# 工业生产过程数据监控客户端软件

# 设计规格说明书

版本号：V1.1

修改日期：2018-12-8

责任人：卢安

# 目 录

目录

[工业生产过程数据监控客户端软件 1](#_Toc532232779)

[设计规格说明书 1](#_Toc532232780)

[目 录 2](#_Toc532232781)

[第1章 概述 3](#_Toc532232782)

[1.1 背景与需求综述 3](#_Toc532232783)

[1.2 实现平台 3](#_Toc532232784)

[1.3 历史版本 4](#_Toc532232785)

[第2章 软件功能 5](#_Toc532232786)

[第3章 软件框架结构 6](#_Toc532232787)

[3.1 一级框架结构 6](#_Toc532232788)

[3.2 SAction框架结构 7](#_Toc532232789)

[3.3 While(1)框架结构 8](#_Toc532232790)

[第4章 功能模块定义 9](#_Toc532232791)

[4.1 与文件、目录、时间有关的功能模块 9](#_Toc532232792)

[4.2 与日志操作有关的功能模块 9](#_Toc532232793)

[4.3 与序列文件相关的功能模块 10](#_Toc532232794)

# 第1章 概述

## 1.1 背景与需求综述

工业生产过程有部分关键工艺，其工业参数往往与物料、环境、客户使用条件、客户特殊要求直接相关。对于部分工艺参数，其设置值与执行设备的反馈检测值与产品质量和品质直接相关。

对于部分关键工艺参数，需要追溯其设置的全过程，执行设备反馈的全过程。而部分老旧生产设备，其控制软件不具备追溯全过程的功能。对其进行软件的升级改造十分必要。

考虑到部分老旧工业设备，硬件配置低，操作系统版本低，在其上进行开发无发有效运行第三方开发库，只能使用基础的WindowsAPI开发，故本软件在仅能使用WindowsAPI的强约束下，进行对生产过程屏幕显示的抓屏、简单图形压缩存储，硬盘循环存储等功能。

考虑到工业现场设备硬件配置低，对追溯信息的恢复不在现场设备中完成。故将整个生产过程的数据监控功能划分为客户端——即在工业现场设备中安装并运行的软件，和服务器端——针对压缩的数据文件进行恢复的软件，这两部分软件。

## 1.2 实现平台

本软件针对工业生产过程大量的老旧设备，这些设备通常仅有32M的内存，运行WindowsNT/Windows2000/WindowsXP等操作系统，硬盘通常为40G左右，通常不具备或不允许网络连接。这样的要求通常不允许使用第三方的中间件，也无法高效的运行Windows的Framework。故本软件仅运行使用WindowsNT之后的版本操作系统共同支出的API函数，为基于API函数的纯API编程开发。

## 1.3 历史版本

0.1版，2017-4-8，在某型号中空玻璃全自动生产线钢化炉工控计算机开始试用。

0.2版，2017-7-30，支持更多型号的中空玻璃全自动生产线钢化炉工控计算机。

1.0版，2018-3-10，支持X射线真空排气室工控计算机。

1.1版，2018-12-3，支持X射线靶盘加热炉工控计算机。

# 第2章 软件功能

本软件为工业生产过程数据监控的客户端软件，软件开机后自动运行，在设定路径下建立相关的文件夹，无界面显示，实时抓取屏幕显示内容，压缩并存储图像文件。同时检测硬盘剩余空间，循环存储记录内容。主要功能如下：

* 开机自动运行
* 在约定路径下按日期建立文件夹，记录时间过零点后，建立新的文件夹
* 定时触发存储显示界面图像
* 按照公司独立知识产权的压缩算法压缩图像文件，并将多个图像文件存储为一个序列文件
* 循环存储序列文件到硬盘
* 记录数据存储日志

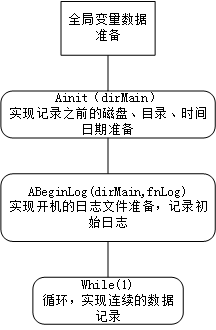
# 第3章 软件框架结构

## 3.1 一级框架结构



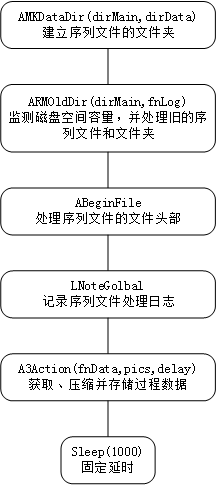
一级框架仅包含SAA3.cpp和S3Action.cpp两个cpp文件，由于无图形界面和联网功能，全部的功能均在SAction函数中，这里预留图形界面和联网功能。

## 3.2 SAction框架结构



SAction是数据记录的主函数，分为初试化过程和while（1）过程。

## 3.3 While(1)框架结构



While（1）为数据记录的关键过程，围绕着磁盘准备、新旧文件夹的处理，序列文件头部处理，过程数据获取、压缩和存储等固定过程，反复循环执行。

# 第4章 功能模块定义

## 4.1 与文件、目录、时间有关的功能模块

4.1.1 int GetMainDir(char \* dirMain)

基于配置文件（SAS.cfg）得到预设的主工作路径。

4.1.2 int Mkdir(const char \* dir)

建立名称为dir的文件夹。

4.1.3 int RMdir(const char \* dir)

删除名称为dir的文件夹，若文件夹不存在，不做任何操作。

4.1.4 int GetStDay(char \* stDay)

得到自定义字符串格式的日期信息。

4.1.5 int GetStNow(char \* stNow)

得到自定义字符串格式的时间信息。

4.1.6 int GetDiskFreeByte(\_\_int64 \* bytes,int \* MByte,const char \* dirMain)

得到当前磁盘的剩余空间，单位为字节数。

4.1.7 int FindOldestDIR(const char \* dir,char \* dirOldest)

查询并返回最早建立的序列文件夹，为循环记录“删旧存新”做准备。

## 4.2 与日志操作有关的功能模块

4.2.1 int LBuildDirLog(char \* dirLog,const char \* dirMain)

得到日志文件的文件夹名称。

4.2.2 int LBuildFnLog(char \* fnLog, const char \* dirLog)

建立日志文件。

4.2.3 int LNoteFirst(const char \* fnLog)

写入日志的首行文字，即统一的标题和分割符文字。

4.2.4 int LWRStr(const char \* str, int len, const char \* fn)

向日志文件中写入特定长度的字符串。

4.2.5 int LWRLine(const char \* line, int len,const char \* fn)

向日志文件中写入一行字符。

4.2.6 int LNoteRMdir(const char \* dir,const char \* fn)

在日志文件中记录删除的序列文件夹。

4.2.7 int LNoteGolbal(const char \*dirMain,const char \*dirData,const char \*fnData,const char \*fnLog,long fIndex)

在日志文件中记录主工作路径、序列文件夹路径、序列文件名、文件序列号等信息。

## 4.3 与序列文件相关的功能模块

4.3.1 int A2Init(int \* width,int \* height,int \*bitpp)

序列文件初始化函数，同时获取显示设备的像素长度、像素宽度和颜色深度。

4.3.2 int A2WFileHead(const char \* fnData,int pics,int width,int height)

序列文件头写入函数，将序列文件的头部信息写入指定的文件中。

4.3.3 int A2Cap(char \* srcBMP,int width,int height,int bitpp)

获取显示设备的图像像素，将其存在在一个字符数组中。

4.3.4 int A2Decode(char \* dstBMP,int \* dstLen,const char \*srcBMP,int width,int height)

序列数组压缩函数，将序列数组进行有损压缩。

4.3.5 int A2Check()

检查序列数组的正确性。

4.3.6 int A2WFrameContent(const char \*decBMP,int decLen,const char \* fnData)

将压缩后的序列数组写入序列文件。

4.3.7 int A2WFrameStamp(const char \* fnData)

在序列文件中写入保护性标签。