移动应用的软件工程，关于挑战与解决方案的调查

Shehab Eldeen Ayman Mounir
Shehab.ayman@bue.edu.eg 埃及
英国大学

摘要

移动应用开发已成为软件工程的前沿领域。近年来，智能手机平台不断增加，包括但不限于 webOS、BlackBerry OS、Tizen、安卓和 iOS。这些平台并存，导致了一个充满挑战的局面：应用程序必须达到同等水平，并得到持续维护。近年来，移动应用开发领域明显出现了越来越多的应用程序，这些应用程序采用 HTML5 等 Web 元素来生成类似原生的应用程序；从本质上说，它们是封装在容器中的 Web 视图，从而呈现为普通应用程序的外观。这意味着应用程序行为可能因用户而发生巨大变化，也就是说，应用的行为可能被大幅改变。因此，应用开发人员依赖敏捷或临时的方法进行开发，而这种开发大多是自主进行的。本文介绍了移动应用开发中上下文感知的当前技术发展水平。

引言

毋庸置疑，过去十年智能手机普及度的惊人增长重塑了软件工程行业。

随着应用下载量达到数十亿次，移动应用开发已成为软件工程的前沿领域。近年来，智能手机平台不断发展，包括但不限于 webOS、BlackBerry OS、Tizen、安卓和 iOS。iOS 和安卓占据了智能手机平台的大部分市场份额。一些开发多功能软件时常见的挑战已从桌面端转移到移动设备，但为智能手机开发软件也带来了一系列需要克服的新挑战。

2017 年的最新估算显示，Apple App Store 中的应用数量超过 220 万个，而 Google Play Store 中则有 280 万个，并且两者都在持续增长。尽管 Windows App Store 和 BlackBerry World 均已停止运营，但它们此前分别拥有 60 万个和 20 万个应用 [1]。

针对这些移动平台进行开发，不像桌面端开发那样灵活。开发过程所需的编程语言、工具、框架和 API 都因平台而异。例如，原生 Android应用程序使用 Java编程语言在 Android Studio 集成开发环境中构建

而苹果的 iOS 应用程序则使用 Swift 编程语言和 Xcode 工具开发 [2]。这些平台并存，导致应用程序开发和维护面临挑战，必须确保它们达到相同的水平。

在移动开发领域，传统的软件工程方法已在一定程度上被弃用，因为这些方法无法直接应用于移动环境。移动图形用户界面被视为人机交互研究中的全新范式。移动电话提供了一种全新的交互方式；与需要坐下来、集中注意力和资源进行交互的计算机使用情境相比，这种方式被认为并不传统。交互方式也已从典型的鼠标和键盘大幅转变为触摸屏和手势。而交互方式似乎还在不断演变，因为语音命令以及增强现实／混合现实正逐渐成为可能 [3]。如今，硬件和软件平台的多样性迫使开发人员为不同平台制作一组不同的应用程序：这些应用程序表面上看起来可能相同，内部却截然不同，以适配每个平台 [4]。

移动开发面临的挑战

1. 创建通用图形用户界面

Tarasewich 和 Gong 开展的研究表明，Shneiderman 的设计原则中至少有四项——后文将进一步探讨——无需修改即可应用于移动电话。这些原则包括“让频繁使用的用户能够使用快捷方式”“设计对话以形成完整闭环”“提供信息丰富的反馈”以及“支持内部控制点”[5]。随着技术不断发展，挑战也随之演变，因此近期的研究工作致力于探索如何简化应用开发，而不受硬件或软件平台的限制。简化这一流程能够释放大量资源，进而用于打造更好的用户界面。

2. 在不同平台间复用软件

企业面临艰难抉择：是专注于一个或两个平台，提升用户体验；还是全力支持当前市场上的每一个平台，并以牺牲软件质量为代价 [6]。

有时，用户界面无法在 iOS 和安卓之间统一，这并非因为两个平台存在技术限制，而是因为各个平台的用户

已经习惯了某些设计特点。移动开发者很容易将带有右下角插入圆形图标、左上角三个竖点的应用程序识别为安卓应用程序；点击这三个竖点会打开超级按钮菜单。而右上角通常设有返回/上一页按钮的应用程序，则往往是 iOS 应用程序。安卓应用程序通常不设返回按钮，因为安卓用户习惯使用操作系统本身提供的返回按钮。iOS 用户则习惯在应用程序中寻找返回按钮；如果没有，向右滑动即可实现与返回按钮完全相同的功能。

许多软件公司会设立不同的开发团队，每个团队一次只专注于一个单一平台。这意味着，如果要为 iOS 和安卓开发一款应用程序——例如——那么，为了在这两个操作系统上提供相同的功能，所需的软件工程工作量就会翻倍。软件复用和重构的基本软件工程理念在此并不适用，因为这些内容无法从一种软件转移到另一种软件。这导致开发团队之间的协调非常有限，主要依赖临时安排，几乎没有真正投入精力来减少项目所分配的资源，尤其是时间和成本。

近年来，移动开发领域中，采用 HTML5 等 Web 元素来构建类似原生应用程序的应用程序，其数量显著增加。这类应用程序本质上是封装在容器中的 Web 视图，呈现效果与普通应用程序无异。这类应用程序（后文将进一步讨论）无法像原生应用程序那样具备丰富的功能，因为它们无法访问平台的 API，因而功能受到限制；但另一方面，它们允许在多个平台上复用几乎完整的应用程序界面 [7]。

3. 上下文感知应用

移动设备与传统的固定式计算机平台截然不同。移动电话具有高度的可定制性，能够适应用户的需求，这意味着它们必须持续监测环境。因此，移动设备上的应用程序需要时刻感知时间、天气、位置、方向、距离等信息。[8] 如今，移动应用能够利用所有这些情境感知信息，为每位用户打造高度具体、专属的体验，这意味着应用程序行为可能因用户而异，差异极大。上下文感知并非新功能；多年来，Web应用一直提供类似的用户定制体验，但由于移动电话的

传感器和功能，这种体验从未达到移动应用的程度。传统软件工程中从未出现或处理过这种传感器过载；它一直与机器人学和智能对象相关，而这些通常通过面向智能体的软件工程来处理[9]。这些传感器的普及使其应用成为应用开发的重点。因此，开发人员会仔细分析应用需求，并利用上下文感知大幅提升应用程序的质量，从而带来更好的用户体验。

面向智能体的软件工程（AOSE）致力于构建高度自主的软件智能体。这种方法提供了构建多智能体系统（MAS）上下文感知应用程序所需的全部模型、抽象以及“纯粹”的软件工程方法 [10]。为便于理解，我们可以设想多智能体系统是一台实体机器人。机器人必须利用其传感器感知并理解周围环境，并据此做出相应反应，以实现其目标。

实现。如今，定期收到应用程序更新已是再正常不过的事，这些更新会添加新功能，或着重于稳定性改进和错误修复。这意味着，传统的瀑布模型已无法切实应用于移动开发，因此，应用程序开发人员会采用敏捷方法或临时方法进行开发。对更多上下文感知应用、定制化用户体验的需求持续增长，竞争异常激烈，而用户对应用程序无响应的容忍度又很低，这些因素促使开发过程更多地采用半形式化方法，不一定拘泥于既定规范。这种方法必须以某种方式融入敏捷方法。移动应用所提供的体验极具动态性且因用户而异，因此会出现功能需求和非功能需求未必能完全涵盖的场景与情况。如果移动应用严格遵循其功能需求和非功能需求，用户体验质量就会下降。这意味着，应用程序需要持续运行、自主调整并修改其行为，还要提供超出严格规定范围的更多功能。

4. 平衡敏捷性与不确定性

不可否认，移动应用的存在改变了人们看待开发过程的方式。过去，几乎所有应用需求都能得到明确规定，并在首次运行时完全

很难想象，应用程序如何能够提供比设计文档中最初规定的更完善的服务，以及这种持续自适应的服务如何提升应用程序的质量和用户参与度。A

自适应服务的一个绝佳例子是用户定制广告体验。这已成为数字广告的重要方式，广告位中展示的产品会根据用户兴趣而定。如今，收集用户兴趣相对容易。例如，谷歌广告服务会追踪每位用户的搜索历史，以判断这位用户的兴趣所在，然后展示用户可能想要购买的商品广告。另一个绝佳例子是 Facebook 应用程序，它利用定位服务和个人资料分析来推荐好友。这项情境感知功能会频繁更新。例如，如果某人离开了工作场所、大学，甚至离开了所在国家，应用程序会根据更新后的位置，持续提供相关的新推荐。Facebook 涉足并产生重大影响的另一个领域是 Facebook Marketplace 服务，人们几乎可以在这里买卖任何东西，从二手手机到二手汽车。该应用程序利用上下文感知，将用户与最近的买家联系起来，并向感兴趣的用户推广商品，因为这样最有可能促成成功的买卖。

跨平台开发

移动开发中一个持续存在的挑战，尤其是对于初创公司和规模相对较小的公司而言，就是选择要针对哪些平台进行开发。公司往往会将开发重点放在 iOS 和安卓上，以尽可能覆盖更多智能手机用户。让尽可能多的用户都能使用应用程序可以带来更多利润，也能提升其在市场上的影响力。因此，公司需要花费时间和资金，聘请技术娴熟的开发人员，确保他们能在每个平台上开发出达到同等标准的应用程序。

跨平台应用程序开发的挑战

1. 创建通用图形用户界面

无论实际设计本身如何，每个移动平台都有自己的方式，供开发人员满足用户界面要求，并根据开发者的需求进行调整。一个好的起点是应对不同智能手机和平板电脑的各种屏幕尺寸和分辨率。例如，安卓在这方面非常灵活，为开发人员处理不同尺寸和分辨率的屏幕提供了更大的灵活性。苹果在这方面则似乎非常严格。iOS 应用程序的尺寸和分辨率受到其所面向的特定 iPhone/iPad 机型的限制。因此，开发人员要在不同平台上实现统一用户界面设计颇具挑战。

发表于《International Journal of Advanced Computer Science and Applications》的一项调查采访了多位拥有不同年限经验的智能手机应用程序开发者，以了解他们如何进行用户界面设计。调查发现，iOS 开发人员倾向于遵循苹果自己的用户界面指南。这些指南对尺寸和分辨率作出了限制，同时也让开发人员在设计方法上保有一定灵活性，例如选择使用苹果自家的 XIB、故事板或模型-视图-控制器。另一方面，安卓开发人员似乎更倾向于遵循谷歌自己的 Material Design 指南，因为它能带来更好的用户体验 [11]。

但它将无法使用谷歌提供的免费服务，例如 Google Play Store 和 Google Play 服务。这一问题给各家公司带来了诸多麻烦，因为如今决策者必须选择：是整合 Google Play 服务（它们无疑是开发符合最高标准的应用程序所必不可少的），还是放弃这些服务，以便开发能够脱离谷歌自有应用商店独立发布的应用程序。这是一个艰难的决定，因为决策者必须权衡：是整合这些服务，吸引北美和欧洲用户，却失去东南亚多达十亿智能手机用户的潜在市场；还是开发一款能在亚洲市场上架的应用程序，但要承担其质量不及竞争对手应用程序的风险。

2. 单一平台中的问题

即使在单一平台上，也可能出现内部挑战，使开发变得更加困难。安卓就是一个典型例子。市场上并存着各种各样的安卓操作系统版本。三星、Xaiomi、OPPO 和华为都在安卓操作系统之上使用各自的界面皮肤，这有时会导致应用程序的某些部分出现兼容性问题。2020 年初，美利坚合众国与中国之间的贸易战带来了新的困境，并影响了软件开发。受近期事态发展的影响，中国电信公司华为无法再使用美国制造的软件。尽管华为仍会使用安卓操作系统，

有一些跨平台框架旨在简化开发过程，让同一应用的不同版本能够在完全不同的平台上运行。任何跨平台框架最初的目标都是“编写一次，多处编译”，但框架能否成功并不总是有保证的。最著名的跨平台框架之一是 Xamarin。它由微软开发，同时支持 iOS 和安卓。用于为这两个平台编写应用的编程语言是 C#。开发人员编写应用时，就像是在为 Windows Phone 设备开发一样。借助兼容性库，Xamarin 随后会获取

Windows 软件开发工具包的 API 调用，并将其转换为 iOS 和安卓平台上对应的 API 调用。另一个已成为热门话题的新框架是 React Native，它由 Facebook 开发，旨在提供一个同时支持 iOS 和安卓的跨平台开发环境。

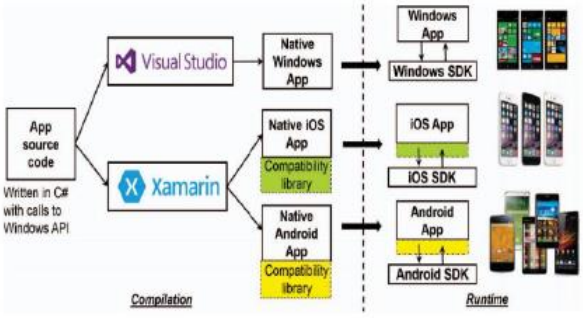


图 *1* 展示了源代码如何转换为 *iOS* 和安卓代码 *[12]*

开发人员在尝试使用这些跨平台框架时经常会面临许多挑战。这些框架通常缺乏对其自身库的适当支持，因此当开发人员遇到问题时，很难获得社区的适当支持。此外，其中一些库在访问移动设备特定功能和实现适当的硬件利用率方面往往能力有限，这会导致内存泄漏问题和较低的 CPU 利用率。

很难找到优秀的跨平台开发人员，尤其是那些具备特定框架开发经验，并曾在以往项目中负责编写、

集成和测试的开发人员。涉及重大风险或受到限制的项目——无论是时间还是资金方面的限制——往往倾向于采用更稳定、更安全的原生开发方式，并避免使用跨平台方案。

调查中，一些开发人员表示，坚持采用原生应用有助于简化开发过程，因为社区支持更完善；而要让跨平台应用模仿原生应用的外观和使用体验，则需要投入大量时间和精力。另一些开发人员表示，他们愿意学习开发应用程序的新方法，但所处的环境并不利于此。承担大型项目的公司在交付项目时往往受到时间和资金的限制，因此，如果在一个相对较新的环境中开发项目（对开发人员而言是新的环境），项目就会面临风险。一位开发人员表示，只要给予优秀的开发人员充足的时间和资源，学习任何一种新的跨平台框架都不难。另一位开发人员则表示，学习用于跨平台开发的框架一点也不难，因为它们都共享相同的面向对象编程基本概念，而这些概念构成了编程的基础。

3. 需求变更

无论是桌面应用程序还是移动应用程序，如果客户在开发初期更改或修改了某些需求，都可以进行相应调整。如果在给予充足开发时间并进行适当测试的情况下，需求仍在不断变化，那也没有问题。但如果需求在开发生命周期的后期发生变化，修复这些问题的相关成本就会相当高。开发过程中频繁变更，无论是因为一开始需求含糊不清，还是客户需求突然改变，都会造成高昂的成本，因为必须考虑这些变更，有时甚至需要从头重新启动整个开发过程。变更不明确会浪费时间和金钱，因此敏捷流程会预留应对此类情况的时间段 [13]。

4. 开发过程中测试的重要性

测试是开发过程中的重要环节；其目的是找出并修正应用程序中的错误，从而提升应用程序的效果。总体而言，测试旨在提高应用程序的质量。但小型组织往往无力为测试分配资源。有时，小型组织也无力聘请测试人员，因此开发人员还要兼任

测试人员 [14]。而且，开发人员通常很难发现自己编写的代码中的缺陷。有些组织可能完全没有资源进行测试，另一些组织则只在模拟器上测试应用程序。模拟器本身的功能非常有限，无法充分模拟移动性、动态分辨率和宽高比，也不具备面向移动设备的指南针、陀螺仪和全球定位系统等功能。

调查发现，各公司的测试方式各不相同。有些公司倾向于自动化测试，而另一些公司则倾向于手动测试。还有一些公司会进行负载测试、Alpha 测试、Beta 测试和冒烟测试。

一位开发人员表示，他们的公司采用 Scrum 方法，尽管他们也明白，这并不适合规模相对较小的应用。另一位开发人员则表示，他们的公司设有专门的质量保证部门，由该部门在实际设备上对应用进行手动测试。总的来说，几乎所有开发人员都认同，测试对于交付无缺陷的应用绝对必不可少。

5. 调整旧版 API 以支持新功能

开发人员在尝试开发应用时，无论选择原生方式还是跨平台方式，面临的另一个重大难题是是否采用前沿技术。这些技术在应用中实施可能颇具吸引力，但

问题始终在于如何支持旧软件和旧硬件，因为大多数应用用户无法使用最新资源，因此必须将这一点考虑在内。解决这一问题的一种方法是使用支持库，它们有助于将新 API 中的新功能带到旧 API 中。

6. 维护

无论是开发桌面端还是移动设备上的软件，另一个挑战都是维护应用并分析崩溃。由于硬件和软件几乎存在无限种组合，因此很难复现崩溃，以便进行分析和修复。

面向移动设备的软件工程改进研究

1. 为有无障碍访问问题的人士设计专用用户界面

根据美国人口普查局发布的一份报告，美国约有 15% 的人口至少患有一种残障。这种残障可能是肢体、感官或其他方面的。由于关于各个群体与其移动电话之间关系的研究有限，针对人机移动交互这一领域开展的研究进展寥寥无几 [15]。

有一些通用指南，介绍如何修改现有应用程序以帮助视力障碍人士。这些指南大多建议使用“文本转语音”服务来帮助低视力或失明用户。文本转语音服务充当屏幕阅读器，以清晰响亮的声音朗读屏幕上的内容，从而减轻开发人员的工作，因为无需对应用程序的设计进行重大修改即可使其正常运行。例如，苹果在 iOS 平台中内置了“旁白”服务，因此它几乎可以适用于所有应用程序[16]。

2. 软件产品线工程

旨在降低开发成本的一种开发方法是软件产品线工程。该方法的目标是识别并归类功能相似的应用程序，并在这些应用程序之间复用现有代码／逻辑。在多平台应用程序时代，这种方法可被视为迄今最高效的开发方法之一 [17]。

《软件产品线工程：一种基于产品族的软件工程流程》一书强调，软件产品线工程主要分为两个阶段：领域工程阶段和应用工程阶段。假设一家软件公司已经定义了自己的产品线，那么领域阶段便会为整个产品线定义通用需求和特定于应用程序的需求。接下来是应用阶段，在此阶段进行实际编码，通用功能

逻辑会在整个产品线中共享和使用。产品线不一定由完全不同的应用程序组成，也可能是同一个应用程序运行在不同平台上。这种方法确实鼓励开发人员更加关注产品线中的共同要素，例如需求和设计。它能帮助开发人员理解需求，而不论软件将针对哪个特定平台编写。这种方法还改变了开发过程，将需求收集和理解置于前期，而不是从设计方法入手；理论上，这应该能加快开发过程。

3. 自适应需求

从本质上说，功能需求往往会吸引开发人员和客户的全部注意力，导致他们忽视同样至关重要的非功能需求 [4]。应用程序需要不断“自适应”，以提供用户所需的功能。在某些情况下，应用程序需要自适应，减少所提供的功能，以便更好地应对动态性。

一种不错的方法是使用现有的自适应系统需求规格说明语言，例如“RELAX” [18]。“ Relax”语言被设计为一种中间件语言，旨在明确表达行为以及

自适应系统所伴随的环境不确定性。Relax 采用简单的方法，将需求划分为必须完全满足的需求（不变项）和无需部分满足或完成的需求（可变项）。随后，所有需求都通过自然语言与模糊逻辑相结合的方式来表达。RELAX 文档接着列出了可变项需求，以及环境可能对它们造成的影响。

将 RELAX 这类自适应规格说明语言融入现有的敏捷软件工程方法中，可以为需求收集和规格说明提供更完善的结构，并全面提升环境变化情境下非功能系统需求的质量。采用这种方法，开发人员便能更好地考虑并调整应用程序在非理想环境中的行为。

Shneiderman 的八条黄金法则：设计更出色的界面

关于 Shneiderman

Ben Shneiderman 是一位美国计算机科学教授，专攻马里兰大学的人机交互。他撰写了一本关于人机交互的知名著作，名为《设计用户界面：有效策略》

《人机交互》，书中概述了界面设计的八项黄金原则。其中四项原则可以直接应用于应用程序设计，另外四项虽然仍然适用，但需要做些调整才能应用于移动应用开发 [19]。

1. 保持一致性

设计的一致性取决于在整个应用程序中使用相同的设计元素。设计元素包括图标、菜单层级、颜色和操作。用户需要轻松理解应用程序的流程，以及如何按顺序执行一系列操作来实现目标。在某个页面上通过交互获得的经验，应能迁移并应用到同一应用程序的其他页面，而无需学习新的技能。设计的一致性有助于让用户对软件产品产生熟悉感。

实现一致设计的更好方式，不仅是在应用程序的所有页面中保持一致，还要与平台的设计理念保持一致。许多安卓开发人员采用的一个例子是超级按钮菜单。如今，安卓应用程序中很常见的一种设计是在左上角放置三条平行的横线，点击后会弹出滑动式超级按钮菜单。此菜单包含所有必要选项，从设置到账户管理。主要原因在于，谷歌将超级按钮菜单作为“Material Design”界面的一部分予以推荐。运用 Material Design 设计元素，意味着用户可以将使用一种应用程序的体验迁移到另一种应用程序，从而让交互更加轻松。

2. 让常用用户能够使用快捷方式

设计应用程序时，不可忽视的一类重要用户是高级用户。高级用户的行为方式与普通用户并不相同。对高级用户而言，整部手机只是一种用来实现多个目标的工具，有时还要同时实现多个目标。因此，应为高级用户提供快捷方式，让他们能够用更短的时间、更少的步骤实现目标。

快捷方式方面一个出色的实现，是苹果在 iOS 上内置的快捷方式应用程序。该应用程序让用户能够在 iOS 上自动执行几乎任何类型的任务。这些任务可以按任意顺序组合，并能感知上下文，因此可以按时间或地点触发。这些任务的范围从晨间家务到在特殊日子向他人致以祝贺。一个很好的例子是设置一个每天在固定时间触发的晨间例行程序。它会先响起闹钟唤醒用户，然后告知用户当天的气温和降雨概率。接下来，它还可以播报当天的重要新闻、打开咖啡机，并为上班叫一辆优步。用户离开后

上班后，手机便会通过智能锁锁上家门。近年来，这类自动化曾被视为未来的梦想，如今随着移动开发的进步，已成为现实。

3. 提供有用的反馈

恰当且有效的反馈应告知用户他们执行了哪些操作，并且应及时提供。一个很好的例子是处理状态栏，它会告知用户某个流程还需要多长时间。一个很糟糕的例子是，应用程序遇到错误时只显示错误代码。用户看不懂错误代码，这会让他们感到沮丧。

4. 设计能够圆满结束对话的流程

沟通至关重要，它是人们相互理解和交流信息的方式。开发应用程序时也应遵循同样的原则。例如，完成交易流程后，应提供确认反馈。最好让应用程序显示一条消息，确认交易已成功完成或失败。即使交易失败，应用程序也应显示一条消息，说明交易失败，并明确指出错误发生的具体位置。

5. 提供简单的错误处理

开发人员绝不能想当然地认为自己的应用程序是由高级用户和经验丰富的测试人员使用的。应用程序应做到防错；即使在极少数情况下发生导致应用程序无法运行的错误，也应通过简单易懂的分步解决方案引导用户，让用户能够按步骤解决问题。

6. 允许轻松执行撤销操作

既然不应指望应用程序用户都是专家，那么当用户误触时，应该始终提供撤销某项操作或一系列操作的方法。撤销按钮常用于照片编辑应用中，因为用户可能会不小心为照片应用错误的滤镜。

7. 支持内部控制点

用户与应用程序之间良好的交互方式，是让用户感受到掌控权，并始终由他们发起操作。

8. 减轻短期记忆负荷

应用程序设计界面应采用恰当的布局和层级，保持简洁明了。设计应鼓励用户依靠再认，而不是回忆，因为前者要容易得多。使用应用程序不应像解谜或动脑筋一样。应用程序应始终为用户提供视觉或听觉线索，并随时提供相关信息。

结论

移动平台正变得日益碎片化，这既是由于软件平台种类繁多，也可能是因为同一平台内存在不同的版本风格。应用程序在两个不同平台上可能名称、设计、外观和体验都相同，但由于各平台的开发方式不同，仍会被视为两个不同的应用程序。因此，为确保一致性，必须手动将每款应用的功能与其在其他平台上的类似版本进行核对。创建通用、可复用的图形用户界面，取决于如何权衡是保持一致，还是遵循各个平台的设计语言。测试仍被视为移动应用开发行业面临的一大挑战。单元测试在移动开发社区中仍不普遍，现有测试架构也不够成熟。但责任不应全归咎于开发人员，因为目前的移动测试工具显得十分乏力，往往缺少移动电话的基本功能，例如传感器、手势导航支持和位置支持。

参考文献

[I. Malavolta，《基于 Web 的混合移动应用程序：实践现状与研究] 机遇》，2016年。

[微软，《构建跨平台[应用程序》，]在线] 。网址：_https://developer.xamarin.com/guides/cross - platform/application_fundamentals/building_cross_platform_0_applications/part_-overview/。

[J. D. a. J. Dixon，《移动应用程序] 软件工程：挑战与研究方向》，陶森大学计算机与信息科学系。

[A. I. Wasserman，《移动应用开发中的软件工程] 问题》，载于软件工程研究未来 *FSE/SDP* 研讨会，2010 年。[J. G. a. P. Tarasewich，《手持移动设备界面设计] 指南》，载于*DSI 2004* 年会论文集，2004年。[B. Fling，《移动设计与开发》，]O'Reilly，2009年。[S. Allen，《专业智能手机跨平台] 开发：iPhone、BlackBerry、Windows Mobile 和安卓的开发与发行》，2010年：Apress。

[W. S. M. P. G. L. J. A. a. W. R. T. Hofer, 8 《移动设备上的上下文感知——] Hydrogen 方法》，载于第*36*届夏威夷年会

系统科学国际会议，*2003*，夏威夷，2003。

[N. Jennings，《面向智能体的软件工程》，载于 多智能体系统] 工程，1999年。

[10] N. Jennings，《多智能体系统工程》，载于面向智能体的软件工程，1999年。

]

[M. S. M. A. S. B. M.-u.-d. A. S. Naila Kousar，1《软件工程：移动应用开发中的挑战及其解决方案》，] *(IJACSA) International Journal of AdvancedComputer Science and Applications,* 第9卷，第1期，2018年。

[12] N. G. V. N. S. a. I. L. Boushehrinejadmoradi，《跨平台移动应用开发框架测试》，2015年。
]

[A. Wasserman,《移动应用开发中的软件工程问题》，载于 3 软件未来研究研讨会（*FSE/SDP*）] 工程研究，新墨西哥州，2010年。

[S. a. K. A. Amatya，《跨平台移动开发的挑战与机遇》，4 *„ICT Innovations 2013*：*ICT* 创新与] 教育，2013年。

[M. Brault，《残疾状况与集体居住设施中人群的特征：对] 美国社区调查中非机构化平民人口及总人口的残疾患病率进行简要分析》，美国人口普查局，2008年。

[苹果公司，《iOS 人机界面指南：1 引言》，[在线]。网址：6 http://developer.apple.com/library/ios/#DO] CUMENTATION/UserExperience/Conceptual/ 移动设备HIG/Introduction/Introduction.html。

[D. W. a. C. Lai，《软件产品线工程：基于家族的软件工程流程》，Addison-Wesley，1999年。]

[P. S. N. B. B. H. C. C. a. J.-M. B. J. Whittle，《RELAX：一种用于处理] 自适应系统需求不确定性的语言》，需求工程，第15卷，第2期，第177-196页，2010年。

[交互设计基金会，1《Shneiderman 的八条黄金法则将帮助你设计出更好的界面》，2020 年。] [在线]。链接：https://www.interaction-design.org/literature/article/shneiderman-s-eight-golden-rules-will-help-you-design-better-interfaces。