2019年7月，我参与了某医疗信息系统有限公司移动护理项目的开发，担任项目技术负责人，负责系统整体架构设计及开发工作。该系统主要包含患者信息查对，医嘱执行，体征数据采集，护理文书管理等功能。本文以移动护理项目为例，介绍了微服务架构在项目中的具体应用。在服务分层设计阶段，以客户需求为依据，对系统功能进行总体服务分层架构设计，各层各施其职。在服务开发阶段，对较粗粒度的服务进一步细化成粒度更小微服务，并采用开源组件框架进行有效组装，完成项目微服务化开发。在部署阶段，采用容器化技术，完成微服务的容器化部署和集群管理。通过以上阶段的实施，我们完成了软件项目开发工作，保证了项目质量。最终项目在某省三甲医院成功落地，而后也陆续收到其它医院的上线申请。

移动医疗时代的到来，为实现医疗无纸化，实现医嘱的全生命周期的跟踪，有效解决护士站到病房的＂最后１０米＂的信息延伸问题提供了更好的解决方案．最终达到改变医疗的存在方式，改善医疗业务流程，提升医疗服务质量，提高医疗安全性的目的．

2019年７月，某医疗信息系统有限公司启动了移动护理系统项目。此系统以电子病历系统ＥＭＲ和医嘱录入系统ＣＰＯＥ为平台，借助无线手持终端设备PDA进行信息的实时化传输和移动采集，实现患者信息查对，医嘱执行，体征数据采集，护理文书管理等功能．患者信息查对主要实现PDA扫描患者床头卡二维码或患者腕带条形码查询患者基本信息，核对患者身份；医嘱执行主要实现PDA扫描药品信息二维码，进行药品领取核对、记录护士医嘱执行过程，完成皮试药品双人核对，进行检验样本管扫码采集等功能。体征数据采集主要实现体征任务提醒，患者体征数据录入存库。护理文书管理主要实现患者量表，特护单，通用单，评估单等文档的编辑。我在项目中担任技术负责人，负责系统整体架构设计及开发工作，最终项目在某省著名三甲医院成功落地。

随着微服务架构的发展，越来越多的团队采用微服务架构进行项目开发。相比于单体架构，微服务架构在支持异构性、弹性、扩展性以及简化部署等方面有着得天独厚的优势。在支持异构性方面，微服务架构中，每个服务都是一个相对独立的个体，每个服务都可以选择适合于自身的技术来实现；在弹性方面，单体系统的一个部分出现问题，可能导致整体系统的问题。而微服务架构中，每个服务可以内置可用性的解决方案与功能降级方案，所以比单体系统强；在扩展性方面，单体系统往往是整体进行扩展，而微服务架构中，可以针对单个服务进行扩展；在简化部署方面，大型单体系统即使修改一行代码，也需要重新部署整个应用系统，影响很大、风险很高，因此不敢轻易的重新部署。而微服务架构中，每个服务的部署都是独立的，这样就可以更快地对特定部分的代码进行部署。

综合考虑微服务架构的各方面优势，我们决定采用微服务架构开发移动护理项目。下面分别从微服务分层设计、微服务开发、微服务部署三个阶段展开具体说明。  
1.服务分层设计阶段

为了使系统整体各项职能清晰合理、便于扩展维护，我们以客户需求为依据，对系统功能进行总体服务分成架构设计。首先，系统管理员在后台为护士分配病区功能权限、配置医嘱执行分类、安排体征采集任务；然后，护士使用PDA查对患者信息和医嘱药品信息后为患者用药，根据体征任务采集患者的体征数据并填写护理文书；最后，管理员可在后台查看各病区的统计信息，比如医嘱执行情况统计、护士工作量统计。通过分析客户的场景需求，我们将整个系统自上而下划分为前端应用层、API网关层、微服务层、数据存储层。前端应用层涉及Web前端、移动客户端，提高人机交互。API网关层实现身份验证、负载均衡、日志监控、路由等基础服务。微服务层包含患者服务、医嘱执行、体征采集、护理文书管理、工作量统计等。数据存储层包括数据库和分布式缓存。通过服务分层设计，降低了服务开发复杂度，提高了系统的可伸缩性。  
2.服务开发阶段

划分好各个层次后，为了进一步明确各层次的需求功能以及团队的任务分配，我们将各层次粗粒度的服务进一步细化为粒度更小的微服务。前端应用层按技术细分为Web配置管理服务、Android扫码应用服务、H5文书管理服务；微服务层按业务功能细分为患者管理、医嘱执行管理、体征采集管理、护理文书管理等。微服务各部分相互协作才能完成业务流程，因此需要将各个微服务有效组装起来，我们选择开源的微服务治理框架Spring Cloud。服务注册中心使用Eureka组件，专门提供微服务的服务注册和发现功能。路由网关使用Zuul组件，所有前端应用调用后端服务统一从Zuul网关进入然后进行转发，可以完成身份认证和熔断限流等功能。使用Ribbon组件实现客户端负载均衡，Ribbon可自定义负载均衡策略，默认情况下我们采用BestAvialableRule策略，过滤掉故障的服务并选择并发量最小的服务。为了帮助系统容错限流，使用Hystrix组件为系统提供保护和控制。这些功能组件和业务服务都注册到服务注册中心，实现所有微服务的整合，为整个系统提供功能完善、高可靠的服务。  
3.服务部署阶段

系统采用微服务架构设计，原本庞大、复杂的单体系统拆分成数量众多的微服务，这让系统有了更好的可用性、可扩展性、利于小团队开发等优点，但是每个服务都需要进行单独部署，极大增加了系统维护复杂性。如果进行手工部署明显工作量较大。为了解决这个问题，我们采用Jenkins+Docker技术实现自动容器化部署。服务开发者将代码提交到代码管理平台，在Jenkins平台创建自动化流水线任务，拉取远程代码后自动将源代码实现打包，生成Docker镜像并上传到镜像仓库中，然后将项目镜像进行部署。这样就自动化部署完成，再经由测试人员严格测试通过后，一键部署到生产环境中。通过这种自动化部署方式，很好的解决了微服务数量多，手工部署复杂繁琐和服务集群部署的问题，显著提高了开发部署效率。

基于微服务架构设计，项目顺利开发完成，于2021年1月在某省三甲医院的成功上线。项目运行期间，各项性能指标均达标，系统保持95%的使用率，获得患者，护士及医院领导的一致好评。可以说明微服务架构设计在此项目中的运用是一次成功的实践．为我们提供了一套理论结合实践的成功经验。

尽管如此，我们为了减轻数据库服务器压力而采用了Redis缓存数据库，但在系统运维中发现，部分针对缓存的查询，未在缓存系统中找到对应的key,从而引发了大量对数据库服务器的查询请求，最严重时甚至导致了数据库服务宕机。一开始解决方案是在数据库中未查走到该key时，在缓存系统中key设置空值，防止对数据库服务器发起重复查询。但后面发现，缓存中有大量的key值都为空，消耗大量内存资源，导致系统的性能急剧下降。最终我们采用布隆过滤器，将所有可能非空数据的key值哈希到bitmap中，查询缓存之前，过滤掉不在bitmap中的key值，从而避免对缓存和数据库的查询。通过这次软件开发工作，让我认识到优秀的理论知识和实践相结合才能成就好的软件项目，这敦促我在工作中也不忘提升理论知识，最终成为一名优秀的软件架构师。