**Installation Guide**

Contents

[Introduction 1](#_Toc429254994)

[ System structure 1](#_Toc429254995)

[Installation 2](#_Toc429254996)

[ Installation package 2](#_Toc429254997)

[ Chinese type tool(AVF) 3](#_Toc429254998)

[ Equipment configuration 4](#_Toc429254999)

[ Application 4](#_Toc429255000)

[ conf 5](#_Toc429255001)

[ log 5](#_Toc429255002)

[ mock-data 6](#_Toc429255003)

[ test 6](#_Toc429255004)

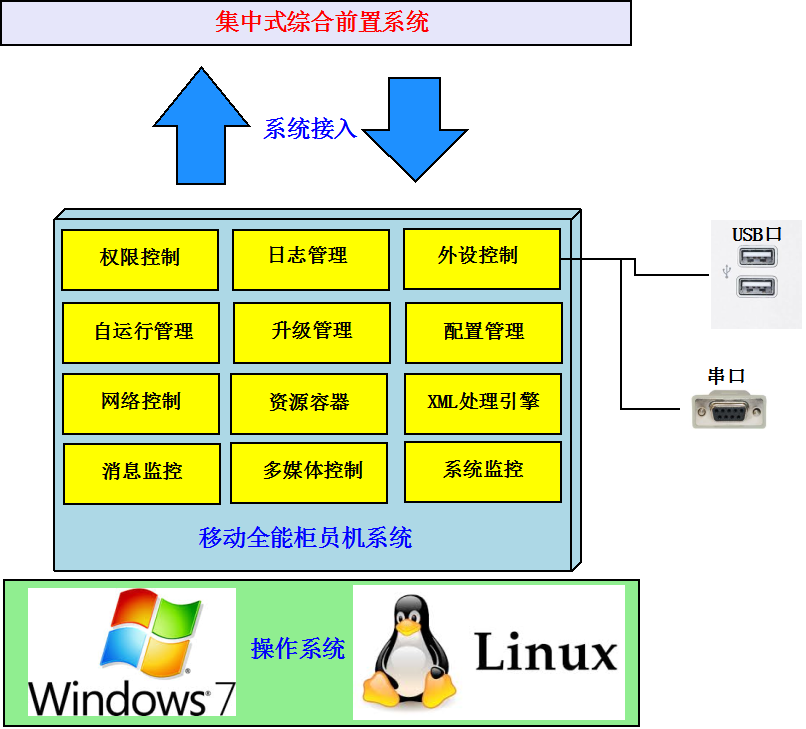
[ Desktop shortcut 6](#_Toc429255005)

## Introduction

The client system is built on java technology, so it intrinsically has a capability of running different platform. However, current smart equipment usually uses window7 as target platform, we suggest installing client system in win7 platform and require smart equipment must support this type of operation system. But if the customer wants to use diverse operation system such as Linux or Mac OS, we suggest that they should offer various drivers in those platform for smart equipment.

### System structure

The architecture picture shows the comprehensive construction of application based on xml-UI framework. Light blue box in picture is smart equipment, the whole client system is running in it.

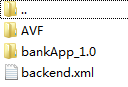


## Installation

In this paragraph, we will detail client installation. For a convenience, we presume the target operation system running in a smart equipment is window7. In the next parts, if we have no special designation, we all use window7 as default target operation system.

### Installation package

Installation package contains a lot of folders, which are for different file types. The basic package structure is described by the picture below.



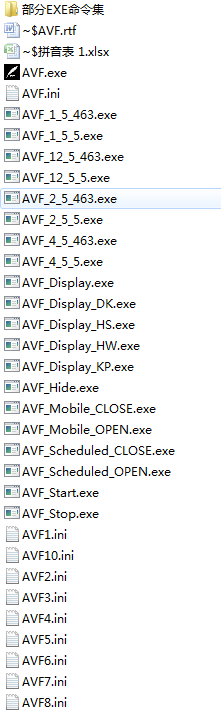
We interpret the installation package structure via following describes,

* Installation folder 🡪 such as ‘D:\bank\’
  + AVF 🡪 this folder includes Chinese type tool
  + bankApp\_1.0 🡪application package
  + backend.xml 🡪equipment configuration

### Chinese type tool(AVF)

This is a Chinese type tool, a lot of bank’s smart equipments have integrated this kind of type tool. User can utilize this to input account name that must be Chinese character.

AVF package is like picture below:



### Equipment configuration

This is configuration file of smart equipment, it includes many local device items used by application.

Sample of equipment configuration:

**<?xml version="1.0" encoding="UTF-8"?>**

**<Transportation**

**xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'**

**xsi:noNamespaceSchemaLocation='TransportationSchema.xsd'>**

**<server>**

**<host>127.0.0.1</host> <!-- IP address of aggregation server -->**

**<port>8080</port>** **<!-- port of aggregation server -->**

**</server>**

**<registration>**

**<prsCode>sanlogin</prsCode> <!-- registration code to aggregation server -->**

**<terminalID>00000001</terminalID> <!—equipment’s ID -->**

**<token>ABCD</token> <!-- equipment’s token -->**

**<heartbeat>100000</heartbeat> <!-- heartbeat frequency between equipment and aggregation server -->**

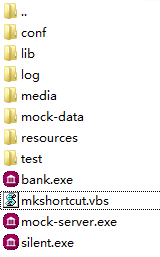
**<branchno>600000</branchno> <!—equipment’s branch ID -->**

**</registration>**

**</Transportation>**

### Application

This is application software package. It includes all software functionalities, just like picture below:

****

We interpret the application package structure via following describes,

* Application folder 🡪 such as ‘D:\bank\app’
  + conf 🡪 this folder includes GUI schema
  + lib 🡪application running library
  + log 🡪application running log files
  + media 🡪wav files used by application
  + mock-data 🡪mock data of mock server
  + resource 🡪application icon files
  + test 🡪application automatic test suite
  + bank.exe 🡪application starter
  + mkshortcut.vbs 🡪shortcut creation
  + mock-server.exe 🡪mock server starter
  + silent.exe 🡪application automatic testing

#### conf

About this part, you can refer to ‘How to config UI.docx‘

#### log

About this part, you can look into product folder. Basic log record likes sample below,

**七月 14, 2015 11:07:50 下午 king.flow.net.TunnelBuilder$HeartBeatTask run**

**信息: Heartbeat message : <?xml version="1.0" encoding="UTF-8" standalone="yes"?><TLS><counter>1</counter><prscode>sanlogin</prscode><terminalid>00000001</terminalid><token>ABCD</token><branchno>600000</branchno><terminalstate>0</terminalstate><startid>1436886470677</startid><keyboardstate>0</keyboardstate><prtstate>1</prtstate><version>1.0</version></TLS>**

**七月 14, 2015 11:07:51 下午 king.flow.net.P2PTunnel$MessageClientHandler channelRead**

**信息: <?xml version="1.0" encoding="UTF-8" standalone="yes"?><TLS><retcode>0</retcode><terminalid>00000001</terminalid><okmsg>终端注册成功</okmsg><errmsg></errmsg><restart>0</restart><changekey>0</changekey></TLS>**

#### mock-data

out this part, you can refer to ‘How to

#### test

out this part, you can refer to ‘How to

### Desktop shortcut

General message is plain-vanilla conversation carrier in this system. All conducts in current system belong to this category, such as query, transfer cash and so forth. These message will catch component’s value and tweak them into an xml string. With a cmd code appending, we will send this string to aggregation server.

Sample general message sent from client:

**<?xml version="1.0" encoding="UTF-8" standalone="yes"?>**

**<TLS>**

**<counter>3</counter>**

**<uid>20150831170718866243</uid> <!—message unique serial number-->**

**<prscode>queryAcount</prscode>**

**<terminalid>00000001</terminalid>**

**<branchno>test\_bank\_division</branchno>**

**<N\_330332>2015-08-31</N\_330332> <!-- component tag and value, tag consists of ‘N\_’+ component id -->**

**<N\_330334>2015-08-31</N\_330334>**

**</TLS>**

Sample general message come from aggregation server:

**<?xml version="1.0" encoding="UTF-8" standalone="yes"?>**

**<TLS>**

**<retcode>1</retcode> <!—the signal of conduct -->**

**<terminalid>00000001</terminalid>**

**<okmsg/> <!—the successful result of conduct -->**

**<errmsg>输入的密码错误,操作无法进行</errmsg> <!—the cause of conduct failure -->**

**<prtmsg/>**

**<cargo/>**

**</TLS>**