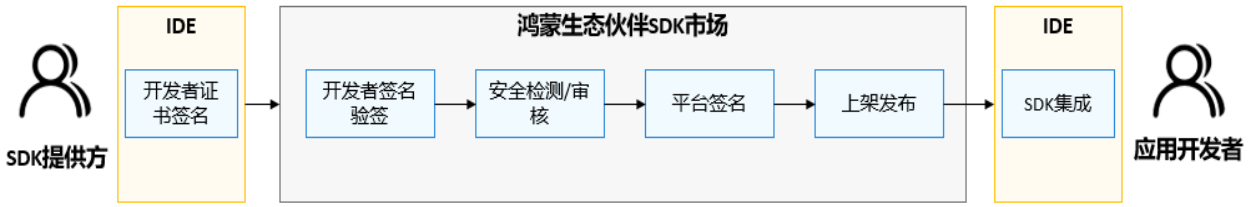


图 4-7：鸿蒙生态伙伴 SDK 市场

SDK 伙伴完成闭源 SDK 开发、签名后，可以提交到伙伴 SDK 市场。伙伴 SDK 市场对 SDK 进行安全检测/审核后，对 SDK 进行平台可信签名并发布到伙伴 SDK 市场。开发者可以在伙伴 SDK 市场高效便捷的获取 SDK，支持开发者通过 IDE 查看、一键集成 SDK，也支持到开发者联盟的伙伴 SDK 市场上查看、搜索、下载 SDK。

5）开发者支持平台

为了更好的支持、服务开发者，开发者官网提供了以下支持平台：

- 社区：开发者技术交流平台，包括专题文章和问答，给开发者提供发帖提问、技术交流与分享和获得开发者解答的平台。
- 智能客服：给开发者提供 7 x 24 小时自助式 AI 智能问答。
- 在线提单：提供人工技术支持和帮助，由技术客服为开发者提供问题处理和技术支持。

Chapter 5

高效开发与测试

- 1）典型开发场景
- 2）ArkTS 语言
- 3）ArkUI 框架
- 4）应用程序框架
- 5）鸿蒙开放能力
- 6）集成开发环境
- 7）测试工具

1）典型开发场景

基于鸿蒙系统开发框架，开发者可以根据自己实际的业务情况选择：

- 独立开发一个应用
- 独立开发一个元服务
- 同时开发应用和元服务

开发者可以选择开发简单、场景聚焦的元服务，渐进迭代演进，按需组合元服务成为一个复杂的应用。对于功能复杂、或者大型游戏类应用，可以直接开发鸿蒙生态应用。不论是选择开发元服务还是应用，均可以考虑提供服务卡片，当用户把卡片添加到桌面、负一屏等入口使用时，可以帮助开发者提供关键信息浅层级显示、核心功能页面一步直达的便捷体验。

2）ArkTS 语言

ArkTS 是鸿蒙生态应用开发高级语言，主打易学易用、生态丰富、极简开发的特征，围绕这些特征进行持续创新。

ArkTS 基于 TypeScript（简称 TS），保持了 TS 的基本语法和风格，同时通过引入静态类型校验模式和类型推断增强规则，强化开发期静态检查和分析能力，提升代码健壮性，并实现更好的程序执行稳定性和性能。ArkTS 同时也支持与 TS/JavaScript（简称 JS）高效互操作，可以完全复用 TS/JS 生态，已广泛应用于鸿蒙应用生态。

在标准 TS 的基础上，ArkTS 结合鸿蒙生态应用开发的诉求进行了创新和能力扩展，新增四大特性如下：

- 并发编程模型：ArkTS 新增提供了 TaskPool 和 Worker 两种并发编程 API 供开发者使用。同时，ArkTS 进一步提出了 Sendable 对象模型的机制来支持对象在并发任务间的引用传递，极大提升 ArkTS 对象在并发实例间的通信性能。

- 声明式原生语法：ArkTS 结合 ArkUI 提供声明式 UI 描述、状态管理、渲染控制等强大的 UI 开发能力，拥有简洁且富有表达力的语法，通过简洁的语法和实时预览功能，大大提高了 UI 开发的效率，使得代码更易于编写和阅读。

- 强大的标准库：ArkTS 拥有一个功能丰富的标准库，涵盖了从数据结构、算法到输入输出等方方面面，例如：高精度浮点运算、二进制 Buffer、XML 生成解析转换和多种容器库等丰富的操作方法，帮助开发者简化开发工作，提升开发效率。

- 模块化管理：ArkTS 支持应用模块化开发、编译、打包和运行，例如：应用模块化按需加载能力，方便大型复杂应用的多模块业务场景，高性能启动运行，提高了代码的模块化管理和重用性。

方舟编译运行时（ArkCompiler）支持 ArkTS、TS、JS 的编译运行，目前它主要分为 ArkTS 编译工具链和 ArkTS 运行时两部分。其中 ArkTS 编译工具链负责在开发侧将高级语言编译为方舟字节码文件（*.abc），而 ArkTS 运行时则负责在设备侧运行字节码文件执行程序逻辑。

ArkTS 会结合鸿蒙生态应用开发的需求持续创新，平滑演进。进一步丰富并发编程、完善类型系统、现代化语法等显著改进和新特性，使开发者能够更快速地构建稳定且性能优越的应用。

鸿蒙生态应用开发者可以从官方开发者网站中获取 ArkTS 语言介绍，快速入门 ArkTS 语言。对于熟悉 TS 语言的应用开发者，官方开发者网站也提供了从 TypeScript 到 ArkTS 的迁移指导，帮助开发者快速将已有 TS 代码重构为 ArkTS 代码。

3）ArkUI 框架

ArkUI 是鸿蒙生态的 UI 开发框架。主体结构如下图所示：

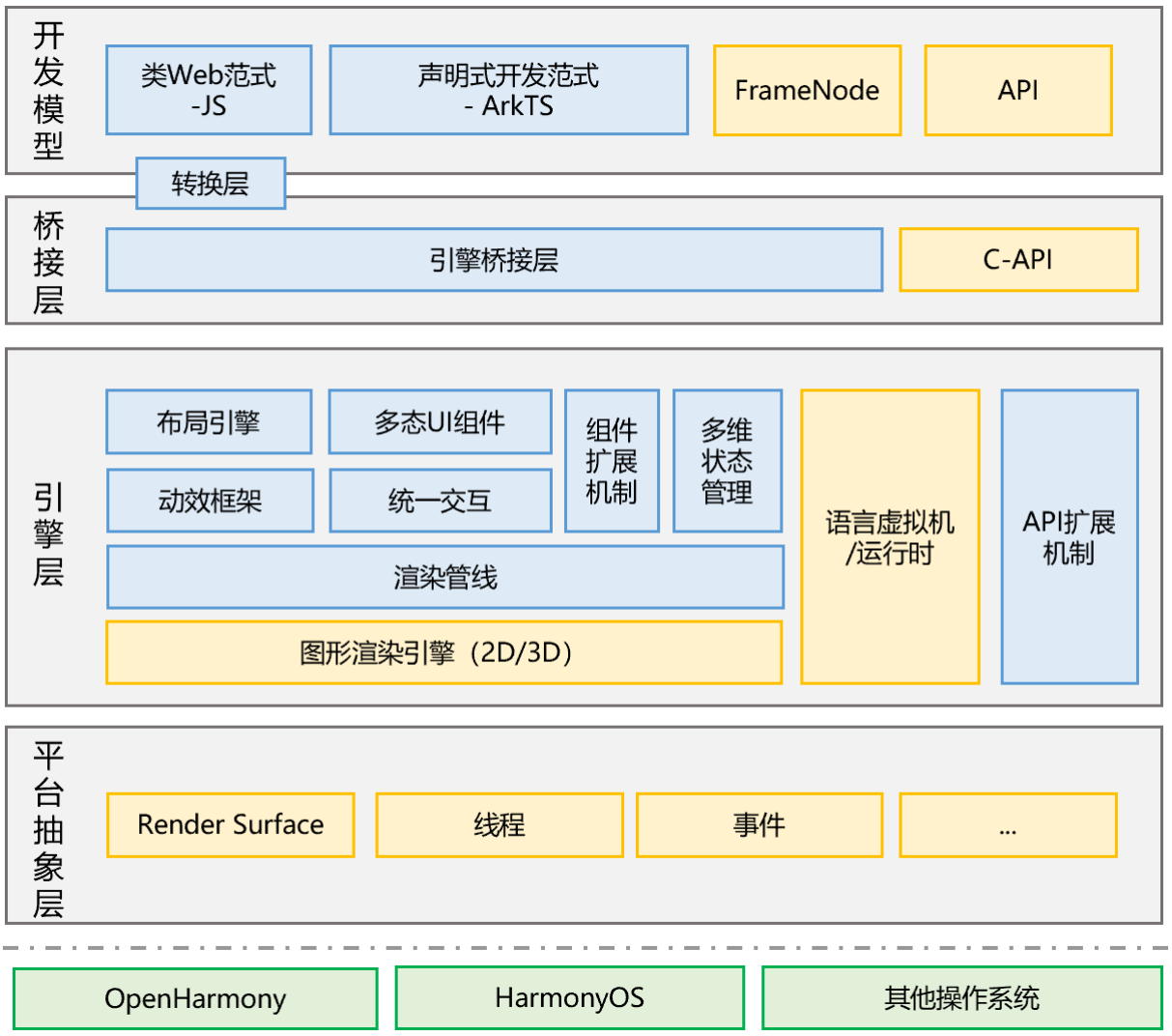


图 5-1：ArkUI 框架组成结构

ArkUI 框架提供给开发者多种开发方式：基于 ArkTS 的声明式开发范式，基于 JS 扩展的类 Web 开发范式，基于 C 语言的命令式开发方式。声明式开发范式更加简洁高效，C-API 开发方式适用于框架对接场景，同时提供了基于 ArkTS 的命令式 FrameNode 接口，作为对声明式范式的能力补充。

声明式开发范式

在声明式开发范式模式下，通过语言增强、渲染管线扁平化，最小化更新等手段，在功能和性能方面对比类 Web 开发范式有了全面提升。采用声明式开发范式进行应用开发，相同场景下，对比类 Web 开发范式代码更为精简，并且在性能、内存方面进一步优化提升。另外 ArkUI 框架还提供了 API 扩展机制，通过此种机制进行封装风格统一的 JS 接口。下面针对重点功能进行分别介绍说明。

1. 状态管理

声明式开发范式的核心思想是数据驱动 UI 变化，通过提供的状态进行数据管理，这里状态管理指的是，管理数据发生变化时，框架能自动更新这些数据关联的最小范围的 UI。

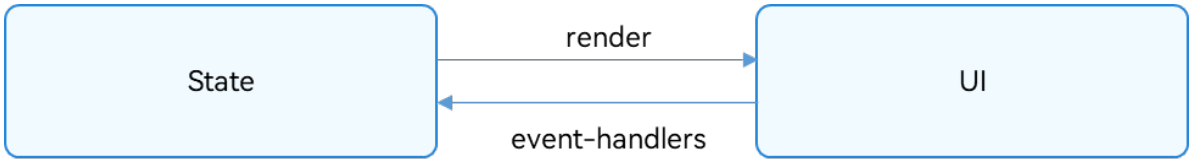


图 5-2：UI 与数据之间关系图

自定义组件中的变量，需要使用装饰器装饰，才能成为状态变量，状态变量的改变会引起 UI 的渲染刷新。如果不使用状态变量，UI 只能在初始化时渲染，后续将不会再刷新。

2. 布局

布局指用特定的组件或者属性来管理用户页面所放置 UI 组件的大小和位置。在实际的开发过程中，需要遵守以下流程保证整体的布局效果：

- 确定页面的布局结构。
- 分析页面中的元素构成。
- 选用适合的布局容器组件或属性控制页面中各个元素的位置和大小约束。

ArkUI 框架支持多种布局方式，如弹性布局、列表、宫格、栅格布局等。

3. 组件化

组件是 ArkUI 框架中的基础显示单元，一切 UI 显示的内容都是组件。ArkUI 框架提供多种开箱即用的 UI 组件，如文本显示、图片显示、按键等，并提供了面向多种设备形态的多态 UI 能力。另外，ArkUI 框架还提供了组合式扩展相应 UI 组件的机制，由 ArkUI 框架直接提供的称为系统组件，由开发者定义的称为自定义组件，其具有以下特点：

- 可组合：允许开发人员组合使用系统组件、其他组件、公共属性和方法。
- 可重用：自定义组件可以被其他组件重用，并作为不同的实例在不同的父组件或容器中使用。
- 配置化生命周期回调：生命周期的回调方法可以在组件中配置，用于业务逻辑处理。
- 数据驱动更新：由状态变量的数据驱动，实现 UI 自动更新。

4. 装饰器

自定义组件的场景中，通常会遇到需要动态传入不同的 UI 元素的情况，为了满足该场景 ArkUI 框架同时提供了动态构建 UI 元素的能力。

- @Builder：可通过 @Builder 装饰器进行描述，该装饰器可以修饰一个函数，此函数可以在 build() 函数之外声明，并在 build() 函数中或其他 @Builder 修饰的函数中使用，在一个自定义组件内快速生成多个布局内容。
- @Styles：声明式范式为了避免开发者对重复样式的设置，通过@Styles 装饰器可以将多条样式设置提炼成一个方法，直接在组件声明的位置使用。@Styles 装饰器将新的属性函数添加到基本组件上，如 Text、Column、Button 等，当前@Styles 仅支持通用属性。通过@Styles 装饰器可以快速定义并复用组件的自定义样式。@Styles 可以定义在组件内或组件外，在组件外定义时需在方法前添加 function 关键字，组件内定义时不需要添加 function 关键字。
- @Extend：为了满足开发者拓展系统组件的诉求，提供了@Extend 装饰器，可以将新的属性函数添加到系统组件上，如 Text、Column、Button 等。通过@Extend 装饰器可以快速地扩展系统组件。

5. 动效

声明式范式中一大特点体现在动效的使用上，与传统开发方式不同，声明式的动画是由数据变化驱动动画启动，而不再是直接控制动画的播放。UI 框架根据开发者的配置，自动地进行动画执行，根据动画场景不同进行如下分类：

- 属性动画：组件的某些通用属性变化时，可以通过属性动画实现渐变效果，提升用户体验。
- 显式动画：全局 animateTo 显式动画接口，指定由于闭包代码导致的状态变化插入过渡动效，开发者可以在事件回调中通过显式动画对指定数据变化增加动画效果。
- 转场动画：转场动画包括页面间转场、组件内过渡转场和共享元素转场三种，通过路由接口进行页面路由时，会触发动画的执行。

6. 事件交互

ArkUI 框架提供了很多交互事件，这些事件提供了不同的信息用于处理相关程序交互逻辑，目前提供了 UI 组件事件以下几类事件：

- UI 组件事件：由 UI 组件内置交互逻辑触发，不同的 UI 组件有不同的 UI 组件事件，比如 TextInput 输入框产生的 onEditChange 输入文本变更事件，List 列表组件产生的 onScrollIndex 列表项滚动事件，这类事件属于非冒泡事件（非冒泡事件指的是当一个组件上的事件被触发后，该事件不会向父节点传递）；挂载卸载事件，当 UI 组件挂载到 UI 组件树或者从 UI 组件树上卸载时触发，典型的场景比如通过 if 渲染语法控制 UI 组件的显隐状态，该事件属于非冒泡事件。
- 交互事件：点击事件、拖拽事件、焦点事件、触摸事件、按键事件、鼠标事件、手势事件等。

7. 绘制能力

ArkUI 框架提供两种 2D 自定义绘制能力。一种是通过图形组合的方式，利用布局、绝对定位和各种图形进行组合实现；另一种是通过绘制 API 在 Canvas 画布上进行绘制。

8. 混合开发

应用的场景是多样的，部分场景直接采用声明式 UI 组件组合无法满足诉求，例如游戏、地图这种需要依赖 C++ SDK 进行独立渲染，又或者开发相机、视频播放器这种需要独立纹理填充的场景，因此需要框架提供一种能够在C++侧进行自定义绘制的组件。ArkUI 框架提供了 XComponent 组件，支持加载应用动态库、NAPI 跨语言调用，进行 C++绘制能力的开发。

9. 跨平台

鸿蒙生态构建了 ArkUI 跨平台框架的核心设施，将相应的能力扩展到 iOS 和 Android 平台上，后续会进一步拓展到更多的平台。开发者可以通过一份代码，结合相应的工具链，同时生成多个 OS 平台的应用工程，并可编译出相应的应用程序，在相应的平台上高效的运行。

10. 分层能力开放

鸿蒙生态中，除了页面和框架本身跨平台能力外，应用还有很多场景是基于三方框架构建。因此，ArkUI 框架提供了 ArkTS 声明式范式、FrameNode 机制和 C-API 多层次开放的接口。其中基本应用界面可直接使用 ArkTS 声明式范式进行开发；三方框架可以使用 C-API

接口和 FrameNode 框架进行开发。针对不同框架特征，采取不同机制，达到更优秀的性能效果。具体差异如下图所示：

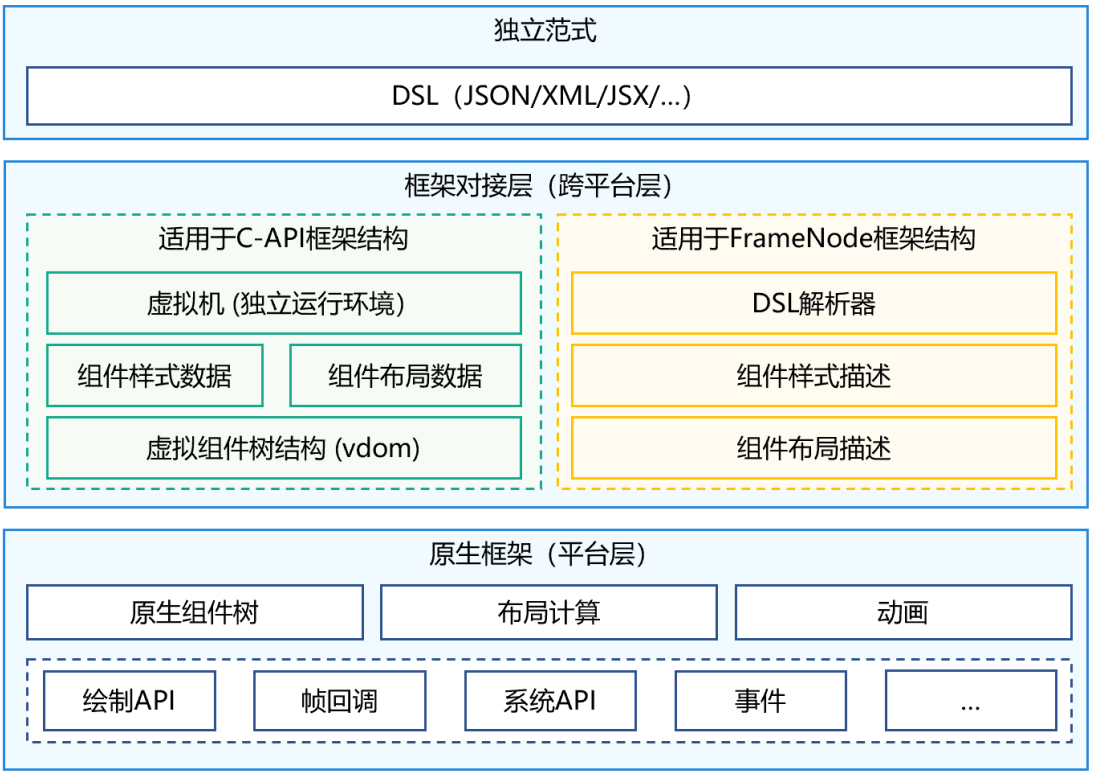


图 5-3：不同类型框架对比

针对图中两种不同类型框架对接，提供 C-API 和 FrameNode 两种机制，进行框架接入。当接入的三方框架存在 C 层数据处理情况或需要进行样式布局计算等逻辑操作时，针对该场景，ArkUI 框架提供了 C-API 接口，以减少跨语言带来的性能损失。

图中左侧适用 C-API 进行接入的框架具备以下特征：

- 具有独立的开发范式和运行环境
- 框架自行管理 UI 树形结构，具有虚拟组件节点
- 对接系统原生组件进行渲染

基于 ArkUI 框架进行页面开发时，存在需要进行页面内容的动态变化的情况，根据 XML 或者 JSON 动态生成页面。针对该场景，ArkUI 框架提供了 FrameNode 接口，满足动态创建、修改、删除组件的诉求。如图中右侧适用 FrameNode 进行接入的框架具备以下特征：

- 动态创建、添加子节点、删除、属性修改
- 支持内容自定义绘制

应用仅通过 DSL 解析后，直接对接原生组件进行渲染，该场景可以优先选用 FrameNode 进行页面构建。

使用上述机制，可以更细粒度的进行组件控制和减少跨语言带来的性能开销，保障三方框架高性能接入。

类 Web 开发范式

使用 HML 标签文件进行布局搭建，使用 CSS 文件进行样式描述，使用 JS 文件进行逻辑处理。UI 组件与数据之间通过单向数据绑定的方式建立关联，当数据发生变化时，UI 界面自动触发更新。此种开发范式，对 Web 前端开发者更为友好。

类 Web 范式的整体接口采用与传统 Web 页面开发相似的设计理念，采用 HML、CSS 与 JS 三种类型的文件进行页面开发，开发者可以基于此范式方便地进行 UI 构建，同时提供数据绑定机制，支持通过 JS 进行数据更新，进而更新 UI。

- HML 语法：是一套类 HTML 的标记语言，通过组件、事件构建出页面的内容。页面具备数据绑定、事件绑定、列表渲染、条件渲染和逻辑控制等高级能力。在 HML 文件中不仅可以进行架构描述，也可以进行数据绑定，通过{{}}方式进行数据绑定后，

也需要在 JS 文件中进行数据的定义，运行时将使用 JS 文件中提供的数据 content 进行替换。

- CSS 语法：CSS 是描述 HML 页面结构的样式语言。所有组件均存在系统默认样式，也可在页面 CSS 样式文件中对组件、页面自定义不同的样式。ArkUI 开发框架提供标准 CSS 语法的核心功能集，满足应用开发者的诉求。
- JS 语法：在类 Web 开发范式中，提供了一系列的全局方法与全局对象，进行数据操作与逻辑处理。

框架后端采用 C++开发语言实现，提升了框架的运行性能，使用方舟编译器运行时作为 JS 引擎，具有更优的 JS 执行性能，同时还提供了一套完整的包含 UI 组件、布局机制、动画能力的渲染框架，通过渲染引擎对 UI 元素进行绘制。

类 Web 范式实现层面可以进一步部署到轻量化的设备上。通过轻量化设计的思路，将 JS Framework 下沉到 C++层，以减小 JS 的内存占用，使用 C++进行更为严格的内存分配管理，并采用更为轻量的 JS 引擎，UI 部分采用轻量的 UIKit 并结合轻量图形引擎最终实现百 K 级别设备的支持，从而在轻量化设备上可执行的应用，也可以在硬件规格更高的设备上执行，而无需重新开发。这也就是采用类 Web 开发范式的优势所在，采用统一的开发范式，开发者无需关心具体运行时的前端框架、JS 引擎与后端 UI 组件，系统会根据运行平台不同，采用最佳的模块，保障应用在不同平台都可具有出色的运行性能。具体的实现原理如下图所示：

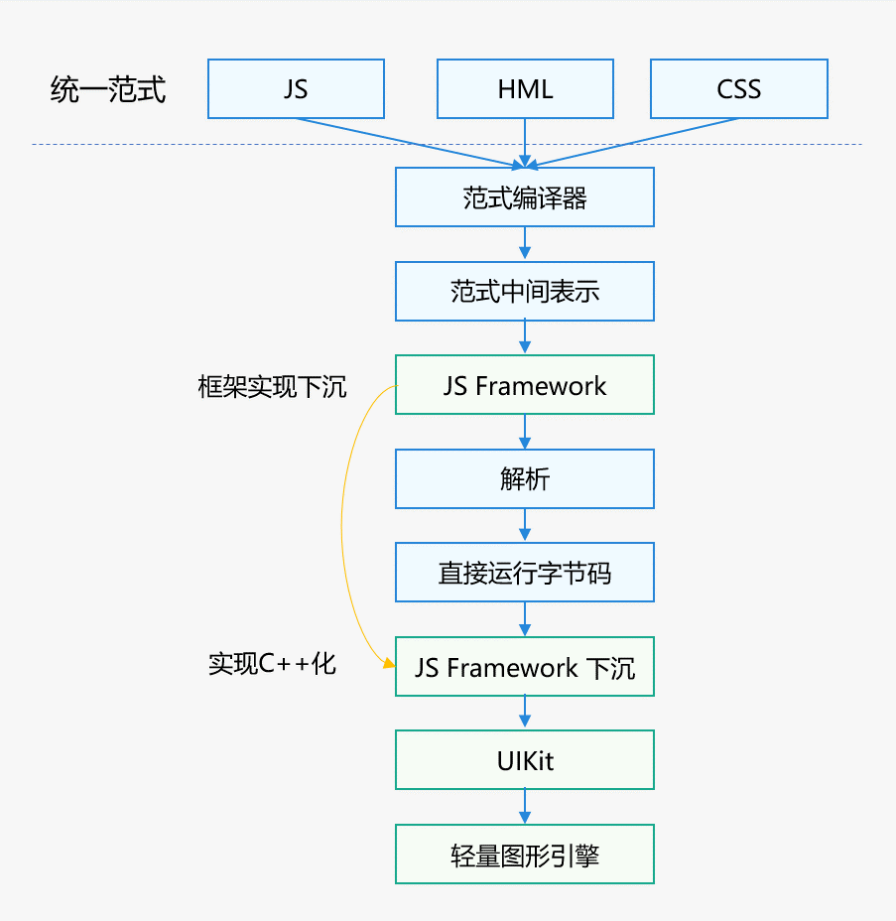


图 5-4：类 Web 开发范式

可视可说

可视可说框架提供“系统级”和“应用级”两种实现方式。其中，“系统级”无需应用适配自动支持标准控件文本的语音操控功能；“应用级”接入方式允许开发者对控件场景、角标、别名、个性化播报等元素进行适配，从而提供良好的用户体验。应用级和系统级两种实现是互补关系，应用级优化用户体验，系统级保证覆盖率。

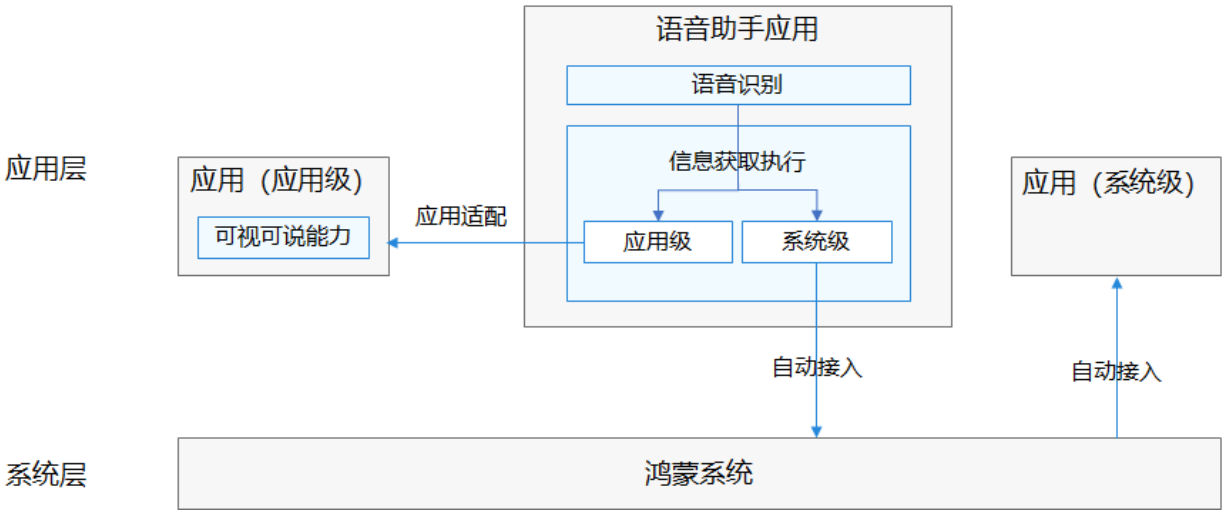


图 5-5：可视可说框架

1. 系统级

应用使用标准控件开发，无需额外适配，自动识别标准控件界面文本和位置执行，天然支持界面文本可视可说基础体验。

2. 应用级

系统级基础体验无法满足体验目标时，应用可以按照业务特征进行灵活定制适配，以此获得可视可说良好体验。

3. 语音交互生命周期

可视可说分为信息获取和识别执行，信息获取模块基于界面变化用户监听界面变化获取信息热词，识别执行模块将信息热词传递到语音系统进行 AI 识别。

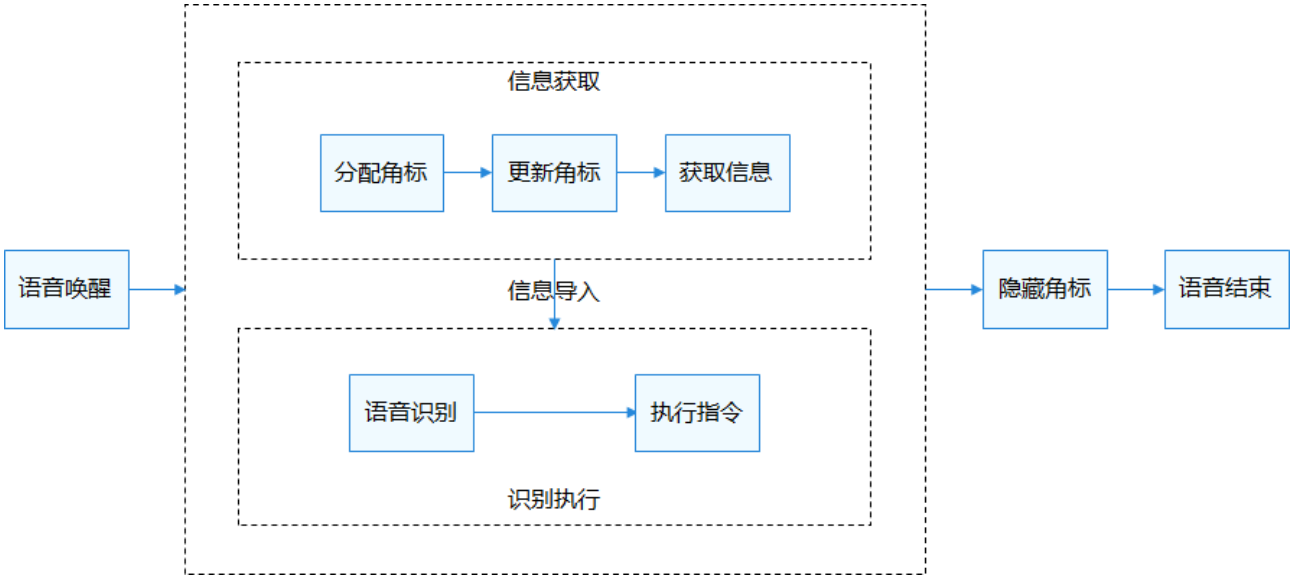


图 5-6：单次语音交互生命周期

- 语音唤醒：语音助手唤醒后，发送建立连接给前台应用，应用收到请求后启动可视可说初始化。
- 信息获取
  - ✧ 分配角标（可选）：应用反馈界面需要展示的角标数量给智慧语音。
  - ✧ 更新角标（可选）：界面变化时，自行管理展示角标刷新。
  - ✧ 获取信息：智慧语音获取界面热词信息，包括文本，图标和角标信息。
- 识别执行
  - ✧ 语音识别：智慧语音系统根据用户语音和界面信息热词，识别出可视可说意图和指令。
  - ✧ 执行指令：收到可视可说意图和指令触发执行。
- 隐藏角标：通知应用隐藏界面角标（隐藏角标动作也由适配应用自己完成）。

- 语音结束：语音助手释放可视可说，发送解除绑定给前台应用，应用收到解除连接后停止可视可说相关动作（比如角标未隐藏则隐藏角标信息）。

4）应用程序框架

应用程序框架定义了应用程序的模型与结构。鸿蒙系统上的应用模型称之为“Stage 模型”。应用程序框架定义了应用的全生命周期。鸿蒙系统是一个支持 1+8+N 多设备的统一操作系统，其生命周期的管理和定义就更为复杂且重要。Stage 模型主要特点包括：

- 规范化后台进程管理：为了保障用户体验，鸿蒙系统上的运行环境对后台进程进行了有序管理，当应用程序处于后台状态时，不应该处于高度活跃状态。为此，系统定义了四类后台任务：

  ✧ 短时任务：应用退到后台之后，系统提供了一个短期的可运行时间，任务将被冻结。适用于实时性要求高、耗时不长的任务，例如状态保存。

  ✧ 长时任务：适用于长时间运行在后台、用户可感知的任务，例如后台播放音乐、导航、设备连接等，使用长时任务避免应用进程被挂起。长时任务的类型是由系统定义的，应用应当根据实际需要来使用，不应当过度滥用。

  ✧ 延时任务：对于实时性要求不高、可延迟执行的任务，系统提供了延迟任务，即满足条件的应用退至后台后被放入执行队列，系统会根据内存、功耗等统一调度。

✧ 代理提醒：代理提醒是指应用退后台或进程终止后，系统会代理应用做相应的提醒。适用于定时提醒类业务，当前支持的提醒类型包括倒计时、日历和闹钟三类。

- 支持分布式：鸿蒙系统的诞生很大程度上就是为了解决多设备时代的交互问题。鸿蒙系统的应用框架从设计之初就包含对于分布式的考虑。
- 支持多设备的统一窗口管理：过去，面向不同的设备诞生了不同的操作系统。例如，面向 PC 设备的操作系统，面向移动设备的操作系统，以及面向穿戴设备的操作系统等。由于这些系统诞生在不同的时代，面向不同的屏幕形态，因此其窗口系统存在很大的差异。为更好的管理这些差异性，鸿蒙系统设计了统一的窗口系统，给开发者提供统一的编程模型。
- 组件共享及面向对象：多个应用组件在运行时共享同一个虚拟机引擎，从而减少复杂应用运行内存的占用。采用面向对象的开发方式，使得复杂应用代码可读性高、易维护好、可扩展性强。
- 逻辑与界面解耦：窗口部分可单独销毁和重建，窗口与应用组件可跨设备运行，应用组件可在不启动界面的情况下响应请求。
- 灵活扩展机制：支持服务卡片、输入法、延时任务等应用开发。

在鸿蒙系统中，Ability 是应用程序框架中最基本的抽象单位，是能够完成独立功能的应用组件。在 Stage 模型中，Ability 分为两大类：

- UIAbility：应用的主入口，对应桌面上的图标。一个 UIAbility 实例对应一个任务。一个 UIAbility 中的通常包含多个 ArkUI 页面。

- ExtensionAbility：ExtensionAbility 有多个具体的子类型，例如：FormExtensionAbility 用来开发服务卡片，InputMethodExtensionAbility 用来开发输入法等。ExtensionAbility 正如其名称那样，在鸿蒙新的版本，面向不同的设备，可能会持续扩展和增加。

下图展示了 Stage 模型的主要结构：

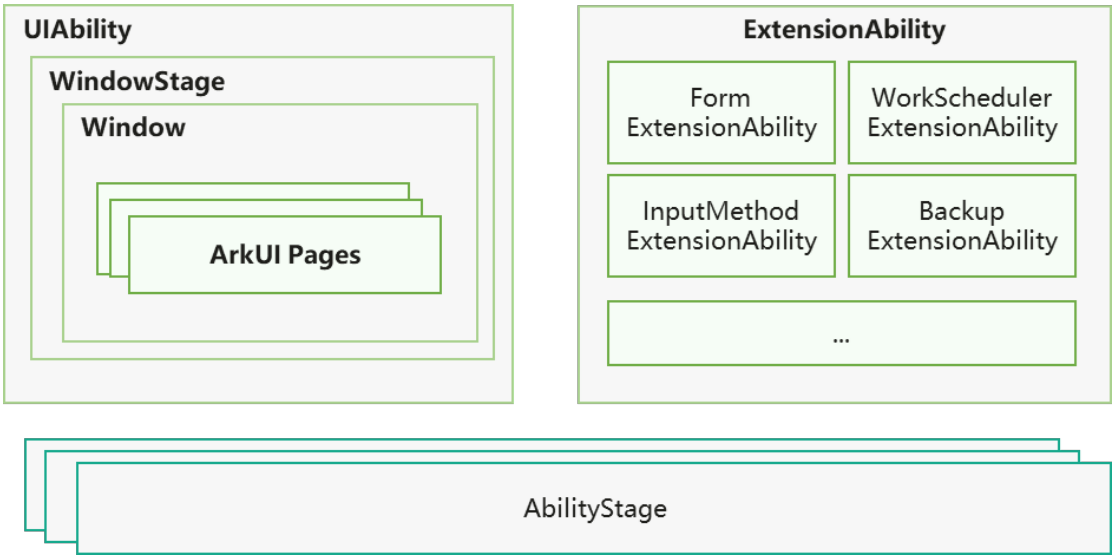


图 5-7：Stage 模型相关概念

针对大型应用开发，从开发态到部署态，鸿蒙系统都提供模块化解耦的方式，开发团队都是以模块化能够便捷的构建出可以复用的软件组件，也能够轻松的完成彼此间的分工协作。

开发者可以通过 HUAWEI DevEco Studio 工程中创建多个 Module，以 Module 为单位进行分工。每个 Module 既可以包含业务代码，也可以包含资源文件。针对不同的部署态需求，模块的编译结果有三种目标形态：

- HAR（HarmonyOS Archive）：这是一种中间编译产物格式，它最终将被编译合并到 HSP 或 HAP 格式的包中。

- HSP（HarmonyOS Shared Package）：这是一种新增的编译产物。HSP 使得模块可以以运行态复用的形式共享。相较于 HAR，当有多个 HAP 包依赖与同一个 HSP 时，最终的打包产物中，HSP 只会存在一份。
- HAP（HarmonyOS Ability Package）：HAP 包是鸿蒙应用可单独安装的容器包。借助鸿蒙统一开发框架的能力，同一个 HAP 包可以支持在多个设备上运行。当设备差异较大时（例如手机与手表），开发者可以为不同的设备设计不同的 HAP 包。在上架时，通过同一个 App 包来包含多个 HAP 包。这样可以达到一次开发上架，多端分发、部署的效果。

5）鸿蒙开放能力

为了在开发者界面提供清晰的开放能力分类，便于查询，学习和使用， HarmonyOS SDK 开放能力 API 按特定业务领域分类，归属到不同的 Kit 中。具体的划分为 6 大领域：

1、应用框架（APP Framework）：提供应用程序开发的基础设施，如运行时、引擎、框架等相关开放能力。

2、应用服务（APP Services）：提供应用程序开发的核心功能和服务，包括推送消息，认证登录，支付等开放能力。

3、媒体（Media）：存储、处理、传输视频、音频、图片等多媒体内容的相关开放能力。

4、图形（Graphics）：渲染、显示和交互的图形技术，包括 2D 和 3D 图形、动画、游戏等相关的开放能力。

5、AI（Artificial Intelligence）：机器学习、深度学习、自然语言处理、计算机视觉等相关的开放能力。

6、系统（System）：通信、安全、驱动程序、DFX、诊断和测试等相关的开放能力。

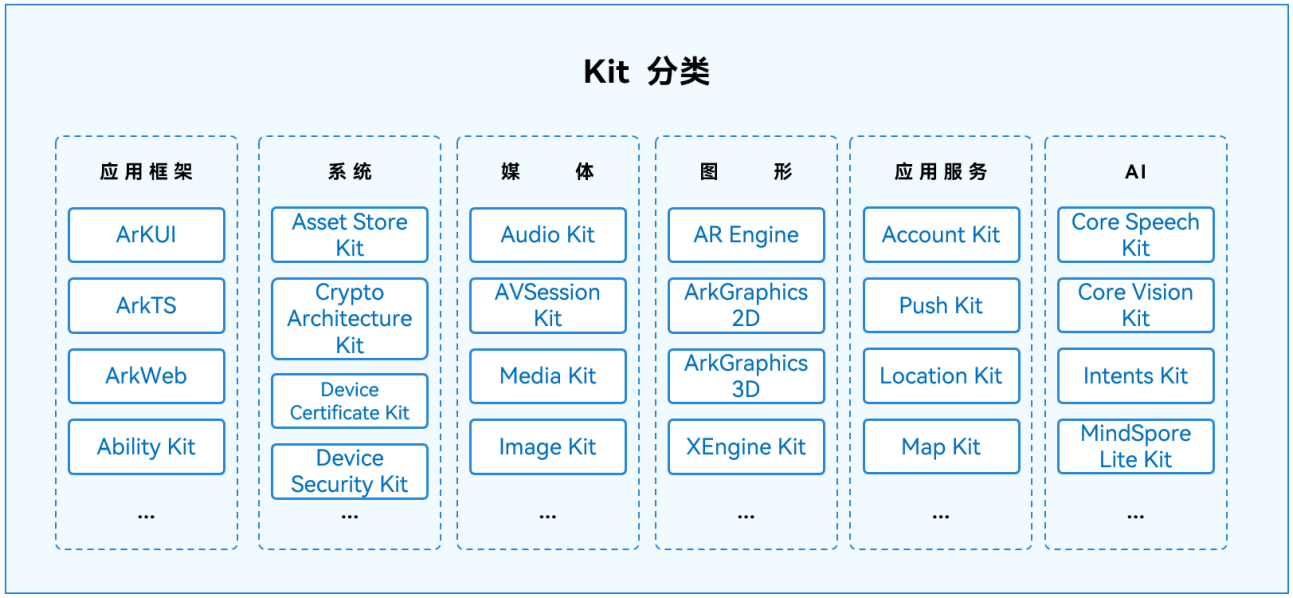


图 5-8：Kit 分类图

开放能力的检索和使用

在华为开发者网站上，开发指南和 API 参考，均以 Kit 的形式呈现，开发者可以查询某个 Kit 的相关资料。

在开发工具 DevEco Studio 上，提供 Kit 化导入方式，支持导入对象语法补齐，使用 Kit 开放能力更方便。

方舟工具链

传统的 JS 程序开发中，应用程序往往带的是经过前端打包工具处理过的 JS bundle 文件，在程序运行阶段进行解释执行；这种运行方式需要设备有强大的计算能力。鸿蒙系统能够支持的设备范围广泛，覆盖从低端的 IoT 设备到高性能手机设备。采用传统的方式，无法保证多类型设备的体验一致性。

在鸿蒙开发环境中，应用代码是通过前端编译器完成编译的。前端编译器按照语言规范解析源代码，编译成方舟运行时能够理解的二进制字节码格式（ABC，ArkCompiler ByteCode），最后打包到应用中。前端编译器是鸿蒙应用框架与其它 JS 应用框架主要的差别之一。下图展示了两种编译运行方式的差别，方舟前端工具链把解析源码、编译字节码的过程从运行时迁移到编译时，降低运行时的开销。

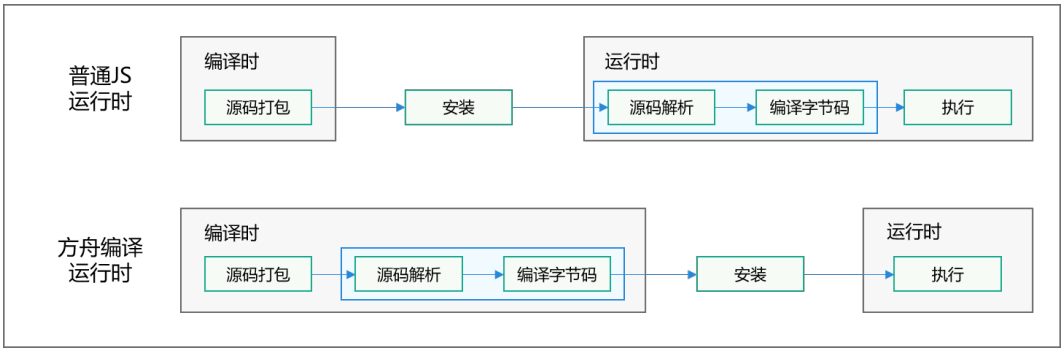


图 5-9：普通 JS 运行时与方舟编译运行时比对图

前端流水线在发起编译时，进行工程参数解析，依赖分析，语法校验，语法转换，代码编译等各个编译动作的编排。前端编译器负责编译流水线中源代码编译，提供对应的触发接口给编译流水线。

下图为鸿蒙生态应用的编译流水线流程：

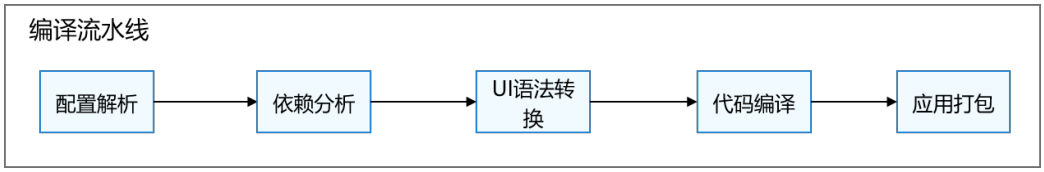


图 5-10：编译流水线

- 配置解析：解析 IDE 工程中的配置文件，解析程序组件，入口组件，组件包名，代码位置等信息。
- 依赖分析：根据代码中的 import 等语句，进行各个代码的依赖关系分析。
- UI 语法转换：对 UI 中的各种语法糖进行转换。
- 代码编译：输入转换后代码，解析编译，输出 ABC 字节码文件。
- 应用打包：获取应用的资源，ABC 字节码文件，应用配置文件等，使用用户签名进行应用打包，输出应用包。

前端编译器负责将 ArkTS 代码编译成方舟字节码 ABC，鸿蒙生态应用编译流程中，分为两种编译模式。分别是 bundle 和 esmodule 编译模式。两者的区别主要在源码文件的处理上，bundle 编译把各个有依赖关系的源代码通过打包方式打成一个 bundle 文件，然后通过前端编译器编译成 ABC 字节码文件；而 esmodule 编译是保持用户写的 ArkTS 模块不变，通过前端编译器编译成 ABC 字节码文件，字节码文件内保留各个模块的代码段，依赖关系等信息；当前推荐开发者使用 esmodule 模式，保持模块语义。

前端编译器架构

前端编译器是根据输入的 ArkTS 源码，进行词法，语法解析、转换、编译、输出字节码文件；在这个过程中会提取代码中标注的类型信息，进行类型检查，类型绑定，最终作为元数据生成到字节码 ABC 文件中。

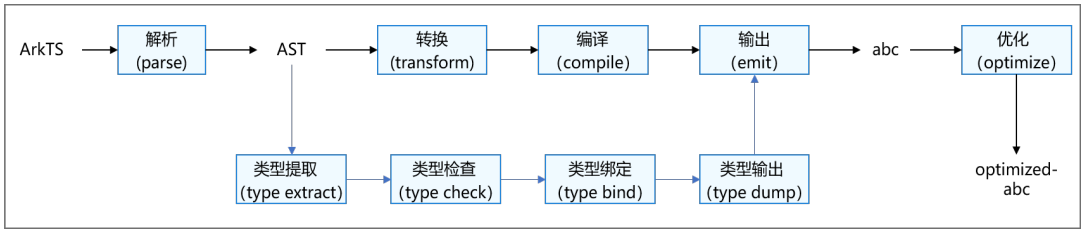


图 5-11：前端编译器架构

- 解析：前端编译器读取 ArkTS 源码，进行词法，语法解析，输出抽象语法树（AST）。
- 转换：前端编译器识别语法糖，转换成基础语法。
- 编译：根据抽象语法树，生成对应的中间表示（IR）。
- 输出：收集 IR，字符资源，常量，等各种元素，按照 ABC 文件格式生成字节码文件。
- 优化：读取 ABC 文件中的字节码信息，生成 IR 表示，进行优化处理，重新生成更优的字节码文件。

6）集成开发环境

HUAWEI DevEco Studio 是面向鸿蒙生态的集成开发环境，提供了一站式的鸿蒙生态应用、元服务开发能力，详细能力如图所示。

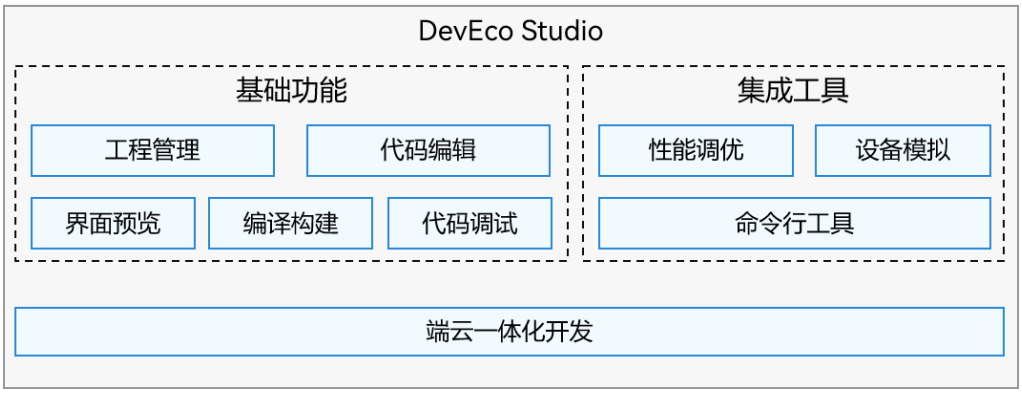


图 5-12：HUAWEI DevEco Studio 集成开发环境

工程管理

提供基础的工程管理能力，包括工程向导，工程模板，鸿蒙视图，样例导入等，并提供模板市场，支持扩展丰富的模板。开发者可以方便地安装和更新鸿蒙 SDK，利用模板创建鸿蒙生态应用、元服务，使用鸿蒙视图聚焦到关键文件及配置，也能导入样例快速学习了解鸿蒙 API 的用法：

- 工程向导：开发者基于模板，方便地创建出工程（Project），应用模块（Module），库模块（Library），Ability，服务卡片（Service Widget）等开发态元素，快速得到鸿蒙生态应用开发所需的项目结构。
- 鸿蒙视图：通过鸿蒙视图，可以过滤掉应用开发中无需特别关注的文件，如工具自动生成的文件，聚焦鸿蒙开发的代码文件及配置文件。
- 模板市场：模板市场提供了丰富的工程模板，支持模板的发布及更新，HUAWEI DevEco Studio 可以检测到新版本并更新。开发者也可以通过模板市场分享自己开发的工程模板，供其他开发者下载使用。
- 样例导入：样例提供了常用鸿蒙 API 的使用指导，开发者可以将样例工程导入到 HUAWEI DevEco Studio，学习常用 API 的使用，也可以基于样例工程快速开始开发。

代码编辑

面向鸿蒙生态应用、元服务开发场景，HUAWEI DevEco Studio 针对 ArkTS 语言及 ArkUI 框架，提供了代码补全、跳转、校验、重构、高亮、折叠、格式化等一系列编辑功能，辅助开发者便捷地阅读代码，高效地编写代码，实时地纠正代码错误。相较于传统的代码编

辑，HUAWEI DevEco Studio 还结合了人工智能技术，根据待补全位置的上下文代码特征进行预测和推荐，使补全项更精准，推荐内容更完整，帮助开发者可以更快速地完成鸿蒙生态应用、元服务开发。

界面预览

在开发过程中，开发者需频繁修改界面代码，查看对应的呈现效果，确保开发与实现目标一致。传统的开发模式下，开发者每次修改代码后，执行编译构建，并推送应用到设备上重新运行，才能查看到界面的呈现效果，整个过程冗长，产生较大的时间浪费。HUAWEI DevEco Studio 提供了界面预览能力，使开发者更方便快速地调测应用界面，大幅提升界面开发效率。

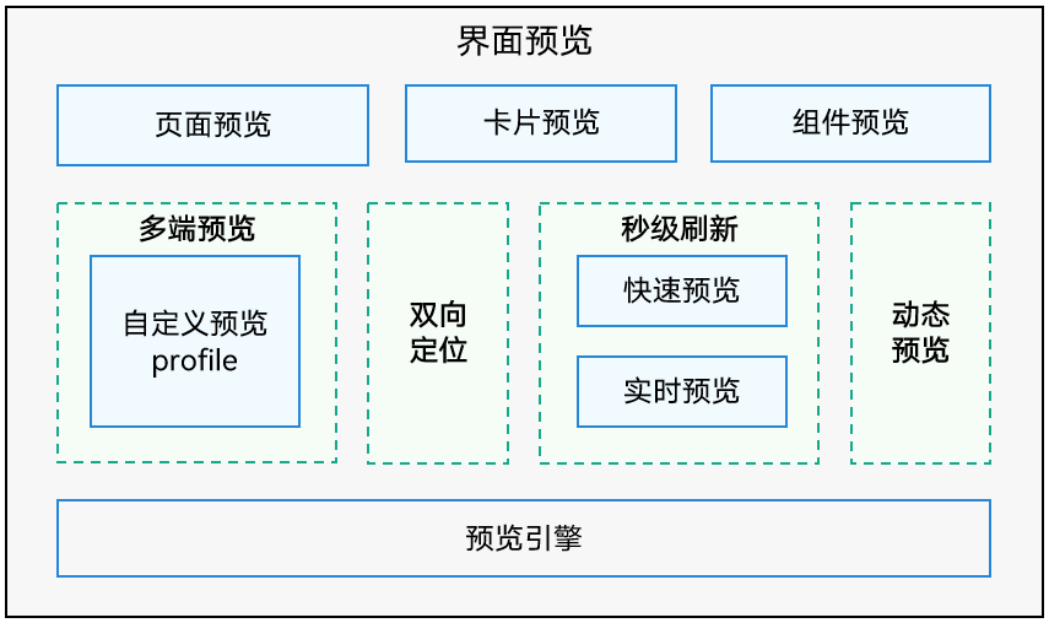


图 5-13：界面预览全景

- 页面预览：可快速查看应用/服务中 UI 代码的呈现效果。
- 卡片预览：可查看多种卡片规格、多种卡片尺寸（最小/标准/最大）的呈现效果。

- 组件预览：可独立查看组件的呈现效果，支持开发者注入组件参数，灵活查看组件在不同上下文中的预览效果。
- 自定义预览 profile：预览 profile 是设备显示能力的抽象定义，典型的 profile 信息有设备名称、 设备类型、屏幕分辨率、屏幕密度、语言、亮暗模式、横竖屏状态等。通过自定义预览 profile，开发者能自由组合设备显示能力，查看 UI 代码在不同设备上的呈现效果。
- 快速预览：UI 代码和预览效果可双向定位；组件属性修改，无需保存和编译，快速呈现效果；修改 UI 代码，保存后秒级刷新；预览界面支持点击、滑动、键盘输入等交互能力。

编译构建

HUAWEI DevEco Hvigor 是一款华为自研轻量级编译构建工具，将编译操作进行任务化管理，为开发者提供自动化的构建服务。其具备强大的构建能力，支持多种语言（ArkTS、C/C++等）、多种产物类型的快速编译，最终生成 HAR/HSP/HAP/APP 包。Hvigor 具备以下特点：

- 高效编译：充分利用系统资源，并行执行编译请求，提升编译效率；综合历史信息，精确增量检查，高度复用往次构建产物，缩短编译时长；优化编排任务序列，异步化执行编译操作，减小等待间隙，加速构建流程。
- 多目标构建：具有多目标构建能力，允许开发者灵活选择源码文件、资源信息、部署设备等应用要素，形成多种组合。通过简易配置，匹配自定义构建目标，一键打包生成不同产物，实现“一套代码，多种产物”。

- 灵活扩展：支持开发者自定义编写构建任务，匹配自身业务需求，扩展编译构建流程。
- 独立运行：拥有完善的命令行工具，兼备良好的跨平台能力，可以脱离 HUAWEI DevEco Studio，独立运行在 Windows、Mac、Linux 等操作系统上，支持不同环境下的流水线搭建。
- 可视化分析：构建全流程任务耗时数据展示能力，展示任务时长、任务线程、日志、历史记录等，通过可视化的方式查看构建任务流程，帮助开发者进行构建瓶颈优化。

代码调试

在开发过程中，代码调试是使用频率最高的功能之一，开发者可以使用断点跟踪或日志分析，快速定位代码缺陷。HUAWEI DevEco Studio 提供了常用的代码调试功能，如设置断点（普通断点、条件断点、异常断点、符号断点等）、断点跳转（Step Over/Into/Out）、变量值查询、表达式计算、调试堆栈、命令行工具等。此外，基于鸿蒙系统的特点，还提供了以下功能，进一步提升效率：

- 跨语言调试：支持 ArkTS 和 C/C++两种语言同时调试，并支持断点从 ArkTS 语言跳转到被调用的 C/C++语言。
- Hot Reload：修改代码后，无需重新创建调试会话和启动鸿蒙生态应用、元服务，即时生效，大幅缩短调试时间。
- 多维日志：查看系统消息日志时，可根据设备、进程、日志级别以及自定义的规则灵活过滤，快速筛选，协助定位代码缺陷。在多设备场景下，可以同时查看多个设

备的系统消息日志。其中特别重要的异常日志在独立的窗口呈现，避免淹没在大量系统消息日志中。

- ArkUI Inspector：支持显示视图的布局及组件关系，查看组件属性列表，过滤组件及属性，用于定位真机上的 UI 显示问题。
- 反向调试：支持通过反向单步，反向 continue 等操作，查看历史快照，线程，栈帧，变量等信息。通过反向时间线界面，查看多线程场景下的快照时序，用于定位偶现问题以及多线程并发的资源竞争等问题。
- 多线程问题智能检测：提供 ASan/TSan/方舟等多线程智能检测能力，一键式触发应用问题检测，提供详细的问题定位报告，支持源码跳转。

性能调优

应用的运行性能至关重要，一旦出现卡顿、发热、电量消耗过快等问题，便会导致体验急速下降，造成用户流失。性能调优是鸿蒙生态应用开发阶段中非常重要的一环，然而性能优化过程充满挑战，需要开发者了解应用程序框架、系统、硬件各方面知识，并对多维度性能数据进行综合分析。为了降低性能调优技术难度，HUAWEI DevEco Studio 推出了场景化调优工具 DevEco Profiler，提供以下关键能力：

- 场景化调优模板：针对各类典型场景的性能问题，提炼出对应的场景化调优模板，自动采集相应维度性能数据。常用场景化调优模板如表 5-1 所示。
- 模板自动推荐：根据实时监控观测到的性能异常事件，自动推荐对应的场景化模板。
- 高效数据分析：关联分析不同维度性能数据，结合同一时刻的代码调用栈，快速分析代码和性能问题之间的因果关系。

- 一键定位代码行：分析结果中代码堆栈并一键跳转至编辑器中的对应代码行。

表 5-1：常用场景化调优模板

模板名称	用途
Time Insight	耗时分析模板：通过周期性采集调用栈，识别 CPU 耗时高的热点代码段，用于分析卡顿、CPU 占用高、运行速度慢等问题。
Allocations Insight	内存分析模板：录制和分析内存分配记录，用于分析内存峰值高，内存泄漏，内存不足导致应用被强杀等问题。
CPU Insight	CPU 分析模板：录制 CPU 调度事件、线程运行状态、CPU 核频率、Trace 等数据，可用于分析卡顿、运行速度慢、应用无响应等问题。
Energy Insight	能耗分析模板：录制和分析能耗异常事件、硬件资源使用记录、功耗关联的系统状态，可用于分析电量消耗过快的问题。
Launch Insight	启动分析模板：录制和还原从点击应用图标，到显示首帧过程中的CPU、内存等资源使用情况，用于分析启动耗时长的问题。
Frame Insight	卡顿丢帧分析模板：录制卡顿过程中的关键数据，标注出应用侧、RenderService 侧卡顿帧，用于分析应用卡顿、丢帧的问题。
Snapshot Insight	快照分析模板：录制和分析应用程序中 ArkTS 对象的分布，通过快照方式对比 ArkTS 对象分布区别，用于分析内存泄漏问题。
ArkUI Insight	ArkUI 组件耗时分析模板：录制页面布局每一帧数据，标注出卡顿帧，用于分析组件耗时、状态变量更新导致的卡顿问题。
Concurrency Insight	任务并发度分析模板：统计展示出 ArkTS 侧、Native 侧任务并发度，以及它们的生命周期、吞吐量和时间消耗，用于分析并发任务卡顿问题。

模板名称	用途
Network Insight	网络分析模板：帮助用户在应用运行过程中查看 http 协议栈网络信息，包括请求分段耗时以及请求具体内容，方便对网络问题进行调优。
ArkWeb Insight	ArkWeb 分析模板：分析应用中嵌入 WebView 的 ArkWeb 加载和丢帧问题，结合 ArkWeb 执行流程的关键 trace 来定位问题发生的阶段。

设备模拟

HUAWEI DevEco Studio 提供了设备模拟的能力，解决鸿蒙生态应用、元服务开发过程中遇到的真机设备不足、GPS 定位刷新等应用场景难以复现等问题，为开发者提供低成本、易获取的调测验证环境。

多设备模拟：支持对手机、折叠屏、平板、PC 等多种设备进行模拟，针对不同模拟设备提供了差异化的交互界面，方便开发者快速在多个模拟设备上开发调试应用。

- 自定义屏幕参数：创建模拟器时可以对屏幕参数进行定制，除了预置的 mate、pura、nova 等机型外，还可以自定义屏幕参数，模拟未上市的机型。
- 多屏模拟：支持在单个模拟器上添加 3 块额外的屏幕，屏幕参数支持预置机型和自定义，可实现在单个模拟器上同时调试四种不同设备的 UI 效果。
- 丰富的器件模拟：提供了多种常用器件、外设、传感器的模拟，包括 GPS、WIFI、电池、麦克风、陀螺仪等，支持开发者调用模拟器件的能力，进行特定功能的开发。

- 场景化数据注入：通过场景化的数据注入能力，开发者能快速模拟一些常见的设备使用场景，方便调试应用在特定场景下的功能。包括低电量、网络代理、摇一摇、GPS 导航、户外跑步运动等场景。

命令行工具

HUAWEI DevEco Studio 提供了一系列命令行工具，辅助开发者更高效的开发：

- codelinter：支持代码检查及问题修复，用于检查代码规范、代码风格、安全及性能、最佳实践等。
- ohpm：鸿蒙三方库的包管理工具，支持共享包的发布、安装和依赖管理。
- hstack：支持将混淆后的 crash 堆栈还原为源码对应的堆栈，用于快速定位 release 应用的问题。
- hdc：管理设备、本地和设备之间传输文件、安装和卸载应用、启动和终止应用。
- bytrace：对内核 ftrace 进行了封装和扩展，配合应用打点，追踪进程轨迹，分析应用性能。

端云一体化开发

DevEco Studio 中提供了端云一体化开发的开发体验，开发者可以基于统一的技术栈，高效、协同地完成端、云代码的编写、调试、编译和部署，极大提高构建 HarmonyOS 应用和元服务的效率。

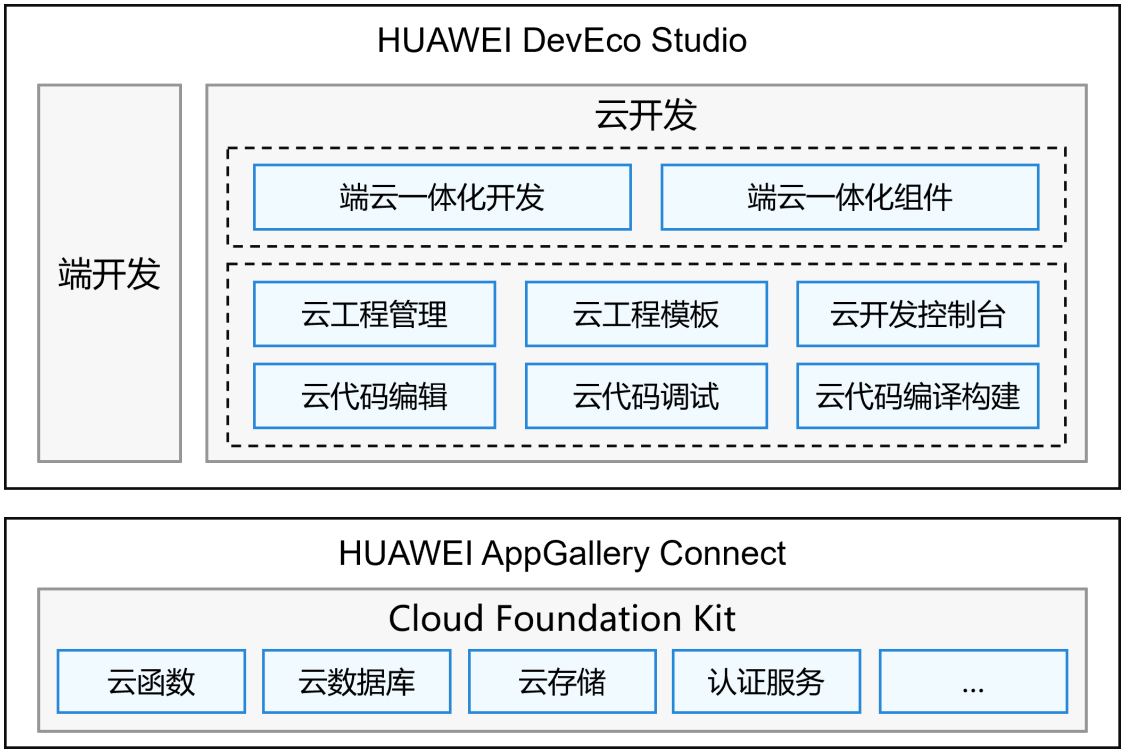


图 5-14：端云一体化开发全景图

- 端云一体化开发：支持端侧代码和云侧代码的协同开发，统一管理端侧和云侧代码目录，进行端云代码的端到端开发、调试和部署。
- 端云一体化组件：内置完整的云侧逻辑，开发者在集成 UI 组件的同时即可自动实现云侧逻辑，快速实现特定场景的功能。
- 云开发服务（Cloud Foundation Kit）：可以按需为应用提供云函数、云数据库、云存储、预加载等云端服务。应用运行所需的服务器和环境可以皆由云端平台提供，开发者只需关注应用的业务逻辑，而无需关心基础设施（例如：服务器、操作系统、容器等）
- 预加载服务：预加载是 Cloud Foundation Kit 提供的一种可提前加载所需资源的服务。通过预加载，可以将页面所需的文本、图片、音频、视频等资源数据提前加载到本地进行缓存，以提升应用页面加载速度。预加载仅以原始二进制数据进行缓存，

应用使用预加载时不需要修改原有数据格式，获取缓存后可直接进行解析，并且可以对隐私、敏感数据进行加密。

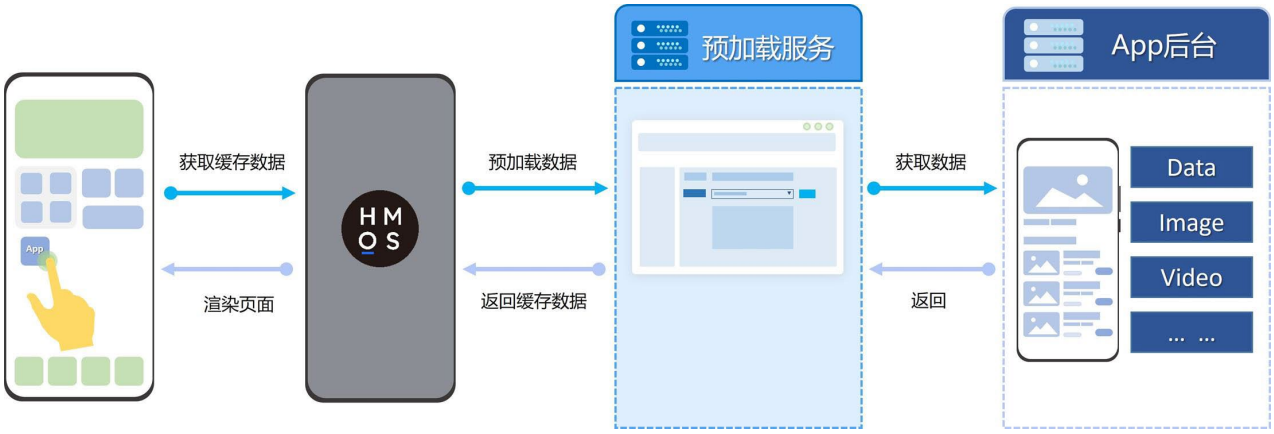


图 5-15：预加载流程图

7）测试工具

鸿蒙生态应用、元服务的测试分层模型分为：单元测试、集成测试、专项测试。

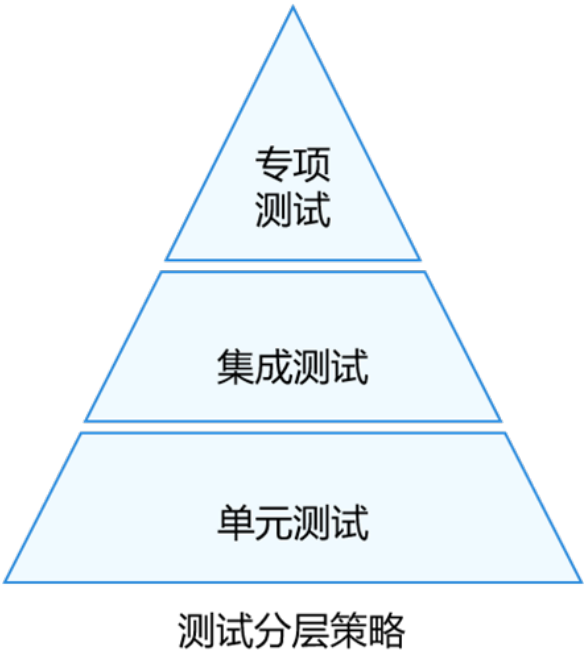


图 5-16：鸿蒙生态应用、元服务分层测试模型

单元测试

在软件开发中，代码质量是产品的生命线。单元测试作为保障代码质量的重要手段之一，其作用不言而喻。为了确保鸿蒙应用代码质量，建议单元测试覆盖以下内容：

- 界面业务逻辑层（ViewModel）：验证页面逻辑的正确性。
- 业务逻辑和数据层：验证业务逻辑、数据处理的正确性。
- 模块层：对于中大型应用，需验证多个模块接口的正确性，包括 HAP、HAR、HSP 等模块。

单元测试框架：

鸿蒙单元测试框架包括 Core 核心模块及 Ext 扩展模块（见图 7-2）：

Core 模块的功能包括：测试套、测试用例管理，预置、清理动作支持，断言能力，执行结果，执行调度等。

Ext 扩展模块提供了高阶的扩展能力，包括用例筛选、执行超时机制、数据驱动、mock 能力、随机执行、压力执行等能力，支持对 HarmonyOS 组件（Ability、Worker 等）进行单元测试。

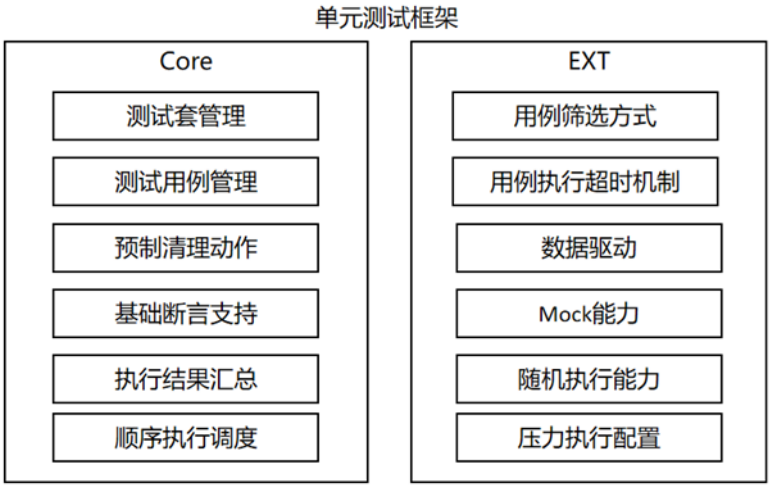


图 5-17：单元测试框架的主要功能

集成测试

集成测试分为模块测试和特性测试。模块测试把若干个单元组装，发现模块缺陷；特性测试把若干个模块集成，发现特性缺陷。鸿蒙生态为开发者提供多种集成测试的能力，方便开发者针对不同的集成测试场景，快速便捷的进行测试。

1 自动化测试工程

开展功能集成自动化测试需要使用目标系统提供的自动化测框架，鸿蒙官方自动化测试框架 DevEco Testing Hypium（下文简称 Hypium）面向开发者提供高效自动化开发和测试的能力。

2 UI 自适应自动化测试

鸿蒙 Hypium 测试框架除了支持通用的自动化测试用例开发之外，还提供了 UI 自适应自动化测试技术，解决了 UI 变化导致自动化脚本维护成本高的历史难题。

专项测试

专项测试是应用/服务的多维度测试，可以使用 DevEco Testing 测试工具辅助开展专项测试，提供了多项测试能力，包含性能、稳定性、功耗、安全、UX 交互体验、兼容性测试等服务，可用于开发自测、集成验证和回归测试等阶段。

图 5-18：DevEco Testing 专项测试工具服务

1 稳定性测试

稳定性测试通过持续模拟真实使用场景中的长期运行状态，验证应用程序是否能够在不同负载条件下保持连续可靠的运行能力，其核心价值不仅体现在预防软件崩溃、界面卡顿等直接影响用户操作的显性问题，更在于深入检测程序运行过程中逐渐累积的潜在隐患。

详细的稳定性测试可以参考鸿蒙开发者官网《应用稳定性体验建议》开展测试，重点包含两部分内容：

- 应用运行异常：应用运行稳定无闪退卡死等异常，且出现异常后能进行自恢复。

- 应用资源异常：应用运行过程中无内存泄漏、内存越界、句柄/线程过载等。

2 性能测试

性能测试看护应用及时响应、运行流畅体验，在使用过程中时延、帧率和资源占用是三类会对应用用户体验产生负面影响的指标。DevEco Testing 提供了自动和自定义两种检测能力，可帮助用户快速检测性能体验指标。

为系统开展应用性能体验测试，可以参考鸿蒙开发者官网发布的《应用性能体验建议》，重点测试以下三类性能体验：

- 时延类：代表用户操作响应和完成效率
- 帧率类：代表用户操作的流畅性，涉及应用内点击/滑动等操作
- 资源占用类：代表用户完成操作所消耗的资源

3 UX 测试

鸿蒙应用开发中的 UX 测试通常是指针对用户界面（User Interface，UI）的交互体验测试，UX 测试不止关注应用图形界面是否符合设计规范，更注重用户的使用感受，确保用户使用时能够获得良好的体验，并且所有的界面元素都能正常工作。

为了系统的开展应用 UX 测试，可以参考鸿蒙开发者官网的《应用 UX 体验建议》重点测试以下内容：

- 基础体验：界面文字、图片及各控件在多设备上布局合理，无遮挡，截断，显示不

全等现象；应用的桌面图标和应用内图标尺寸、清晰度满足要求等。

- 视觉风格：界面内容的色彩比对度合理，字体大小满足要求；深色模式下，应用页面内容展示正常，色彩比对度合理等。
- 交互易用：应用所有界面都支持返回操作，应用手势时长满足于要求，热区大小合理，元素可以正常响应等。
- 系统特性适配：底部导航条和顶部状态栏应用适配显示正常，无遮挡、背景不融合等现象；实况窗和通知符合设计规格，遵守通知规范等。
- 动效：相邻界面切换有动效过渡，同时确保跟手和离手动效的一致性等。

4 功耗测试

功耗测试旨在验证应用使用过程中的功耗体验，避免应用因高耗电导致设备快速发热、续航缩短。功耗测试主要关注应用全生命周期是否对软硬件资源合理使用，进而提升用户功耗体验。

为了更好的开展应用功耗测试，可以参考鸿蒙开发者官网《应用功耗体验建议》，主要涉及前台和后台功耗管控两种场景：

- 后台场景指的是当应用未处于设备屏幕顶层时，用户无法直接看到或进行操作的场景，应用需满足最基本的后台功耗体验要求，如应用退后台对蓝牙、麦克风、定位等器件的合理使用，持锁资源的及时释放，以及申请长/短时任务的应用退后台后CPU 负载约束等。

- 前台场景指的是当应用处于设备屏幕的顶层时，用户可以直接看到并进行操作的场景，应用需满足最基本的前台功耗体验要求，如不可见动效及时停止刷新，音乐类、导航类正确配置应用类型，正确使用平台的视频硬件编码器等。

5 安全测试

安全隐私测试是在软件产品开发基本完成时，验证产品是否符合安全需求定义和产品质量标准的过程。其目标是通过对系统进行全面的安全测试，发现系潜在的安全隐私隐患并提出相关建议，确保系统的安全性和合规性。

可以参考开发者官网发布的《安全隐私体验建议》，对鸿蒙应用进行全面的安全测试，主要包含以下内容：

- 安全性测试主要包含组件安全、WebView安全、配置安全、签名安全等内容。
- 隐私测试主要包含隐私政策信息、权限申请、个人信息收集等内容。
- 纯净测试主要包含恶意弹窗、隐藏误导、保活拉活、病毒植入等内容。

6 兼容性测试

兼容性测试旨在确保应用在不同操作系统、设备类型（手机、平板电脑、PC 等）及网络环境下的稳定运营和良好使用体验。其主要关注点包括应用在安装、启动、运行、卸载过程中的稳定性，也要确保应用的功能完整性，数据交互的准确性，进而满足用户使用应用的基础体验要求。

为了尽早发现问题，提升用户使用体验，可以参考鸿蒙开发者官网的《应用基础功能和兼容性体检建议》展开兼容性测试活动，主要包含以下内容：

- 系统特性与基础功能设计约束：鸿蒙应用在开发构建和系统交互时应该遵循的设计规格要求。
- 基础兼容性：鸿蒙应用在不同 OS 版本、应用版本及设备类型上的基础兼容性体验要求。

Chapter 6

快速上架与多端分发

- 1）快速上架
- 2）应用分发
- 3）服务分发

HUAWEI AppGallery Connect 为开发者提供全球化、全场景一站式应用分发能力，并为开发者提供质量、安全、工程管理等领域的能力，大幅降低应用开发与运维难度，提升版本质量，帮助开发者获得用户并实现收入的规模增长。

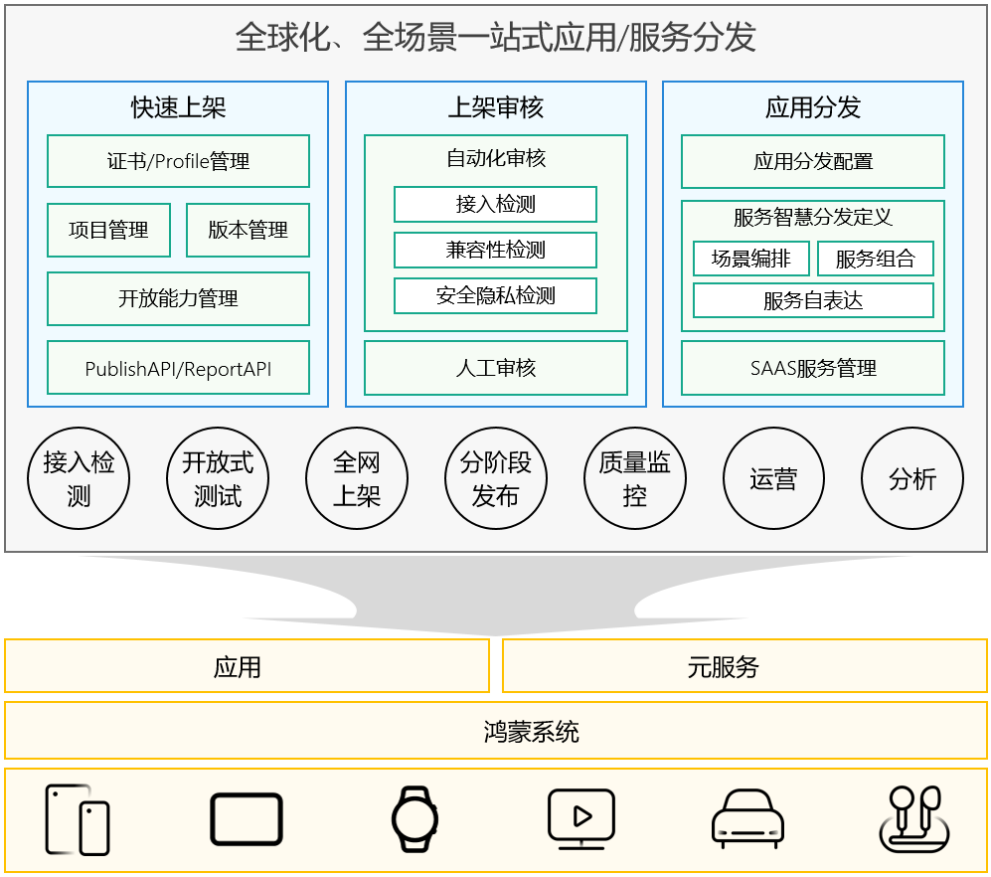


图 6-1：一站式分发能力

1）快速上架

HUAWEI AppGallery Connect 作为开发者统一提交入口，集成证书管理、项目管理、版本管理等功能，支持鸿蒙生态应用、元服务的快速上架与分发。

证书颁发

提供统一的证书颁发入口，支持软件证书和 Provision Profile 管理，保障开发者合法性与应用合法性。

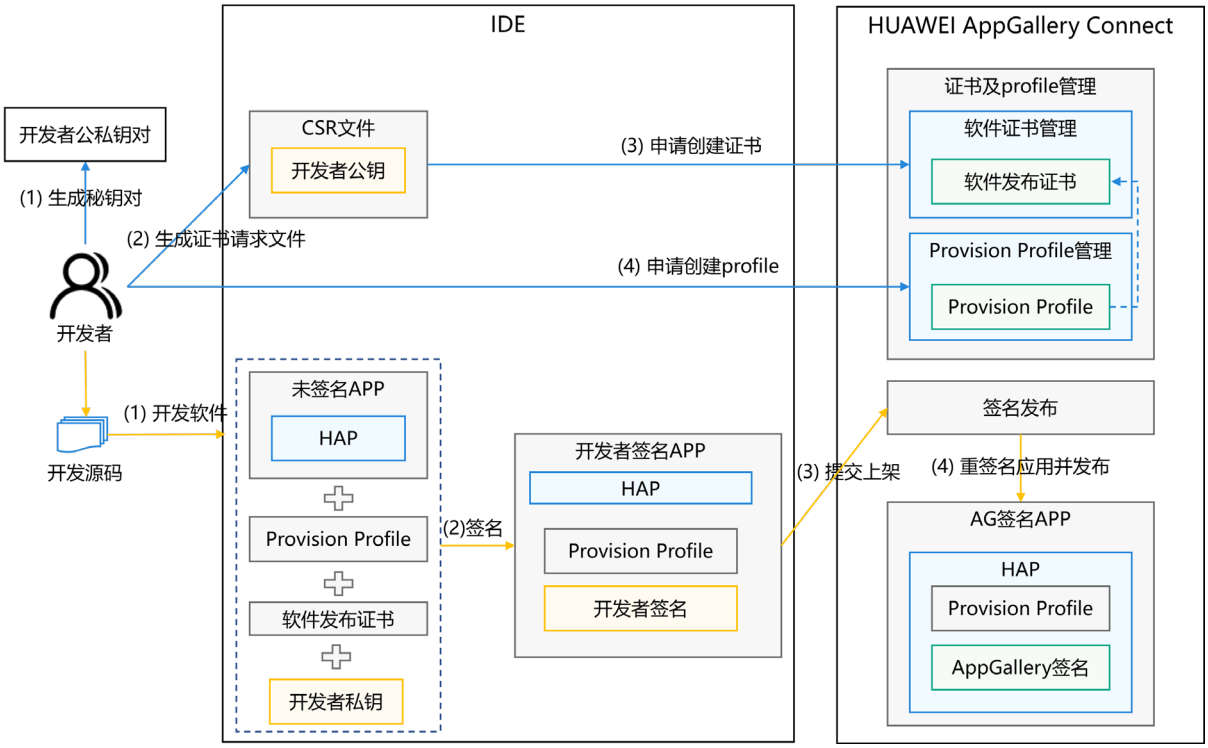


图 6-2：签名方案

- 开发者生成自己的公私钥对（Key Store），并且使用 IDE 生成软件证书请求文件（CSR）。
- 开发者上传软件证书请求文件，生成软件发布证书，选择对应的证书可以创建应用的 Provision Profile。
- 开发者使用 Provision Profile 和开发者私钥完成应用的本地签名，打包应用并提交。
- 只有经过 HUAWEI AppGallery Connect 合法性检测、拥有开发者所属软件证书签名的应用才允许提交。

- 经过合法验签之后，使用 HUAWEI AppGallery Connect 的秘钥对应用进行重签名。
- 允许通过以上步骤的应用安装到搭载鸿蒙系统的设备上。

上架

开发者开发完成之后，上传包体、描述信息、素材等，提交上架审核。也可委托 SaaS 厂商为其代理完成开发上架。

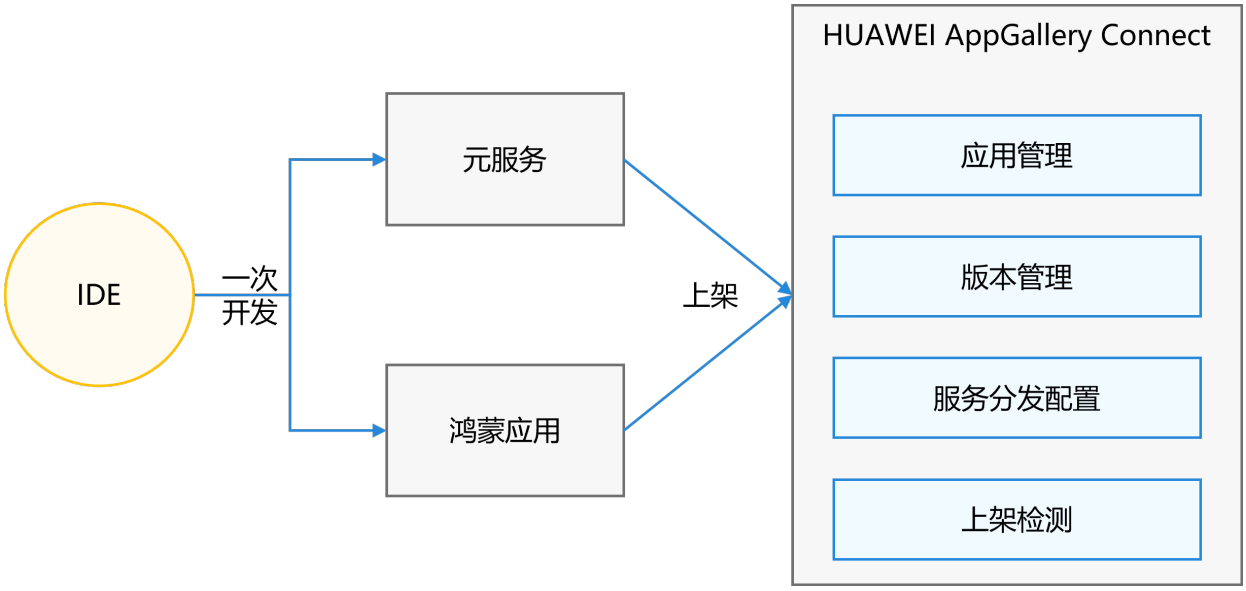


图 6-3：上架能力

- 应用管理：提供应用的基本信息管理，支持配置全球化名称，管理图标、截图、描述、应用分类、开发者服务信息等内容。
- 版本管理：维护应用的版本信息，支持配置发布区域、发布范围、应用资费、内容分级、隐私声明、版权信息等内容。
- 上架检测：支持对应用进行基础信息检测，包括 API、包名、签名、资源等，以及对 SDK 接入情况进行检查。

上架审核

为了给用户提供更安全且出色的体验，HUAWEI AppGallery Connect 对开发者提交的鸿蒙生态应用、元服务进行严格的审核与测试。开发者需了解并遵循《华为应用市场审核指南》。

- 应用审核指南：

https://developer.huawei.com/consumer/cn/doc/app/50104

- 元服务审核指南：

https://developer.huawei.com/consumer/cn/doc/app/50129

完成实名认证才能享受开发者联盟开放的各类能力和服务。

表 6-1：开发者检测

检测	简介
实名认证	只有实名认证过的开发者，才允许进行应用上架分发；应用市场支持个人开发者和企业开发者认证，认证方式多样化。个人开发者：银行卡认证、身份证认证、华为云授权认证；企业开发者：对公银行认证、企业资料认证、华为云授权认证。

开发者需提供资质文件以证明其内容符合法律、法规或政策的要求，同时为保障软件在设备上具备良好的使用体验，会对其兼容性、安全性、稳定性、隐私、性能、功耗等进行全方位检测。

表 6-2：应用/服务检测

检测	简介
行业资质检测	支持全行业资质自动化检测，包括游戏版号、计划及软件著作权证书、支付业务许可证等。
公司资质检测	对公司的合法性进行检测，包括合同、协议、授权书、安全评估报告等资质的自动化检测。
安全检测	进行安全风险问题检测，包括病毒木马、恶意行为、代码安全、逃逸对抗等。
隐私检测	通过动/静态检测，识别是否存在隐私风险，比如违规收集个人信息、超范围收集个人信息、违规使用个人信息、强制/频繁/过渡索取权限、强制用户使用定向推送等问题。
兼容性检测	通过真机检测，保障分发设备的兼容性，支持检测是否存在崩溃、无响应、运行错误、功能异常、界面异常等问题。
合规检测	通过 AI 技术，识别图片、描述、文本等信息，自动检测内容是否存在色情、暴恐违禁、赌博、毒品、政敏、低俗、禁播等违规行为。

2）应用分发

HUAWEI AppGallery Connect 提供了灵活的分发能力，支持按阶段、维度、场景等多种形式，高效、精准地分发到用户设备上。

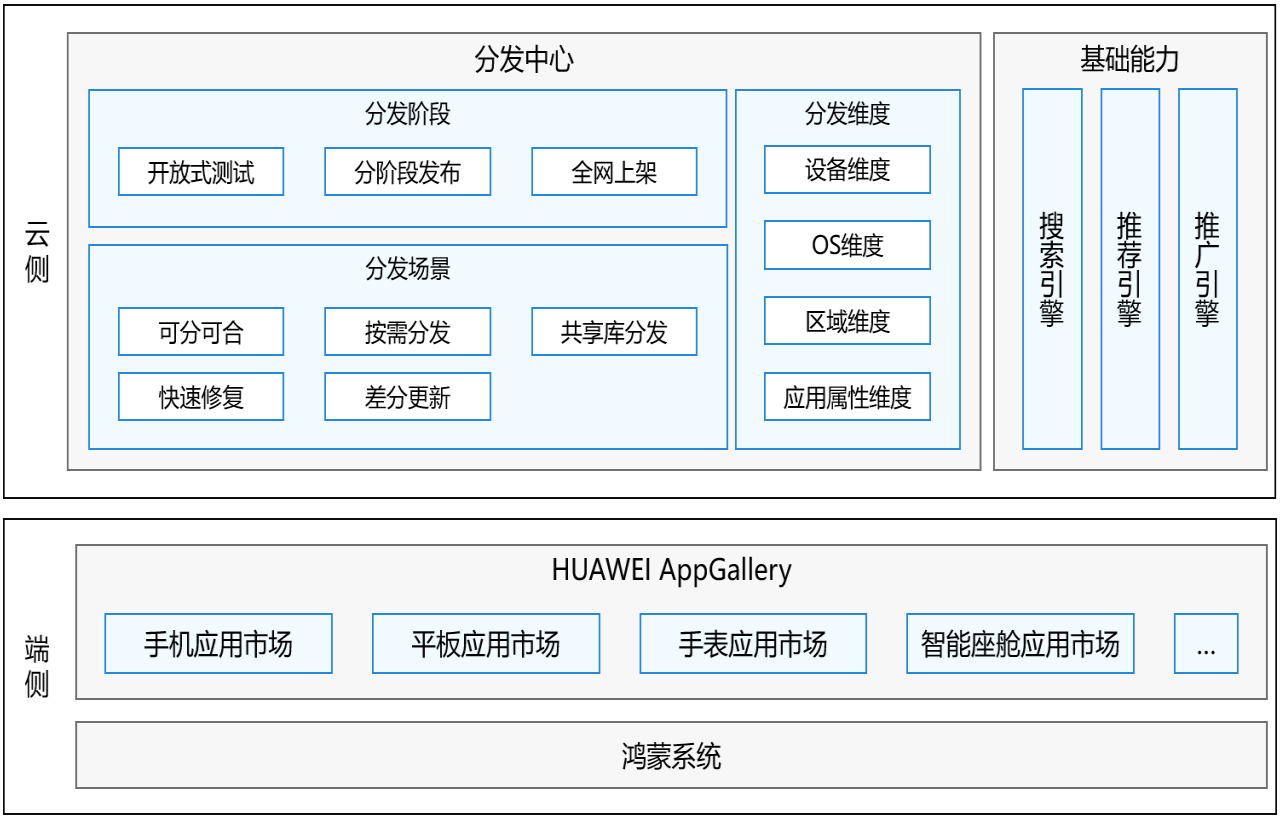


图 6-4：应用分发全景图

分发阶段

开发者可以在应用的不同成熟阶段采用不同的分发手段，结合应用的运行数据与用户声音，不断改进应用质量，持续提供优质服务。

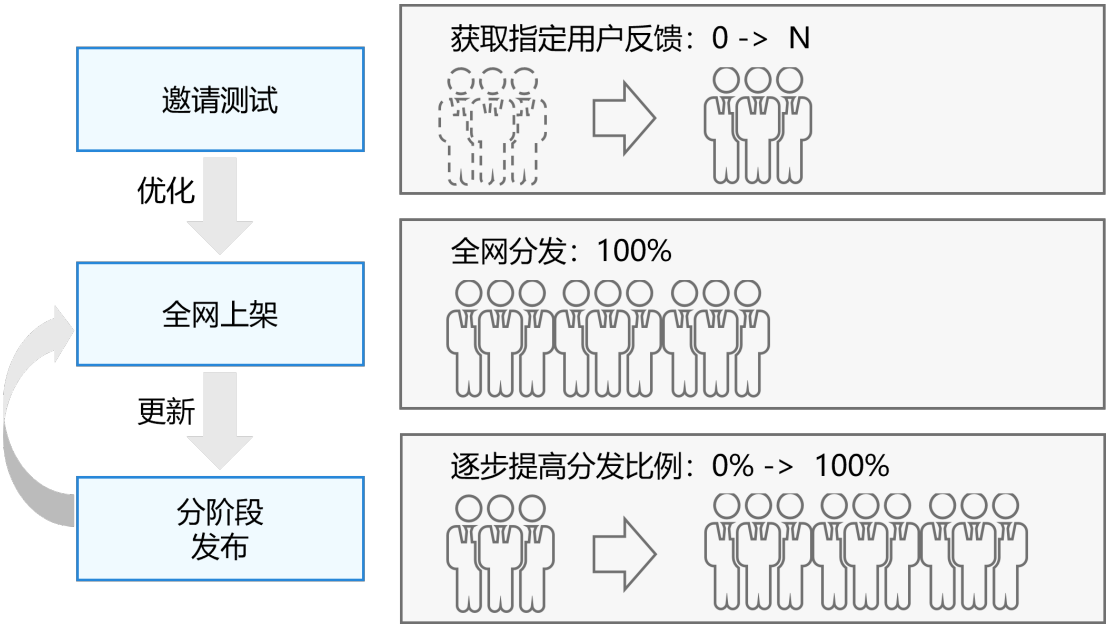


图 6-5：分发阶段

分发维度

开发者可以通过多种维度灵活分发应用：

- 设备维度：设备类型、PCID、屏幕类型等。
- OS 维度：API Level 等。
- 区域维度：发布国家区域、漫游分发等。
- 应用属性维度：年龄等级、应用分类、软件版本等。

分发场景

开发者上架应用之后，HUAWEI AppGallery 会根据不同的分发场景，自适应选择分发方式，从而实现高效分发，提升用户体验。

1. 按需分发

开发者可以将应用进行合理拆分，将非核心的功能做成动态特性；用户首次下载应用时，只下载基本功能模块，仅在执行到动态特性时才按需下载，既满足了业务功能，也减少了存储空间的占用，消耗更少的网络流量，提升下载转换率。

2. 智慧分发

HUAWEI AppGallery 构建了搜索、推荐、推广三大引擎，从海量数据中构建丰富的画像和知识图谱，基于 AI 能力实现精准送达。

表 6-3：引擎介绍

能力	简介
搜索引擎	基于精准的用户意图识别与丰富的鸿蒙生态应用、元服务标签体系，构建以用户体验为中心的多场景、多模态、全球化搜索引擎，高效连接鸿蒙生态和全球消费者。
推荐引擎	基于海量用户数据，使用机器学习和深度学习算法，提供千人千面的个性化推荐服务，精准触达目标用户。
推广引擎	面向合作伙伴提供精准、优质、高效的推广服务，支持面向安装、激活、次留、付费等目标的投放，助力合作伙伴快速精准获量，实现商业成功。

3）服务分发

元服务是 HarmonyOS 提供的一种轻量应用程序形态，具备秒开直达，纯净清爽等特征，实现多入口、场景化分发。多入口分发包括设备入口、系统入口、应用入口，场景化分

发是系统在理解用户的基础上，结合用户旅程的一种多服务组合分发，从用户场景出发，围绕用户旅程的场景化闭环。例如用户想去旅游，出行前要查看天气、预订机票酒店、购买门票，旅途中要去机场、打车、结束后要照片分享，发表感受等，在整个用户旅程中，都需要用户自己规划，自己寻找服务完成对应的操作。多服务的场景化分发是在理解用户意图后，将用户旅程中需要用到的服务组合起来，在用户需要的时候分发给用户，比如查看天气、预订机票酒店、购买门票、打车、航班提醒、分享感受等服务，让用户在旅程中感受到智慧便捷，实现场景的闭环。

入口丰富

1. 协同配合

鸿蒙系统的多入口不仅体现在数量多，层次多，并且体现在用户场景上的协同配合。如用户自驾去某景点，先使用手机导航选定路线，进入车后，导航从手机流转到车机上，并基于用户意图途中语音介绍景点，接近景点时，启动购票服务。

2. 流量矩阵

流量矩阵是入口流量的有机组合，能实现服务的触达、留存、召回全生命周期管理。

- 触达：系统根据用户偏好和所处的时间、地点等场景，识别用户意图，匹配用户所需内容主动推荐服务，使用户能发现所需服务。
- 留存：系统提供优质元服务和内容，吸引用户将服务留存在桌面和智慧生活·今天，并为用户提供统一的元服务查看、搜索、收藏和管理功能。
- 召回：通过系统能力再次推荐优先匹配服务，唤醒用户再次使用，召回用户。

智能分发

元服务数量的持续增加给用户带来查找服务不方便、选择困难等问题，若无法提供精准快捷的服务触发，用户将面临信息过载和获取服务时间成本过大的困扰。AI 的“感知”、“理解”、“推理&决策”等能力，能有效解决用户查找服务不方便及选择困难问题。智慧分发核心能力分为感知、知识与理解、推理三层：

- 感知层：精确感知用户场景，是服务智能分发的基础。根据多个终端的硬件传感信号和软件感知能力，感知层可以感知时间、空间、动作等信息进一步支撑对场景的理解。结合用户偏好，辅以知识图谱提供的结构化数据，系统实现了场景的精准融合感知。随着用户使用时长和次数的增加，场景的感知能力也将更加精准，推荐的服务将更加贴心，更好地满足用户的需求。场景感知分为如下几类：
  - ✧ 基于时间的场景感知：工作日、节假日、首次亮屏、午休、睡前时光等。
  - ✧ 基于地点的场景感知：家、公司、公共交通、商城、旅游景点等。
  - ✧ 基于设备的场景感知：音频播放状态、网络连接能力、运动状态等。
  - ✧ 基于人物的场景感知：性别，年龄，应用使用偏好等。
  - ✧ 基于事件的场景感知；快递物流，异常天气，订单状态、未接来电等。
- 特别说明，以上场景感知信息均在用户知情同意的前提下收集。知识与理解层：知识与理解层是智能分发决策的重要依据，围绕核心场景，持续构建、学习、丰富知识，并基于全面感知与知识增强，精准理解用户意图。感知数据结合用户的行为习惯，辅以知识图谱提供的结构化数据作为输入，通过对用户、场景建模以及 NLU 等技术，实现了用户此刻的意图理解。同时在元服务上架的过程中，开发者可以选择

服务分类和标签，通过服务分类、标签和智能理解，系统可以将元服务与统一全局意图进行关联。

- 推理&决策层：依托丰富的服务生态，完备的知识储备，学习型 AI 模型实现精准推理。通过基于规则的召回、热度召回、协同召回、深度学习模型召回等多路召回方式，为每个用户召回与其意图、兴趣相关的元服务，同时通过端云融合排序模型将召回的服务进行排序，并将 top 服务展示给用户。

开发者可以按照服务分发接口规范接入数据，使用户意图和服务数据更精准匹配，从而显著提升元服务的使用。

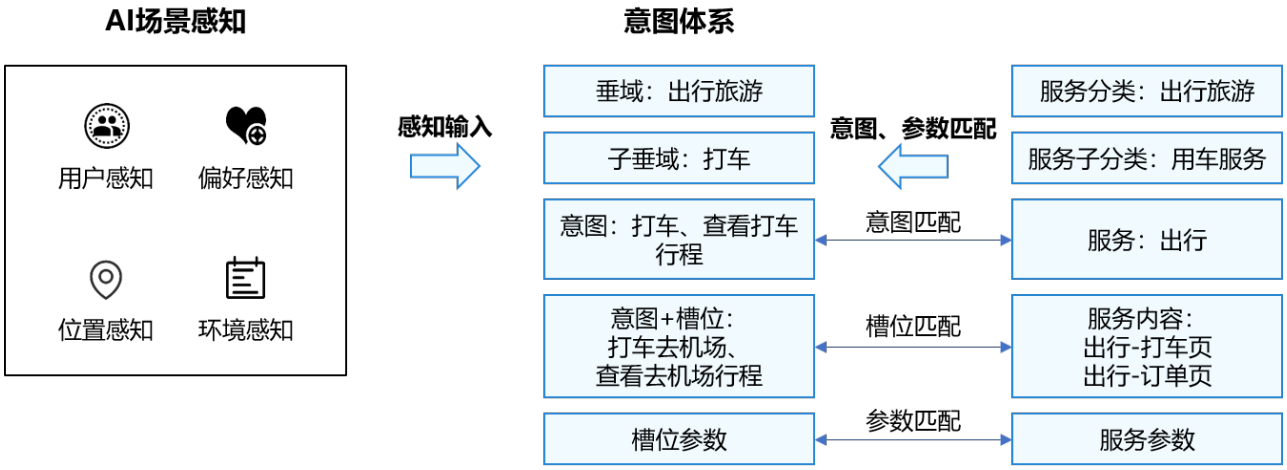


图 6-6：基于场景感知的 AI 分发元服务

数据接入接口按照垂域进行细分，例如附近优惠券，附近服务，热词排行，音乐内容等。开发者可以根据自己的服务具体使用场景，有选择的通过这些接口接入数据。

Chapter 7

自由流转与分布式运行环境

- 1）价值与架构定义
- 2）跨端迁移
- 3）多端协同

随着个人设备数量越来越多，跨多个设备间的交互将成为常态。基于传统 OS 开发跨设备交互的应用程序时，需要解决设备发现、设备认证、设备连接、数据同步等技术难题，不但开发成本高，还存在安全隐私、兼容性、性能等诸多问题。为了适应万物互联时代的环境变化，鸿蒙系统构建了基于分布式运行环境所需要的基础设施，为开发者提供了基础的分布式框架能力，使开发者可以更方便的实现跨设备的业务开发，向用户提供多设备的交互体验。

基于人机交互研究发现，多设备的交互场景按照时间的串、并行，分为以下两种类型：

串行交互：指用户相继使用多个设备。此类交互场景要求具备较高的连续性、一致性。

- 连续性：当用户从一个设备转向另外一个设备的时候，最新的操作状态应当是继续保留的、未被中断的。
- 一致性：当用户在使用手表、手机、大屏等不同设备时，交互方式与基础视觉元素应当是一致的，例如多指手势，控件样式等。这里的“一致”并不等于与“相同”，由于设备的屏幕尺寸、形态的不同，视觉元素可以进行适配调整。

并行交互：指用户同时使用多个设备。此类交互场景要求具备较高的协作性、互补性。

- 协作性：多个设备彼此交互协调，完成一项任务。
- 互补性：利用设备的本身形态差异，完成一项任务。例如，当用户在家里找不到电视遥控器的时候，手机可以变身为遥控器，这就是一种设备能力的互补。

对于并行交互与串行交互两种典型场景，鸿蒙系统分布式运行环境分别提供了与之对应的基础能力，即跨端迁移和多端协同，两者统称为“自由流转”。

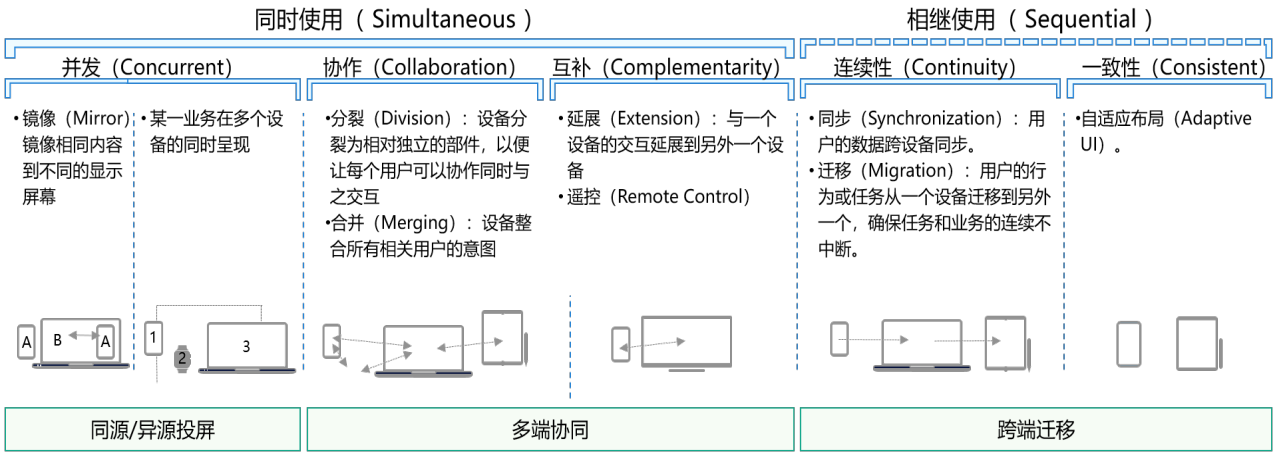


图 7-1：自由流转的两种形态

1）价值与架构定义

价值

自由流转的价值体现在以下几个方面：

- 自由流转提供了应用跨设备流转的能力。应用开发只需遵循框架并适配指定的 API，就能实现设备之间的跨端迁移和多端协同。

- 自由流转框架实现了跨端迁移过程，包括流转任务发布、元服务免安装、数据序列化、兼容性判断等。应用开发只需关注在业务数据本身的同步与恢复，简化了应用的处理逻辑，降低了应用开发跨端特性的复杂度。

- 自由流转框架实现了多端协同过程，包括周边协同目标的发布与发现、协同双方的可信认证、为协同双方建立稳定的高速通信通道，管理协同生命周期等。框架完成了复杂的目标搜索、安全认证与通信建立过程，开发者只需要关注多跨端业务的数据与控制交互，极大降低了应用跨端通信的复杂度。

架构定义

自由流转依赖于鸿蒙系统提供的分布式运行环境，该环境从下而上可以分为四层：

- 核心基础服务：为提供自由流转设备互联的核心基础服务，主要包括如下模块。

  ✧ 设备管理服务 ：提供设备与服务管理相关能力。设备管理服务在系统中的定位是提供设备发现能力、服务发布与发现能力的核心服务。

  ✧ 分布式软总线：主要提供基于近场通信技术的通信网络，实现分布式设备之间的有序通信，使得设备之间的传输变得安全可靠、通信 QoS（Quality of Service）可管理、业务质量可预期。

  ✧ 设备画像（Device Profile）：是设备硬件能力和系统软件特征的管理器。典型的设备 Profile 信息包括设备类型、设备名称、存储容量、是否折叠屏、有无屏幕、分辨率、设备安全等级、设备 OS 类型、OS 版本号等。设备画像可用于支持智能决策服务进行设备筛选、排序，也可用于支持设备信息的查询。

  ✧ 智能决策服务：提供智能化的设备筛选能力、设备排序能力。设备筛选能力主要的数据来源基于设备画像。

  ✧ HiChain：是设备互信认证部件，提供了设备间互信关系建立、管理、认证、解除的全生命周期管理能力，支撑设备间搭建安全的数据传输通道，是鸿蒙系统设备分布式可信互联的安全基础能力。

✧ 身份认证服务：提供端侧统一的用户身份管理、身份认证和访问控制判断能力。支持多用户操作系统，支持多种用户身份认证方式（包含 PIN、指纹、人脸等）。

- 分布式平台服务：负责拉通多个物理设备上的运行状态，同时提供跨设备间的资源访问和控制能力。
- 核心业务框架：对分布式平台服务层和核心基础服务层所提供能力进行调用，完成相应的业务功能，给应用提供更简单友好的使用接口。
- 开发接口层：所有的应用通过这一层来使用系统提供的能力。

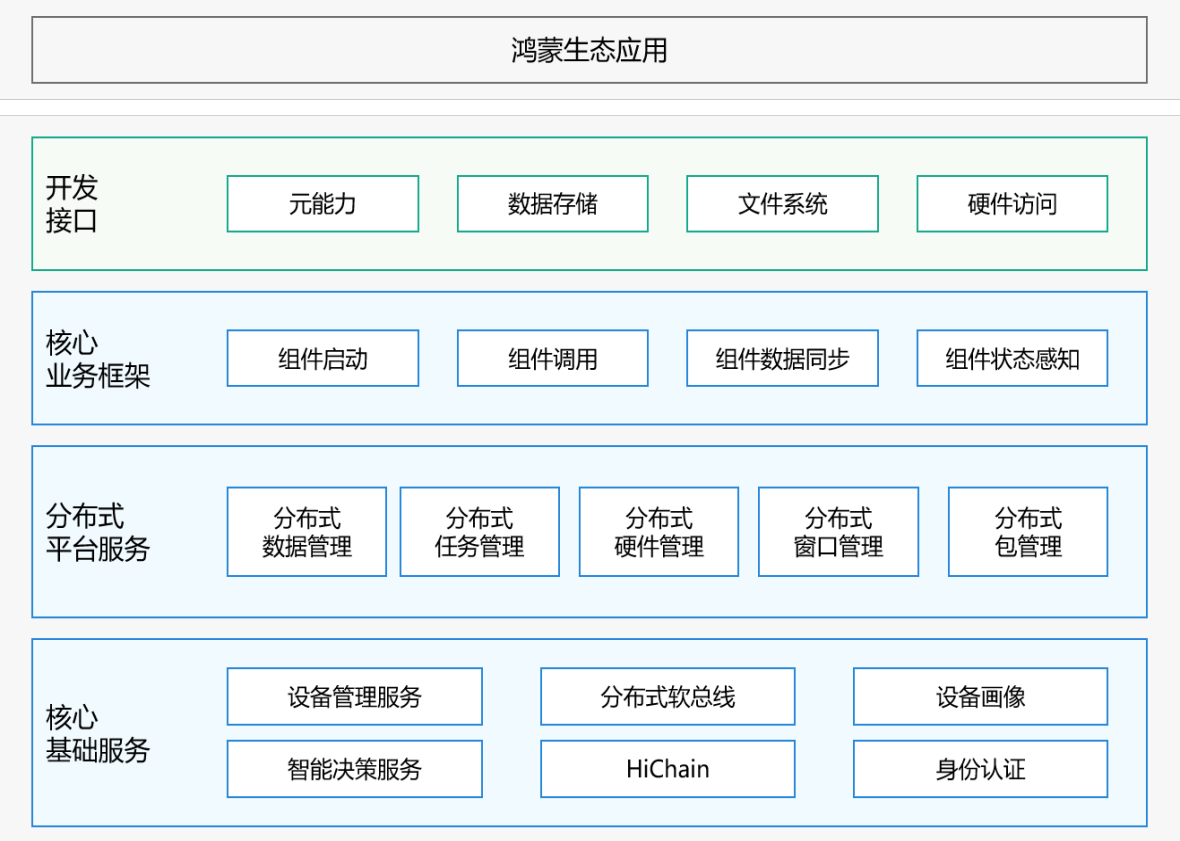


图 7-2：分布式运行环境

2）跨端迁移

跨端迁移是指将一个软件实体从一台设备转移到另外一台设备上运行。借助跨端迁移能力，鸿蒙生态应用可以自由地在多个设备之间流转，为用户带来无缝的用户体验，也会为开发者带来更多的入口和流量。

跨端迁移应用场景

用户使用应用的情境发生变化时（例如从室内走到户外、从办公室到车上等），之前使用的设备可能已经不适合继续当前的任务，或者周围有更合适的设备，此时，可以选择使用新的设备来继续当前的任务。

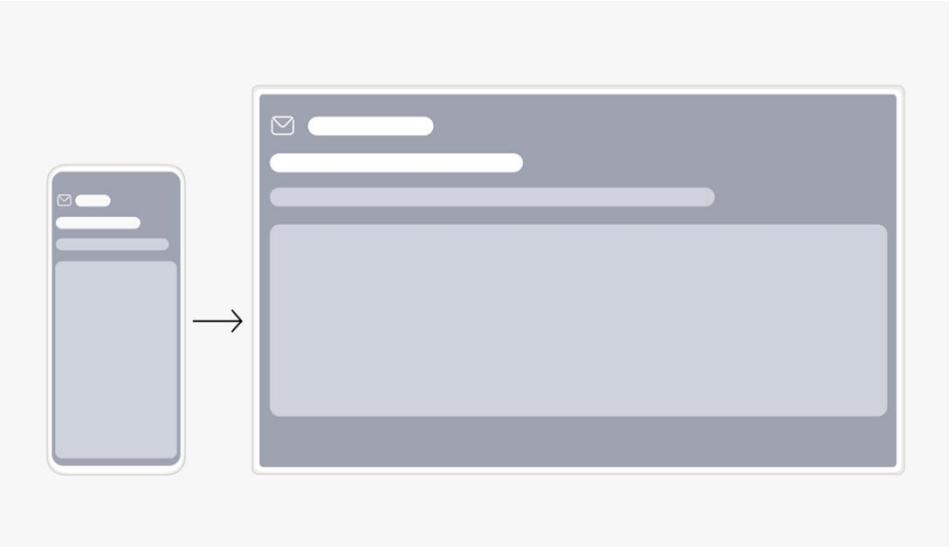


图 7-3：跨端迁移示意图

跨端迁移的应用场景举例：

- 在外时手机上编辑邮件，到家后迁移到平板上继续编辑。
- 在外时手机玩游戏，到家后迁移到平板上继续玩。

- 在家里智慧屏上看视频，出门时迁移到手机上继续观看。
- 手机视频通话迁移到智慧屏，更沉浸地视频聊天。

3）多端协同

多端协同是指运行在多个物理设备上的软件彼此协作完成一项任务。通过充分发挥每种设备的优势能力（例如智慧屏显示能力、手机输入输出能力、穿戴设备的便捷操作优势等），为用户提供更好的体验。

根据协同能力的不同，例如显示能力、交互能力等，可以创造出丰富的协同模式。下面以显示协同、交互协同两种模式为例展开介绍。开发者可以根据应用的实际特点基于鸿蒙系统提供的分布式开放能力，选择合适的模式进行体验设计，或者展开更多探索、创造出新的协同模式。

显示协同

场景举例：文档编辑应用的文档内容和周边工具菜单可以分别显示在智慧屏和手机上，在手机上快速操作编辑菜单，在智慧屏上更清晰地查看编辑的效果。

图 7-4：显示协同示意图

交互协同

场景举例：

- 在智慧屏上进行搜索时，在手机上进行文本输入。通过智慧屏上网课时，在手机上进行手写答题或绘画。
- 在 PC、手表或车机等设备上需要支付时，利用手机上的支付能力完成支付。
- 在地图导航时，手机后台导航，手表上展示简要信息和声震提醒。

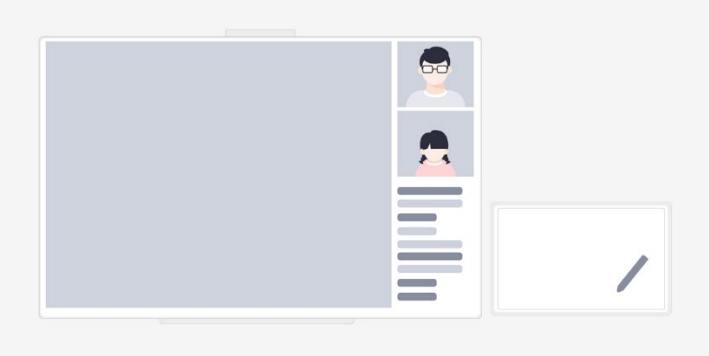


图 7-5：交互协同示意图

统一拖拽

拖拽操作不仅仅是一项功能，作为一种基础交互范式更是一种直观的人机交互方式，允许用户直接操纵屏幕上的对象和数据，这种方式模拟了现实世界中的交互，从而降低了用户的认知负担，并能使复杂的操作感觉更为简单。拖拽交互实现对于提升用户满意度和操作便捷性至关重要。

跨设备拖拽允许用户使用一套键盘和鼠标/触控板无缝控制多台同账号互联的设备，用户可以将光标跨越不同设备的屏幕，并在它们之间拖拽内容，模糊了多个设备的边界,进一步提升了用户无缝交互体验。

场景举例：

- 用户可以将 PC 上的文件拖到 PAD 的文件管理应用中，或者将 PAD 上的草图拖到 PPT 中。

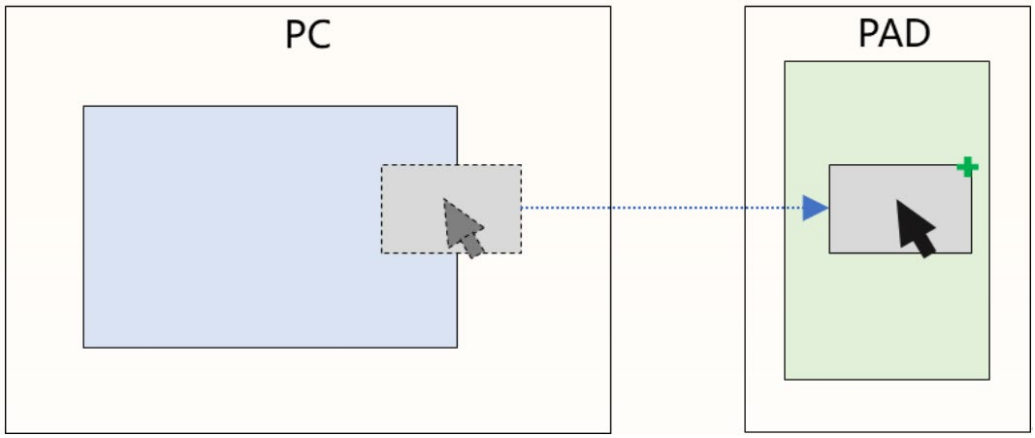


图 7-6：统一拖拽示意图

跨设备剪贴板

剪贴板为应用提供应用数据的复制粘贴能力，支持在应用内或应用间共享复制或剪切的应用数据。

剪贴板按场景分为本地剪贴板和跨设备剪贴板，本地剪贴板提供设备内的内容复制粘贴，跨设备剪贴板基于本地剪贴板提供跨设备的内容复制粘贴。

场景举例：

当用户拥有多台设备时，可以通过跨设备剪贴板的功能，在 A 设备的应用上复制一段文本，粘贴到 B 设备的应用中，高效地完成多设备间的内容共享。

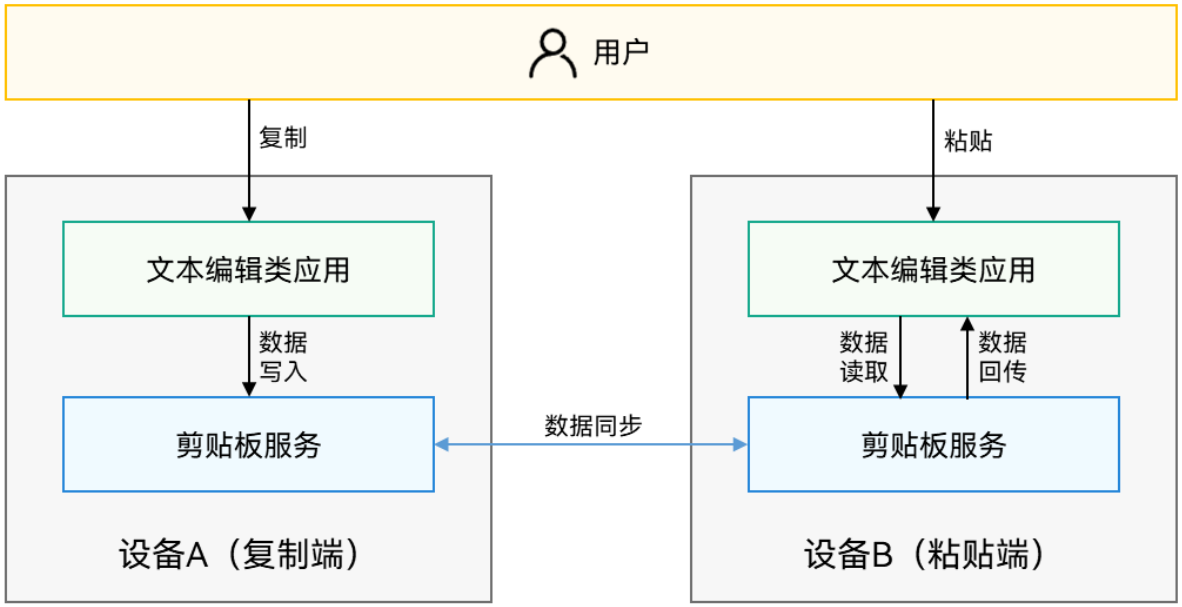


图 7-7：跨设备剪贴示意图

算力协同

场景举例：分布式游戏，在手机/大屏上玩游戏时，利用周边设备（手机、平板、笔记本等）协助完成游戏应用的计算任务（AI 计算、图像渲染），提升游戏帧率、画质。

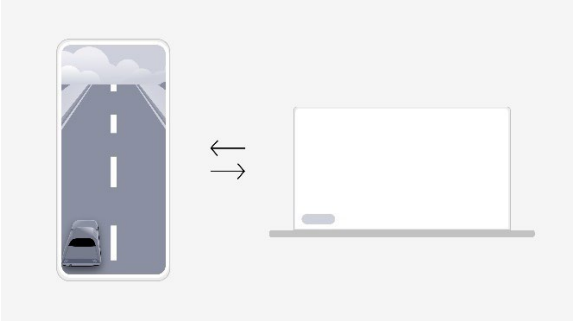


图 7-8 算力协同示意图

Chapter 8

全方位运维分析

HUAWEI AppGallery Connect 提供低门槛、高效率、多场景的大数据能力，包括质量分析、性能调优、故障定位、行业风向等。同时支持多维度数据分析，智能诊断问题并给出解决方案，为开发者明确质量优化方向，提升用户体验。

运维服务提供了低门槛、高效率、多场景的大数据能力，包括质量分析、性能调优、故障定位、行业风向等。同时支持多维度数据分析，智能诊断问题并给出解决方案，为开发者明确质量优化方向，提升用户体验。

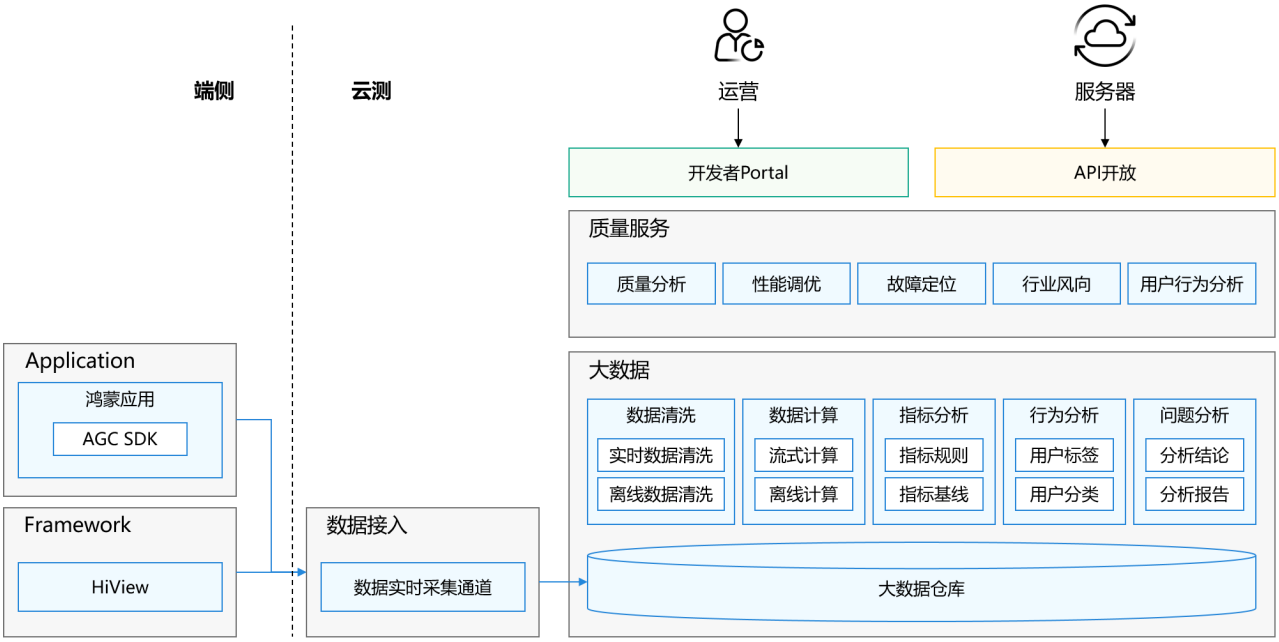


图 8-1：运维监控架构图

应用性能监测服务

帮助监测现网应用的进程崩溃（CppCrash、JS Crash）、应用无响应（AppFreeze）、资源泄漏（Resource Leak）等稳定性指标，以及应用启动、应用丢帧、页面加载、功耗器件等性能指标。

APMS 服务基于全面的异常类型、实时上报的异常数据提供准确的稳定性、性能指标度量，支持多维度的灵活查询分析。APMS 服务为每个稳定性问题提供问题发生时的环境信息、堆栈信息、系统日志等分析数据，并支持基于堆栈关键行进行准确的同类异常汇聚，轻松准确快速发现、识别、定位和解决问题。

表 8-1：应用性能监测服务功能

主要功能	功能描述
可视化实时报告	零代码集成，自动生成包含堆及其他相关信息的可视化数据报告，实时跟踪应用稳定性。
智能分类	大量崩溃会按照异常类型、代码位置自动分类，开发者可以根据对用户的影响程度对崩溃进行排序，确定优先级别。
实时监测和提醒	分钟级实时报告可以实时跟踪应用稳定性，当发生重大崩溃时，系统能及时提醒开发者。

故障定位

智能分析平台致力于帮助开发者高效定位和解决应用运维态问题，通过将专家定位知识转换为自动化定位能力，对日志进行分析得出根因。在智能分析平台上，可以监控到应用在运维态的多类问题，如应用崩溃、应用卡死、卡顿等，并对每类问题进行聚类统计，同时提供故障的详细分析报告，让开发者可以快速地了解应用的各类问题根因。

表 8-2 智能分析平台服务功能

主要功能	功能描述
一键式启用服务	0 代码集成，无需应用集成 SDK，通过平台订阅具体应用后即可使用平台提供的能力和服务。
故障自动感知	基于 OS 系统自动感知应用在运行过程中出现稳定性、性能、功耗等故障，并精准采集故障现场日志。
故障智能分析	将稳定性、性能等领域专家分析经验，转换为分析知识库，平台采集到故障日志后自动基于知识库进行分析，输出定位结论及修复建议，目前覆盖崩溃、卡死、内存泄露、冷启动慢、滑动丢帧等故障场景。
故障辅助分析	支持将本地的故障日志上传至智能分析平台，并自动出发故障智能分析，输出分析报告，同时将原始日志进行格式化呈现，帮助开发者快捷查看和检索日志。
故障根因统计	可按照版本、事件、根因分类提供故障数、严重度、TOP 根因等统计信息。

云监控

云监控提供开放型一站式监控解决方案，可用于监控 AppGallery Connect（简称 AGC）云函数、云数据库、APMS 等服务的统计指标，具备 AGC 服务日志检索和分析能力，并能够针对指定的监控指标设置告警，为优化统计指标（如云函数-平均时延、云函数-成功率等）提供依据。云监控使用简便，无须编写任何代码即可实现可视化数据报告的实时查看，并且可以实时洞察云函数、云数据库使用过程中的问题，持续改进服务质量，提升用户体验。

表 8-3 云监控功能

主要功能	功能描述
指标监控	提供了按照业务规则进行云函数、云数据库监控指标的统计和计算方法，可自动生成监控指标的可视化数据报告，可快速了解应用在哪些方面可优化改进。
日志服务	支持添加多个过滤器查询日志，并以直方图和表格形式展示服务运行过程中产生的日志数量和详细的日志分析数据。
告警管理	提供云函数、云数据库、APMS 服务监控数据的告警能力，可以通过设置告警规则来定义监控项的阈值，并在监控项满足告警条件时发送告警通知。

Chapter 9

全场景案例参考

影音娱乐

1. 场景描述

用户通过手机拍摄短视频/直播时，边拍摄边控制手机非常不方便，且无法做到多视角拍摄。在一些多人场景下如果能支持多机位、多视角拍摄，短视频或者直播的效果会更加出色。例如：乐队直播音乐表演时，智慧屏拍摄整个乐队，多台手机单独拍摄主唱和其他伴奏成员，最后通过一个平板或者手机整体控制切换不同的拍摄视角，完成整个直播的拍摄工作。这样拍摄出来的视频，既可以看到乐队的整体，又可以看到每个乐队成员的细节，体验更佳。

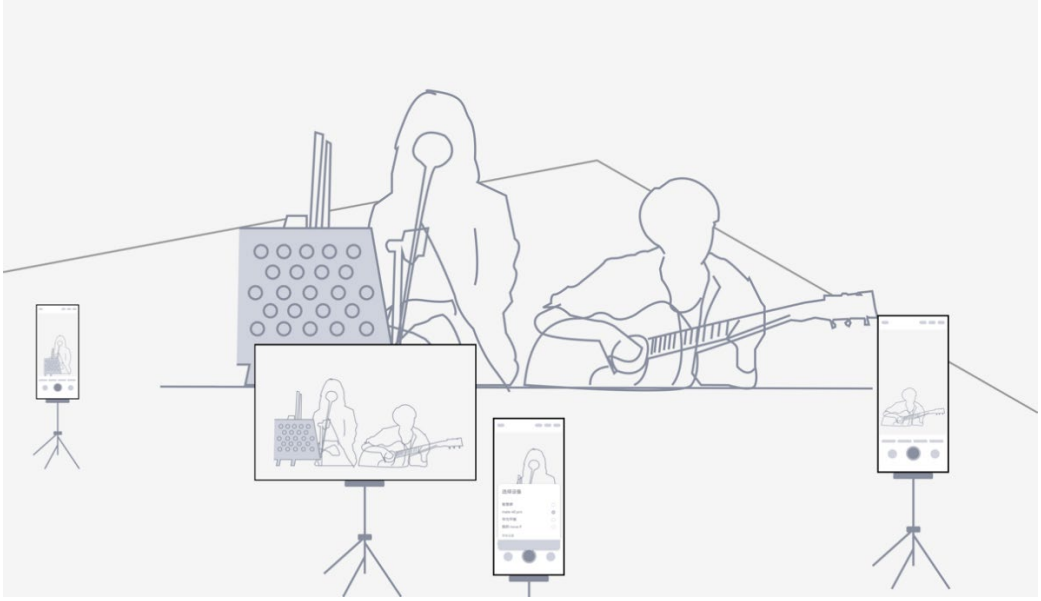


图 9-1：多机位直播场景示意图

2. 鸿蒙系统开放能力

“多端协同”和“分布式硬件（摄像头、麦克风）”等能力。

运动健康

1. 场景描述

用户通过智慧屏上的健身课程视频健身，单设备的智慧屏设备无法获取用户在健身过程中的健康数据（例如：心率、消耗热量等），进而无法根据运动数据给出更加科学的健身建议，可能会导致用户出现运动量过大或者不足等问题。基于鸿蒙系统提供的分布式能力，可以通过手表获取用户实时的心率数据，并把数据同步显示在智慧屏上，帮助用户了解健康状态。

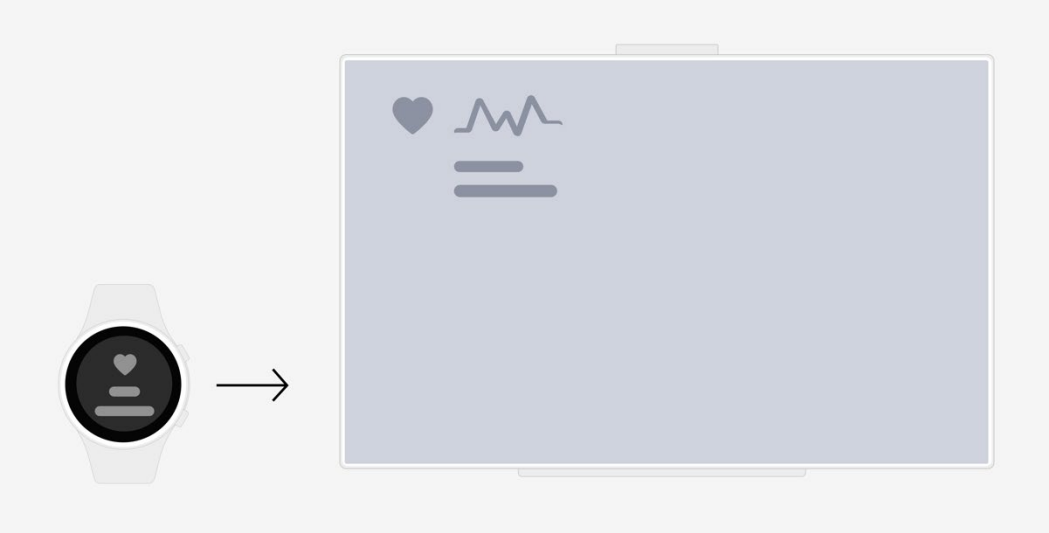


图 9-2：分布式健身场景示意图

2. 鸿蒙系统开放能力

“多端协同”和“分布式硬件（sensor)”等能力。

智慧出行

1. 场景描述

用户外出旅行或者出差，到达机场之后需要用手机打车到目的地，当用户手上拿了行李时，用手机查看出租车的实时状态非常不方便。通过鸿蒙系统提供的分布式能力可以很轻松的解决这个问题，只需要把出租车信息从手机流转到手表，用户抬腕就可以轻松查看出租车的最新状态。

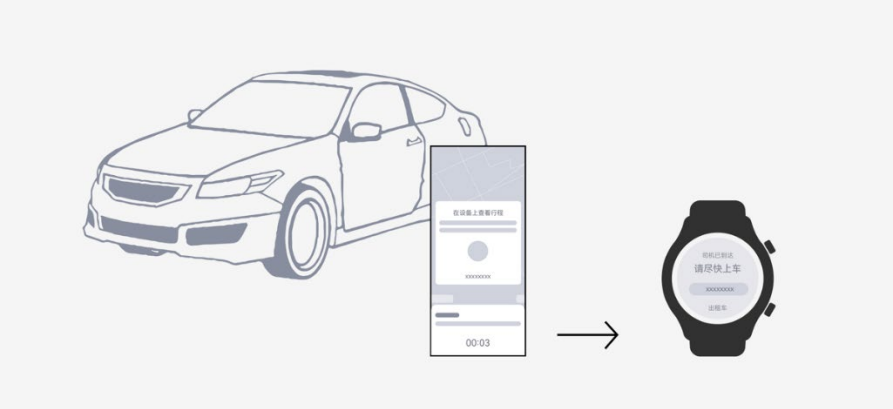


图 9-3：分布式打车场景示意图

2. 鸿蒙系统开放能力

“跨端迁移”等能力。

智慧办公

1. 场景描述

用户在上班路上通过手机编辑文档，到公司之后需要通过 PC 继续编辑时，传统的文档编辑应用需要借助云同步才能实现该需求。从用户体验看，借助“云”同步文件，如果文件

很大，同步速度慢并且需要用户多次操作才能完成文档的接续编辑。从开发者投入成本看，“云”同步文件，需要服务器资源和云文件同步功能的开发，开发者在传统的单设备操作系统上开发“一多”应用也会有很多端侧重复适配工作，大大增加开发成本。

通过鸿蒙系统提供的分布式能力，开发者只需要花费很小的成本便可实现用户的跨设备文档同步并接续编辑的需求。用户只需要在手机编辑文档的页面点击流转按钮，选择需要流转的 PC，手机的文档就会同步到 PC，同步完之后自动打开文档进入编辑页面，用户就可以接着在 PC 上继续编辑这个文档了。

图 9-4：分布式办公场景示意图

2. 鸿蒙系统开放能力

“一次开发，多端部署”、“跨端迁移”和“分布式文件”等能力。

智能家居

1. 场景描述

亲子游戏用于活跃家庭气氛，增进亲子关系。但是一般的多人游戏需要多个游戏手柄才能进入游戏，居家生活中，经常会出现找不到手柄的状况。通过鸿蒙提供的分布式能力，只需要把手机作为游戏手柄，随时可接入游戏。

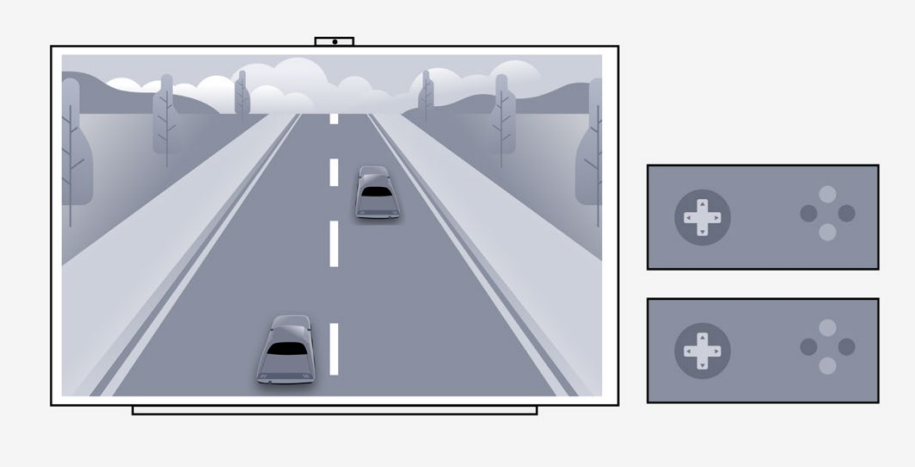


图 9-5：分布式游戏场景示意图

2. 鸿蒙系统开放能力

“多端协同”和“分布式硬件（sensor）”等能力。

Chapter 10

附录：术语

A

- Ability：应用的基本组成部分，是应用所具备能力的抽象。Ability 是系统调度应用的最小单元，是能够完成一个独立功能的组件，一个应用可以包含一个或多个 Ability。
- Application sandbox（应用沙盒）：通过自主访问控制（Discretionary Access Control, DAC）、强制访问控制（Mandatory Access Control, MAC）等访问控制机制，隔离系统资源，用于保护应用自身和系统免受恶意应用的攻击。
- App Pack：HarmonyOS 应用程序的打包形态，后缀为.app，每个 App Pack 可以包含 1 个或多个 HAP 包。
- ArkCompiler：方舟编译器，是鸿蒙系统内置的组件化、可配置的多语言编译和运行平台。
- ArkCompiler Bytecode：即方舟字节码，ArkCompiler 编译工具链负责将 ArkTS/TS/JS 源码编译成字节码，作为运行时的输入，实现对应的语言的语义逻辑。该类字节码文件的后缀缩写为.abc。
- ArkUI：鸿蒙系统的 UI 开发框架，支撑开发者高效地构建跨设备应用 UI 界面。
- ArkTS：鸿蒙生态的应用开发语言。提供了声明式 UI、状态管理等相应的能力，让开发者可以以更简洁、更自然的方式开发高性能应用。
- Atomic Service（元服务）：鸿蒙系统提供的一种全新的应用形态，具有独立入口，用户可通过点击、碰一碰、扫一扫等方式直接触发，无需显式安装，由程序框架后台静默安装后即可使用，可为用户提供便捷服务。

C

- C API：鸿蒙 SDK 提供的 Native 开发接口。

H

- HAP：HarmonyOS Ability Package，一个 HAP 文件包含应用的所有内容，包括代码、资源、三方库及应用配置文件，其文件后缀名为.hap。
- HarmonyOS 平台开放能力：通过 HarmonyOS SDK API 提供，跟随 HarmonyOS 版本节奏发布，开发者通过调用 API 的方式进行使用；

M

- Module：在开发态，专门指 IDE 中工程管理的一种由开发者决定的功能相对聚合的功能单元。一个 IDE 工程可以包含多个 Module。Module 可以被编译打包成一个 HAP，用于在设备上安装运行。
- MSDP（Mobile Sensing Development Platform）：MSDP 子系统提供两类核心能力：分布式融合感知和分布式设备虚拟化两大部分。
  - ✧ 分布式融合感知：借助鸿蒙分布式能力，将各设备感知源进行汇总融合，对用户的空间状态、移动状态、手势、健康状态等进行精准感知，构建全场景泛在基础感知能力，支撑智慧生活新体验。
  - ✧ 分布式器件虚拟化：借助鸿蒙分布式能力，构筑器件虚拟化平台，将外部设备的各类器件（如 Camera、显示器、SPK/MIC 等）虚拟化为本地设备的器件延伸使用。同时具备将自身器件共享给其他设备使用的能力。

O

- OHPM：鸿蒙生态三方库中心仓。

S

- Super virtual device（超级终端）：通过分布式技术将多个终端的能力进行整合，存放在一个虚拟的硬件资源池里，系统可以根据业务需要统一管理和调度终端能力，来对外提供服务。
- SystemCapability：简称 SysCap，即系统能力，指操作系统中每一个相对独立的特性，如蓝牙、WLAN、NFC、摄像头等，都是系统能力之一。每个系统能力对应多个 API，这些 API 绑定在一起，随着目标设备是否支持该系统能力共同存在或消失，也会随着 IDE 一起提供给开发者做联想。

T

- TSAOT：方舟编译运行时利用 TS 携带的类型信息，直接将 TS 语言编译成端侧机器码，使得 TS 运行阶段能获得更高的性能。

X

- XComponent：ArkUI 提供的组件接口，满足开发者自渲染的需求。

HUAWEI