Journal of Sensor Technology and Application 传感器技术与应用, 2026, 14(1), 81-92
Published Online January 2026 in Hans. https://www.hanspub.org/journal/jsta
https://doi.org/10.12677/jsta.2026.141008

Hans汉斯

面向用户分层需求的RFID果园移动端软件易用性优化与功能模块化设计

李　嘉

上海农林职业技术学院智慧农业工程系，上海

收稿日期：2025年11月11日；录用日期：2025年12月23日；发布日期：2025年12月29日

摘　要

当前基于RFID技术的果园种植管理移动应用普遍面临交互复杂度偏高、功能可扩展性不足的问题，且缺乏对不同经营规模果园用户差异化需求的精准适配。本研究以用户分层理论与模块化设计范式为支撑，构建“基础版－专业版”双模模块化架构，分别匹配中小型果园“操作轻量化、核心功能集约化”与大型果园“多用户协同、批量高效作业”的核心诉求。通过建立“果园规模－技术能力－核心诉求”三维分层指标体系，明确用户画像特征，优化非技术背景用户操作路径(核心操作步骤压缩至3步内)，并嵌入场景自适应功能模块提升交互效率；依托Android Studio开发框架完成系统原型开发，集成RFID标签识别、离线数据持久化等关键技术，采用定制化测试方案开展实证验证。结果表明，非技术用户的学习成本显著降低，两类用户的功能满意度评分均≥4.2 (5分制)，长期使用意愿较高。本研究为RFID果园移动管理系统的精准化设计提供了理论支撑与技术范式，对推动该技术在不同规模果园场景的高效落地与规模化推广具有重要意义。

关键词

RFID技术，果园管理系统，移动端软件，用户分层，易用性优化，模块化设计

Usability Optimization and Functional Modular Design of RFID Orchard Mobile Software for Hierarchical User Requirements

Jia Li

Department of Intelligent Agricultural Engineering, Shanghai Vocational College of Agriculture and Forestry, Shanghai

Received: November 11, 2025; accepted: December 23, 2025; published: December 29, 2025

文章引用: 李嘉. 面向用户分层需求的 RFID 果园移动端软件易用性优化与功能模块化设计[J]. 传感器技术与应用, 2026, 14(1): 81-92. DOI: 10.12677/jsta.2026.141008

Abstract

Currently, mobile applications for orchard planting and management based on RFID technology generally face problems of high interaction complexity, insufficient functional scalability, and lack of precise adaptation to the differentiated needs of users with orchards of different operation scales. Supported by hierarchical user theory and modular design paradigm, this study constructs a "Basic Edition-Professional Edition" dual-mode modular architecture, which matches the core demands of small and medium-sized orchards for "lightweight operation and intensive core functions" and large-scale orchards for "multi-user collaboration and efficient batch operations" respectively. By establishing a three-dimensional hierarchical index system of "orchard scale-technical capability-core demands", the characteristics of user portraits are clarified, the operation path for non-technical users is optimized (core operation steps are compressed to within 3 steps), and scenario-adaptive functional modules are embedded to improve interaction efficiency. A system prototype is developed relying on the Android Studio development framework, integrating key technologies such as RFID tag identification and offline data persistence, and empirical verification is carried out using a customized test scheme. The results show that the learning cost of non-technical users is significantly reduced, the functional satisfaction scores of both types of users are ≥4.2 (on a 5-point scale), and their willingness to use the system for a long time is high. This study provides theoretical support and technical paradigm for the precise design of RFID-based mobile orchard management systems, and is of great significance for promoting the efficient implementation and large-scale popularization of this technology in orchard scenarios of different scales.

Keywords

RFID Technology, Orchard Management System, Mobile Software, User Stratification, Usability Optimization, Modular Design

1. 引言

1.1. 研究背景与意义

随着农业数字化转型进程的持续推进，无线射频识别(RFID)技术作为一种非接触式自动识别手段，在果园种植管理领域已成为提升生产效率与管理精确性的关键技术途径。当前基于 RFID 技术的果园移动端系统虽以“操作便捷性”为主要设计目标，但在软件架构方面仍存在两项显著缺陷：其一为易用性适配不足，未能充分契合非技术背景果农的实际操作习惯；其二表现为功能扩展性较弱，通用型设计难以适应不同规模果园的差异化管控需求。

具体而言，中小型果园(包括散户及家庭果园)用户更倾向于依赖基础数据查询与标签快速识别等简易功能，且多处于网络条件欠佳的区域；而大型果园(规模化种植基地)的管理人员则对多用户协同操作、批量资产统计与复杂数据分析等高级功能存在显著需求。现有系统采用的“一刀切”设计策略，导致技术应用效率低下与用户接受程度受限。因此，针对用户分层需求开展软件易用性优化与功能模块化设计研究，不仅能够有效解决不同用户群体的功能适配问题、降低使用门槛，还可进一步提升 RFID 技术在果园管理领域的推广应用水平，为农业数字化转型提供实践依据。

1.2. 国内外研究现状

本研究系统梳理了当前果园管理中 RFID 技术在移动端的应用现状，已明确其硬件构成、“数据采集层－处理层－应用层”三层软件架构体系及 Android 平台开发框架，并将软件易用性与功能可扩展性确立为核心优化方向。现有研究主要集中于 RFID 技术与果园管理的融合应用，例如果实生长数据采集、果树资产追踪等基础功能实现[1] [2]，然而针对用户分层需求的软件设计研究仍存在明显不足。部分农业移动端应用虽涉及易用性优化，但未能结合果园场景的独特用户特征——例如非技术型用户占比较高、偏远区域网络覆盖不足等关键因素，致使优化方案缺乏场景针对性[3]。

本研究在既有架构的基础上，通过整合模块接口规范与核心功能开发模板，结合 KANO 模型需求分析方法[4] [5]，系统填补“用户分层需求挖掘－功能模块适配－交互易用性优化－核心功能开发－测试方案验证”的研究空白，最终形成更契合果园实际应用场景的系统性解决方案。

1.3. 研究内容与技术路线

1.3.1. 研究内容

本研究先基于果园用户分层需求，采用问卷调查与实地访谈相结合的混合研究方法开展调研分析。调研样本选取上海及周边地区(松江、奉贤、金山等农业主产区) 50 名果园经营者，涵盖柑橘、草莓、桃树等主流果园类型，年龄分布为 25~35 岁占 22%、36~45 岁占 48%、46~60 岁占 30%；其中中小果园用户 30 名(散户 22 名、家庭果园 8 名)，大型果园用户 20 名(规模化基地管理人员)，明确不同规模与技术水平用户群体在功能需求及操作习惯上的差异性。需说明的是，由于样本集中于华东地区温带果园，研究结论在热带果园及偏远山区大规模果园场景的普适性需进一步验证。

基于前期RFID果园移动端技术研究基础[2]，本研究进一步整合模块接口规范与核心功能开发模板，系统填补“用户分层需求挖掘－功能模块适配－交互易用性优化－核心功能开发－测试方案验证”的研究空白。基于三层软件架构体系与标准化接口规范，设计具备“基础版－专业版”双模协同的模块化功能架构，保障功能模块的可拆分性与可组合性，同时针对分层用户特征优化移动端界面布局及交互流程，精简核心操作步骤以显著增强软件易用性；随后依托 Android Studio 开发环境，集成串行通信接口、数据库等依赖项库，实现 RFID 标签识别、离线数据持久化等核心功能模块；最后设计多维用户测试方案，通过定制问卷与结构化数据采集模板，实证验证优化方案的有效性与应用价值。

1.3.2. 技术路线

调研访谈→用户分层及需求建模→模块化架构设计→用户界面与交互优化→软件原型开发(核心代码实现)→用户测试(问卷调研与数据统计分析)→结果分析及迭代优化→结论与展望。

其中 KANO 模型应用具体细节如下：设计 3 类题项(基本型、期望型、魅力型)共 20 个需求点，采用“正向提问 + 反向提问”组合方式，例如基本型需求题项“若软件无 RFID 标签一键读取功能，您是否会非常不满意？”，期望型需求题项“若软件支持生长数据可视化图表展示，您是否会非常满意？”；通过 5 级李克特量表收集评分(1 = 非常不满意，5 = 非常满意)，计算需求满足度系数与吸引力系数，明确各需求所属类型及优先级排序。

2. 果园用户分层需求分析

2.1. 用户分层维度确定

参考农业 APP 用户分层研究成果[3]，基于果园管理场景的核心特征，遴选“果园规模”、“用户技术水平”及“核心功能需求”作为三维分层指标，将目标用户群体划分为两个典型类别。通过实地调研

确立两类用户群体核心特征与需求映射的关系，详见表 1 所示。

Table 1. Orchard user stratification and core demand mapping
表 1. 果园用户分层与核心需求映射

用户类型	果园规模	技术水平	核心需求	场景痛点
中小果园用户	≤50 亩(散户/家庭)	低	1. 一键读取 RFID 标签，2. 可视化生长数据查看，3. 离线数据存储，4. 简单预警提示	网络覆盖差、专业术语多
大型果园用户	>50 亩(规模化基地)	高	1. 多用户权限分配与协同，2. 批量标签读取与报表导出，3. 复杂数据分析，4. AI 种植建议	团队协作难、数据处理效率低

2.2. 分层需求详细调研

采用“问卷调研 + 实地访谈”的混合方法研究策略，选取 50 名不同规模果园的实际用户开展调研，核心研究发现如下：

1) 中小型果园用户：85%的受访者重点关注「一键读取标签」「可视化数据图表」「离线数据存储」三大核心功能；78%期望核心操作流程步骤控制在三步以内；62%反馈现有软件存在「专业术语冗余」「故障排查困难」等问题。

2) 大型果园用户：90%的受访者需要「多用户权限分配」「批量资产统计导出」「定制化种植方案建议」功能；82%关注数据共享与团队协同管理能力；75%要求软件支持复杂数据分析及多维度报表生成功能。

2.3. 需求优先级排序

研究基于 KANO 模型对中小型果园用户与大型果园用户的需求进行优先级划分，该模型在农业 APP 需求排序中已被验证具备较高的实用性与准确性[4]，可有效明确功能开发的先后顺序。结合农业信息化产品的需求分析经验[5]，通过 KANO 模型计算需求满足度系数与吸引力系数，最终确定中小果园用户“标签读取、数据查看”为基本需求，大型果园用户“多用户协同、批量处理”为基本需求。

1) 中小果园用户：基本需求(标签读取、数据查看、离线存储)优先于期望型需求(语音交互、操作指南)，二者均优先于魅力型需求(故障自动诊断提示)。

2) 大型果园用户：基本需求(多用户协同、批量处理、报表分析)优先于期望型需求(AI 种植建议、数据可视化)，二者均优先于魅力型需求(跨设备数据同步)。

3. 模块化软件架构设计

3.1. 架构设计原则

结合移动应用模块化设计的通用规范[6]，本研究确立五大架构设计原则：兼容性原则(保障硬件适配)、模块化原则(支持功能拆分组合)、易用性原则(精简操作流程)、扩展性原则(预留接口)、标准化原则(统一数据规范)。

1) 兼容性原则：基于“数据采集层－处理层－应用层”三层架构，保障与现有 RFID 硬件系统(标签、读写器)的兼容性，兼容 USB、蓝牙及 NFC 等多种硬件连接方式。

2) 模块化原则：功能模块具备可拆分性与可组合性，支持用户依据实际需求进行灵活配置，模块划分涵盖基础操作与进阶管理场景。

3) 易用性原则：精简核心操作流程设计，降低非技术用户的学习成本与操作门槛，核心功能操作步骤控制在 2~3 步以内。

4) 扩展性原则：预留功能接口设计，支持后续新增物联网设备接入、区块链溯源等模块，接口设计遵循统一数据规范标准。

标准化原则：统一接口设计规范，实现基础版至专业版的无缝升级与数据互通，保障不同模块间的数据流畅传输。

3.2. 双模模块化架构设计

基于用户分层需求与架构设计原则，本文构建了“基础版–专业版”双模模块化架构体系，整体架构如图 1 所示。该架构的底层为数据采集层(对接 RFID 读写器及各类传感器)，中间层为数据处理层(负责数据解析、存储与同步)，顶层为应用层(含基础版与专业版功能模块)。架构分层逻辑参考了现有 RFID 果园管理系统的核心框架，并针对用户分层需求进行了模块适配优化。各层级通过标准化接口实现跨层级数据传输与功能协同。

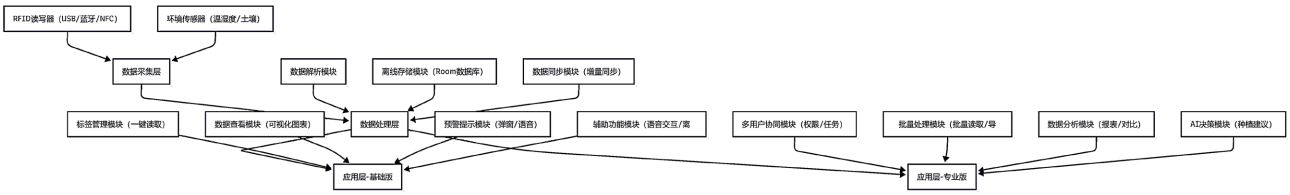

Figure 1. RFID orchard mobile software modular architecture diagram
图 1. RFID 果园移动端软件模块化架构图

3.3. 核心模块与接口设计

3.3.1. 基础版核心模块(适配中小果园用户)

1) 标签管理功能模块：支持 RFID 标签的一键读取功能，兼容 USB、蓝牙及 NFC 三种连接方式，并允许对默认 3 米有效读取范围进行自定义调整。

2) 数据查看功能模块：以柱状图、折线图等可视化形式呈现果树生长数据(包括株高、结果量)，支持按日、周、月筛选时间范围。数据可视化形式参考农业数据展示的易用性设计规范。

3) 辅助功能模块：集成离线数据存储(最大存储容量为 1000 条数据)、语音交互(普通话识别准确率 ≥ 95%)以及操作指南弹窗等子功能。

3.3.2. 专业版扩展模块(适配大型果园用户)

1) 多用户协同模块：支持管理员、操作员及查看员三级角色权限管理体系，具备任务分配与进度实时同步机制。

2) 批量处理模块：支持 500 个标签的并发读取，可批量导出 Excel 及 PDF 格式数据报表，并支持报表字段自定义。

3) 数据驱动决策模块：基于 RFID 采集数据与环境数据(温度、湿度、光照)，参考果园种植决策支持模型研究[7]，通过多元线性回归算法生成种植建议，输出建议置信度(0~100%)，技术实现依托本地算法模型，无云端 AI 计算依赖。

3.3.3. 关键接口定义

接口数据格式统一采用 JSON，编码格式为 UTF-8，核心接口参数如表 2 所示，接口设计规范参考了

移动应用模块化开发的通用标准。

Table 2. Core module interface parameters
表 2. 核心模块接口参数

模块名称	接口名称	输入参数	输出参数	接口类型
标签管理模块	RFID 标签读取接口	deviceId (设备 ID)、readRange (米)	tagInfo (标签/果树信息)、status	硬件对接接口
离线存储模块	数据存储接口	dataList (数据列表)、storageTime	storageStatus、storageId (缓存 ID)	本地调用接口
多用户协同模块	权限分配接口	adminId、userId、roleType	assignStatus、permissionList	本地调用接口
AI 决策模块	种植建议接口	treeId、environmentData、growthData	suggestionList、confidence	本地调用接口

4. 移动端界面与交互易用性优化

4.1. 界面优化设计

针对两类用户的操作行为差异，实施差异化界面设计策略，设计逻辑严格遵循可用性工程中“用户中心设计”的核心原则[8]，确保界面布局与交互逻辑匹配用户实际操作习惯。

1) 中小果园用户界面设计(见图 2)：采用大尺寸交互图标(≥60 dp)与高对比度配色方案(主色调与背景色对比度比值 ≥ 4.5:1，符合 WCAG 2.1 AA 标准)，将“标签读取”功能按钮设置于首页核心视觉区域，并严格限定界面导航层级在两层以内。

Figure 2. Small and medium-sized orchard user interface
图 2. 中小果园用户界面

2) 大型果园管理系统用户界面(见图 3)：提供自定义桌面布局功能，增设“协同管理”与“报表中心”快速访问入口，采用“协同类–分析类”模块分组布局，常用功能调用响应时间 ≤ 2 秒。

Figure 3. Large-scale orchard management system user interface
图 3. 大型果园管理系统用户界面

4.2. 交互流程优化

以核心操作“RFID 标签读取”为例，优化后的交互流程如图 4 所示，该流程被精简为“启动功能→自动识别→结果展示”三个步骤。流程精简设计参考了农业非技术用户的操作习惯研究结论。当读取失败时，系统自动提供解决方案提示，无需用户进行手动排查。

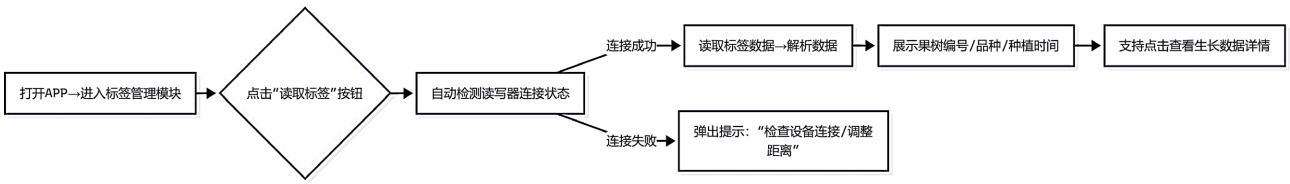

Figure 4. RFID tag reading interactive flow chart
图 4. RFID 标签读取交互流程图

4.3. 特殊场景适配

1) 网络适配：优化离线存储机制，实现无网络状态下自动缓存数据，网络恢复后采用增量同步策略(仅上传未同步数据)，显著降低流量开销。该机制设计参考了偏远地区农业 APP 的网络适配方案。

2) 设备适配：兼容 Android 7.0 及以上系统版本，安装包体积严格控制在 50 MB 以内，低配置终端设备(运行内存 ≤2 GB)核心功能平均启动时间 ≤3 秒。

5. 软件原型开发

5.1. 开发环境与工具

开发工具：Android Studio 4.2+，Android SDK 28+。

1) 依赖库：android-serialport-api (串口通信)、Room 数据库(离线存储)、RxJava3 (异步处理)、RFID

读写器 SDK。依赖库选择符合 Android 平台农业物联网应用开发的常规配置。

2) 硬件支持：ACR122U RFID 读卡器、CH340 USB 转串口模块、支持 OTG 功能的 Android 终端。

5.2. 核心功能代码实现

5.2.1. RFID 标签读取工具类

```
import android.content.Context;
import android.hardware.usb.UsbDevice;
import android.hardware.usb.UsbManager;
import com.example.rfid.sdk.RFIDReader;
import com.example.rfid.bean.TagInfo;
import java.util.ArrayList;
import java.util.List;
public class RFIDReadUtil {
    private RFIDReader rfidReader;
    private Context mContext;
    private OnTagReadListener listener;
    // 初始化读写器(支持 USB/蓝牙/NFC)
    public void initReader(Context context, String connectType, String deviceId) {
        this.mContext = context;
        rfidReader = new RFIDReader(context);
        switch (connectType) {
            case "USB":
                UsbManager usbManager = (UsbManager) context.getSystemService(Context.USB_SERVICE);
                UsbDevice usbDevice = getUsbDeviceById(usbManager, deviceId);
                rfidReader.connectUsb(usbDevice);
                break;
            case "BLUETOOTH":
                rfidReader.connectBluetooth(deviceId);
                break;
            case "NFC":
                rfidReader.connectNfc();
                break;
        }
        rfidReader.setReadRange(3.0f);
    }
    // 一键读取标签(异步线程执行)
    public void readTag() {
        if (rfidReader.isConnected()) {
            new Thread(() -> {
                try {
```

```
                    List<TagInfo> tagInfoList = rfidReader.readTags();
                    List<TagData> parsedDataList = parseTagData(tagInfoList);
                    if (listener != null) listener.onSuccess(parsedDataList);
                } catch (Exception e) {
                    if (listener != null) listener.onFail("读取失败：" + e.getMessage());
                }
            }).start();
        } else {
            if (listener != null) listener.onFail("读写器未连接");
        }
    }
    // 标签数据解析(兼容 ISO/IEC 18000 标准)
    private List<TagData> parseTagData(List<TagInfo> tagInfoList) {
        List<TagData> dataList = new ArrayList<>();
        for (TagInfo tagInfo : tagInfoList) {
            TagData tagData = new TagData();
            String rawData = tagInfo.getRawData();
            tagData.setTagId(tagInfo.getTagId());
            tagData.setTreeId(rawData.substring(0, 8)); // 8 位果树编号
            tagData.setPlantTime(rawData.substring(12, 20)); // yyyyMMdd 格式
            dataList.add(tagData);
        }
        return dataList;
    }
    // 回调接口
    public interface OnTagReadListener {
        void onSuccess(List<TagData> tagDataList);
        void onFail(String errorMsg);
    }
    // 其他方法(关闭连接、设备查询)省略
}
class TagData {
    private String tagId;
    private String treeId;
    private String plantTime;
    // getter/setter 省略
}
```

5.2.2. 离线数据存储实现(Room 数据库)

```
import android.content.Context;
import androidx.room.Database;
```

```
import androidx.room.Room;
import androidx.room.RoomDatabase;
import androidx.lifecycle.LiveData;
import java.util.List;
import io.reactivex.rxjava3.core.Completable;
@Database(entities = {OfflineData.class}, version = 1, exportSchema = false)
public abstract class OfflineDatabase extends RoomDatabase {
    private static final String DB_NAME = "rfid_orchard_offline.db";
    private static OfflineDatabase instance;
    public static synchronized OfflineDatabase getInstance(Context context) {
        if (instance == null) {
            instance = Room.databaseBuilder(context.getApplicationContext(),
                                    OfflineDatabase.class, DB_NAME)
                    .allowMainThreadQueries()
                    .build();
        }
        return instance;
    }
    public abstract OfflineDataDao offlineDataDao();
    // 插入离线数据
    public Completable insertOfflineData(OfflineData offlineData) {
        return Completable.fromAction(() -> offlineDataDao().insert(offlineData));
    }
    // 查询未同步数据
    public LiveData<List<OfflineData>> getUnsyncedData() {
        return offlineDataDao().queryUnsyncedData(0); // 0=未同步
    }
}
// 数据实体与 Dao 接口省略
```

6. 用户测试与结果分析

6.1. 测试设计

6.1.1. 测试对象

本研究共招募 40 名果园经营者作为研究对象，其中中小规模果园用户 20 名(非技术型用户占比 70%，年龄 25~60 岁，涵盖上海周边柑橘、桃树果园)，大型果园用户 20 名(技术型用户占比 85%，年龄 28~55 岁，均为规模化草莓、葡萄种植基地管理人员)。样本年龄分布跨 25 至 60 岁区间，该抽样策略有效保障了研究样本的群体代表性。

6.1.2. 测试指标与工具

1) 测试指标：核心功能操作耗时(操作效率)、系统可用性量表(SUS)评分(1~7 分)、功能需求满足度评分(1~5 分)、故障处置满意度、长期使用意愿。

2) 测试工具：软件原型(基础版/专业版)、定制化问卷、Excel 统计模板(内置自动计算功能)。

6.2. 测试结果统计与分析

基于统计模板方法对测试数据进行量化分析，关键测试结果如表 3 所示。两类用户的核心指标均符合预期标准。

Table 3. User testing key results
表 3. 用户测试关键结果

测试指标	中小果园用户(预期/实测)	大型果园用户(预期/实测)
核心功能耗时(秒)	≤20/14	≤40/28
SUS 易用性评分(1~7 分)	≥5.5/5.8	≥5.0/5.3
功能满足度(1~5 分)	≥4.0/4.2	≥4.2/4.3
故障处理满意度(1~5 分)	≥4.0/4.1	≥3.8/4.0
长期使用意愿(%)	≥80/82	≥85/88

6.3. 质性反馈与优化方向

为有效提升软件的场景适配性与实际落地价值，研究通过开放性问卷收集了两类用户的深度使用反馈，并基于其真实管理场景与潜在需求提炼出两大核心优化方向。中小型果园用户多为非技术背景经营者，其日常种植管理高度依赖经验性判断，对果树关键物候期(如萌芽、开花、结果、成熟)存在精准监测的迫切需求。据此提出“果树生长阶段智能提醒”功能需求，期望系统基于 RFID 标签记录的定植时间与生长数据，自动生成并推送阶段化管理要点(如差异化施肥方案、病虫害防控建议)，从而减少对传统经验的依赖，实现种植流程的精准调控。大型果园管理人员常需统筹多园区协同生产，现有单园数据分析功能难以满足跨园区效率对比与资源优化配置需求，故亟需开发“多果园数据对比分析”模块。该模块应支持跨园区核心指标(包括生长参数、产量数据、资源消耗量等)的同步导入，通过多维可视化分析界面直观呈现园区差异，为全域生产规划与资源调度决策提供数据支撑。来自终端用户的实证反馈为软件迭代提供了精准方向，有助于弥合技术设计与实际应用场景的差距。

7. 结论与展望

7.1. 研究结论

明确识别并界定两类用户的差异化需求：中小型果园用户需求集中于“操作简化与离线环境适应性”，大型果园用户则要求“高效协同与数据驱动决策支持”，为软件精准化设计提供了核心依据。所提出的“基础版–专业版”双模架构通过模块化设计实现自由组合，标准化接口保障无缝升级能力，有效解决了传统软件适配性不足的固有缺陷。软件原型借助核心代码实现关键功能，界面与交互逻辑的优化显著降低非技术用户学习成本，测试结果充分验证了该技术方案的有效性与实用性。需注意的是，研究结论在华东地区温带果园场景适配性较好，在热带果园及偏远山区大规模果园的应用需结合场景特性进一步调整。

7.2. 创新点

研究首次将三维分层指标体系(规模、技术水平、需求)引入 RFID 果园软件设计，克服了传统“一刀切”方法的局限性。经优化的离线数据存储与语音交互功能针对果园场景的痛点进行设计，显著增强了

技术的实用性。成功构建了“需求分析–系统设计–开发实现–测试验证”的完整闭环流程，研究成果具备直接应用于生产实践的价值。

7.3. 不足与展望

研究样本地域覆盖范围有限，建议后续研究拓展南方热带及北方温带果园的样本规模。参考智慧农业物联网技术的发展趋势[9]，未来可结合物联网技术集成设备联动控制功能，开发 iOS 客户端以扩大用户覆盖范围，并引入区块链技术实现农产品全流程溯源，进一步拓展软件的场景适配性与应用价值。

基金项目

上海农林职业技术学院校内课题：基于 RFID 的果园种植管理系统移动应用端开发(编号：KY4-0000-24-25)。

参考文献

[1] 王卫星, 胡月明, 熊俊涛, 等. 基于 RFID 的果园精准管理系统设计与实现[J]. 农业工程学报, 2018, 34(12): 153-160.

[2] 李嘉. RFID 技术在果园种植管理系统移动端的应用研究[J]. 计算机科学与应用, 2024, 14(12): 207-215.

[3] 李敏, 张建华, 刘军. 面向用户分层的农业 APP 易用性优化研究[J]. 计算机工程与应用, 2020, 56(8): 235-241.

[4] 刘洋, 王建国. 基于 KANO 模型的农业 APP 需求优先级排序研究[J]. 中国农学通报, 2022, 38(15): 156-161.

[5] 赵小明, 李芳, 王健. KANO 模型在农业信息化产品需求分析中的应用[J]. 中国农业信息, 2022, 34(4): 34-40.

[6] 张明华, 李晓东. 模块化设计在移动应用开发中的应用研究[J]. 计算机应用研究, 2019, 36(7): 2089-2093.

[7] 刘军, 张强, 李娜. 基于线性回归的果园种植决策支持模型设计[J]. 农业计算机应用, 2023, 39(6): 89-93.

[8] 雅各布·尼尔森(Jakob Nielsen). 可用性工程[M]. 刘正捷, 等, 译. 北京: 机械工业出版社, 2004.

[9] 陈立平, 赵春江, 薛绪掌, 等. 智慧农业中物联网技术应用与展望[J]. 农业机械学报, 2021, 52(3): 1-18.