ICS 35.020
M37

团　　体　　标　　准

T/ZSA 52-2018

企业移动终端管理 API 规范

Enterprise Mobile Terminal Management API Specification

2018-12-29 发布　　2019-3-1 实施

中关村标准化协会　　发布

目　　次

前　言

本《规范》是Android系统移动终端管理（MDM）使用的操作系统调用API标准，规范了移动智能终端管理所需的移动终端管理软件调用接口(EMCG-MTM-API)。可解决终端厂商MDM API不统一，企业MDM系统难以管控多种移动设备问题。

《规范》适用于移动智能终端制造厂商、企业移动管理系统（EMMS）提供者、移动终端操作系统提供商以及本规范评估机构等。

本《规范》具有较强实用性，运用“实名计算”理论，以企业移动信息化产业市场趋势为导向，瞄准Android系统软件控制力弱的痛点，改变Android系统应用软件开发、销售各自为政，缺乏一致有效的管控机制，应用软件安全威胁不断增长的现状。

请注意本文件的某些内容可能涉及专利。中关村标准化协会不承担识别这些专利的责任。

本标准由中关村标准化协会技术委员会提出并归口。

参与本《规范》起草制定的单位：

中关村网络安全与信息化产业联盟

华为技术有限公司

联想集团

联信摩贝软件(北京)有限公司

奇虎360科技有限公司

北京小米科技有限责任公司

中国信息通信研究院

北京元心科技有限公司

北京国信灵通网络科技有限公司

本《规范》主要起草人：

王　克　中关村网络安全与信息化产业联盟（简称联盟）企业移动计算工作组（EMCG）

易平平　华为技术有限公司

王　攀　联想集团

胡文专　中兴通讯股份有限公司

刘长山　中兴通讯股份有限公司

张建国　联信摩贝软件(北京)有限公司

马勺布　北京元心科技有限公司

李一鸣　奇虎360科技有限公司

乜聚虎　北京小米科技有限责任公司

周宏斌　北京元心科技有限公司

闵　栋　中国信息通信研究院

引　言

随着移动互联网及便携智能终端技术的迅猛发展，使用手机、平板电脑等个人智能移动设备处理公务（BYOD）将成为企业、政府新的工作模式，相关技术产品具有广阔的市场前景。目前市场上提供的移动终端，从设备管控到应用软件安全，都不能满足企业信息安全需要。为解决智能移动终端安全问题，国内外相关厂商已经着手研究各种方法及对策。

Android系统的开放性，支持了各厂家推出不同品牌、不同型号、不同Android版本的移动终端，由于缺少统一的软件接口，使得企业建立的MDM平台难以同时对多品牌、多型号、多版本移动终端进行管理。

为解决多品牌、多型号Android移动智能终端管理（MDM）兼容问题，中关村网络安全与信息化产业联盟企业移动计算工作组（简称EMCG）组织多家厂家研究制定移动终端管理软件调用接口规范。

本《规范》为MDM开发者提供统一终端资源管控软件调用接口，提高MDM系统兼容性，降低系统开发和推广成本。

企业移动终端管理 API 规范

1 范围

本文档规范了移动智能终端管理所需的移动终端管理软件调用接口(EMCG-MTM-API)。移动智能终端制造商依据本规范提供通用移动终端设备；企业移动管理系统（EMMS）软件编程人员参照本规范开发终端管理客户端（MTM-App），实现各种品牌、型号移动智能终端管理的兼容。

本规范适用于移动智能终端制造厂商、企业移动管理系统（EMMS）提供者、移动终端操作系统提供商以及本规范评估机构等。

2 规范性引用文件

（略）

3 术语和定义

3.1 企业移动管理（EMM）

是指通过云方式对企业员工使用的处理企业业务的移动终端实施管理，EMM包括移动设备管理（MDM）、应用管理(MAM)以及内容管理(MCM)等。

3.2 企业移动管理系统（EMMS）

由移动智能终端、数据传输通道及企业移动终端管理云服务端组成，完成EMM所需功能。

3.3 终端管理客户端（MTM-App）

安装在移动终端上的应用软件，一般由企业移动管理系统（EMMS）提供者开发，负责接受并执行企业移动终端管理云服务端发出的管控指令。

3.4 企业移动终端管理云服务端

由企业移动管理系统（EMMS）提供者开发，建立在对移动终端进行管理的企业内，或由EMMS提供者提供相关服务的信息系统平台，企业移动终端管理云服务端通过无线网络实施移动设备管理（MDM）、移动应用管理(MAM)及移动内容管理(MCM)等。

3.5 EMCG-MTM-API 调用授权机制

为保证移动终端的安全，确保本规范定义的高级API调用的合法性而设计的授权机制。（见7.授权机制）

3.6 App

英文Application缩写，特指移动智能终端应用程序。

3.7 EMCG 官方机构证书

用于EMCG-APP签名发布。（参见《企业移动智能终端应用开发、安装、运行管控机制（T/ZSA 3001.01-2016）》）

3.8 EMCG-App

经官方机构发布或其授权证书签名的App。（见《企业移动智能终端应用开发、安装、运行管控机制（T/ZSA 3001.01-2016）》）

3.9 官方机构（Official Authority）

由EMCG定义的社会组织，为EMCG标准实施提供服务保障。（见《企业移动智能终端应用开发、安装、运行管控机制（T/ZSA 3001.01-2016）》）

3.10 EMCG(Enterprise Mobile Computing Group)

中关村网络安全与信息化产业联盟下设的企业移动计算工作组。

4 缩略语

API：Application programming interface 应用程序调用接口

App：Application 应用程序

BYOD：Bring Your Own Device 自有设备办公

EMCG：Enterprise Mobile Computing Group 企业移动计算工作组

EMM：Enterprise Mobile Management企业移动化管理

GPS：Global Positioning System 全球定位系统

ICCID：Integrate Circuit Card Identity 集成电路卡识别码（固化在手机SIM卡中，IC卡的唯一识别号码）

IMEI：International Mobile Equipment Identity 移动设备国际身份码（国际移动设备标识）

MAC：Media Access Control（Medium Access Control） 媒体访问控制（物理地址、硬件地址）

MDM：Mobile Device Management 移动设备管理

MTM-API：Mobile Terminal Management-API移动终端管理接口

MTM-App：Mobile Terminal Management-App移动终端管理客户端

SIM：Subscriber Identity Module 用户身份模块

SSID：Service Set Identifier 服务集标识（无线网络名称）

WiFi：Wireless Fidelity 无线局域网

5 系统架构

满足本规范移动终端管理软件调用接口(EMCG-MTM-API)的移动智能终端，在企业移动管理系统（EMMS）管理下实现移动终端配置、终端状态监控、应用（App）管理等。

EMMS基本架构如图1所示，包括：移动智能终端、数据传输通道及企业移动终端管理云服务端。

移动智能终端可以是企业专用设备（只运行企业移动业务专用软件），或者是BYOD设备（混合安装企业移动业务专用软件及个人应用软件）。本规范所描述的移动智能终端应具有EMCG-MTM-API调用授权机制（见7.授权机制）。移动智能终端也可参考EMCG《企业移动终端安全等级产品规范（T/ZSA 3001.01-2016）》进行设计、实现。

作为操作系统的组成部分，移动终端管理软件调用接口(EMCG-MTM-API)接受移动终端管理客户端（MTM-App）发出的本规范定义的系统调用，实现移动智能终端的管控。

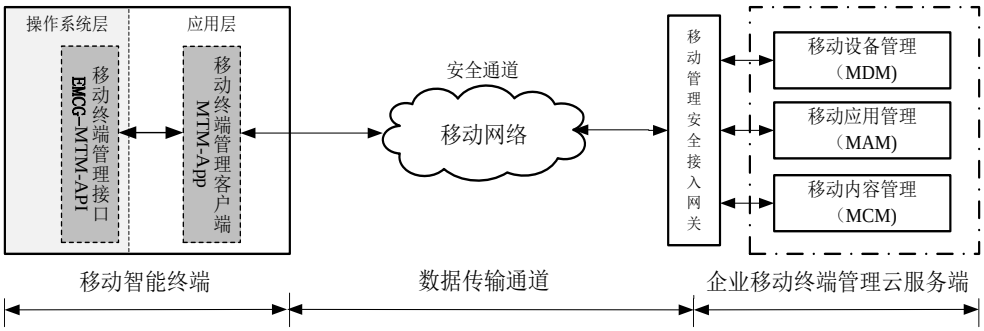


图1 企业移动管理系统（EMMS）框架

企业移动终端管理云服务端通过移动网络向移动终端MTM-App发出指令，实施企业业务处理所需要的移动设备管理、状态监视、安全策略管理及应用管理等。

EMMS使用的数据传输通道须采用必要的安全措施以保证数据传输的安全性。

6 EMCG-MTM-API 分类

6.1 基本API

本规范定义的基本API集（34个功能）可完成企业移动终端管理需要的设备相关信息获取等功能。一般移动终端管理应用软件不需授权即可调用基本API函数。基本API函数如表1。

表 1 EMCG-MTM-API 基本函数集

序号	功 能	描 述	函 数 名	可选函数	附录 A 页码
	网络配置和管理 API	指对 APN、VPN 等终端网络参数的配置管理，如账号、密码、SSID。			27
	蓝牙管理子类 API		子类 BluetoothManager		27
1	蓝牙状态		getState		28
2	蓝牙连接历史记录	可获得蓝牙配对设备信息，供设备安全管控场景使用。	getBondedDevices		28
3	蓝牙 MAC 地址信息获取	可获得蓝牙 MAC 地址设备信息，供设备安全管控场景使用。	getAddress		29
	WiFi 管理子类 API		子类 WifiManager		21
4	SSID 信息获取	可获得 SSID（无线网络名称）信息，供身份认证和设备标识。	getSSID		29
5	WiFi 状态		getWifiState		29
	系统功能控制 API	指对终端提供的系统功能进行控制。			30
	电话信息管理子类 API	SIM 卡电话号码、状态信息、ISO 信息、可用网络类型信息获取（双卡版本获取默认卡信息）。	子类 TelephonyManager		30
6	SIM 卡电话号码信息获取	可获得SIM卡电话号码信息，供身份认证和设备标识。 （备注：不是所有的手机号码都能获取。只是有一部分可以拿到，这个是由于移动运营商没有把手机号码的数据写入到SIM卡中。）	getLine1Number		30
7	SIM 卡可用网络类型信息获取	可获得SIM卡可用网络类型信息，供统计和审计使用。	getNetworkType		31

8	SIM 卡 ISO 信息获取	可获得 SIM 卡 ISO（SIM 卡运营商的国家代码）信息，供统计和审计使用。	getSimCountryIso		24
9	SIM 卡状态信息获取	可获得 SIM 卡状态信息，如无卡、未知状态、需要 pin 解锁、需要 puk 解锁、需要 networkpin 解锁和良好状态。	getSimState		24
10	SIM 卡运营商信息获取	可获得 SIM 卡的运营商名。	getSimOperatorName		25
11	网络运营商信息获取	可获得注册的移动网络运营商名。	getNetworkOperatorName		25
12	移动网络运营商的序列标识	获取当前注册的移动网络运营商的序列标识（MCC+MNC）。	getNetworkOperator		26
13	漫游状态获取	可获得漫游信息，判断是否发生漫游，供漫游控制场景使用。如发生漫游可设置启动/禁用推送服务、数据同步等策略。	isNetworkRoaming		26
	设备信息获取 API				26
14	IMEI（移动通讯设备身份标识）信息获取	IMEI 信息获取：可获得 IMEI（移动通讯设备身份标识）信息，供身份认证和设备标识。 注：无移动通讯模块的 PAD 设备无此标识。	getIMEI		26
15	ICCID（IC 卡的唯一识别号码）信息获取	ICCID 信息获取：可获得 ICCID（IC 卡的唯一识别号码）信息，供身份认证和设备标识。	getICCID		27
16	IMSI（SIM 卡的唯一用户识别号码）信息获取	IMSI 信息获取：可获得 IMSI（SIM 卡的唯一用户识别号码）信息，供身份认证和设备标识。	getIMSI		27
17	MAC 地址信息获取	MAC 地址信息获取：可获得 MAC 信息，供身份认证和设备标识。	getMacAddress		28
18	IP 地址信息获取	可获得移动设备的 IP 地	getIpAddress		28

		址信息，供统计和审计使用。			
19	获取设备唯一序列号 SN	获取设备唯一序列号。	getDeviceSerialNumber		28
20	获取 ROM 版本号信息	可获得移动设备的 ROM 版本号，供设备管理场景使用。	getRomVersion		29
21	CPU 使用率、个数、频率和型号信息获取	可获得 CPU 使用率等信息，供设备检查场景使用。	getCpuInfo		29
22	电量信息获取	可获得电量信息，供设备检查场景使用。	getBatteryLevel		29
23	内存容量和可用空间信息获取	获得 ROM 总容量和可用空间信息。	getMemoryInfo		30
24	内部存储器信息获取	返回设备内部存储空间大小，可用空间。	getInternalStorageInfo		30
25	SD 卡信息获取	返回设备外部存储设备的空间大小，可用空间。	getSDCardInfo		30
26	OS 版本号	可获得移动设备的系统信息，供设备管理场景使用。	getOSVersion		31
27	设备型号	可获得设备型号，供设备管理场景使用。	getDeviceModel		31
	GPS 信息获取 API				31
28	GPS 当前位置信息获取	获取当前设备的经纬度信息。	getCurrentLocation		31
29	GPS 最后位置信息获取	获得用户位置信息如获取经纬度、时间。	getLastKnownLocation		32
	外设信息获取 API				32
30	摄像头个数信息获取	可获得摄像头个数，供设备管理场景使用。	getNumberOfCameras		32
	应用软件信息获取 API				32
31	应用流量统计	可获得移动设备应用流量统计信息，供设备应用管理场景使用。	getAppDataTrafficUsage		32
32	设备当前安装的软件列表	获取设备当前安装的软件包名列表，如 com.tencent.qq、com.baidu.map 等。	getInstalledAppList		33
33	获取指定包名的	获取指定包名的已安装	getAppInfo		33

	已安装 App 详细信息	App 详细信息，包括：程序名、版本号、大小、使用时长、启动次数，供设备应用管理场景使用。			
34	**高级 API 调用授权**	终端授权 MTM-App 调用高级 API。	Authenticate		34

基本API函数详细定义见附录A、附录C 。

6.2 高级API

高级API集（105个功能）定义了“6.1基本API”以外的API集，对企业移动终端实施全面、深度的管控。调用高级API的MTM-App必须是EMCG签名的App，且须经授权机制（见7.授权机制）授权后方可调用高级API函数。高级API函数如表2。

表2 EMCG-MTM-API高级函数集

序号	功 能	描 述	函 数 名	可选函数	附录 B 页码
	网络配置和管理 API	指对 APN、VPN 等终端网络参数的配置管理，如账号、密码、SSID。			37
	APN 管理子类 API		子类 MtmAPNManager		37
35	查询指定 APN		queryApn		37
36	添加 APN		addApn		46
37	删除指定 APN		deleteApn		38
38	更新 APN		updateApn		39
39	设置默认 APN		setPreferApn		39
	VPN 管理 API	添加、删除和更新 VPN，连接和断开 VPN；可自动配置 VPN 参数，供移动设备安全管控场景使用。			39
40	查询 VPN		queryVpn		40
41	添加 VPN		addVpn		40
42	删除指定 VPN		deleteVpn		40
43	更新指定 VPN 配置		updateVpn		41
44	连接指定 VPN		connectVpn		41
45	断开 VPN		disConnectVpn		42
	NFC 控制子类 API		子类 MtmNFSManager		42

46	禁止和允许 NFC		setEnableNFC		42
	蓝牙管理子类 API		子类 MtmBluetoothAdapter		43
47	只允许和指定蓝牙设备类型通信	蓝牙设备类型白名单管理，只允许和指定蓝牙设备类型通信，如与其他蓝牙设备、音频/视频设备、PC 的连接控制。	setWhiteBluetoothType		43
48	禁止和允许蓝牙功能	可开启和关闭蓝牙。	setEnableBluetooth		67
49	禁止和允许蓝牙共享功能	指允许和禁止启用蓝牙共享。	setEnableBluetoothShare		44
	WiFi 管理子类 API		子类 WifiManager		44
50	WiFi 连接历史记录		getConfiguredNetworks		44
51	禁止和允许 WiFi	可开启和关闭 WiFi。	setEnableWifi		69
52	禁止和允许 WiFi 共享	指允许和禁止打开 WiFi 共享。	setEnableWifiShare		45
53	增加 WiFi 网络		addNetwork		45
54	删除 WiFi 网络		removeNetwork		46
55	修改 WiFi 网络		updateNetwork		46
56	设置优先连接的 WiFi 网络		enableNetwork		47
57	禁用指定的 WiFi 网络		disableNetwork		47
58	断开 WiFi 连接		disconnect		47
59	重新建立 WiFi 连接		reconnect		48
	系统功能控制 API	指对终端提供的系统功能进行控制。			48
	系统设置 API				48
60	禁止和允许 home 键	是否允许使用 Home 键，如移动营销，移动课堂等	setEnableHomeKey	*	48

		场景不允许切换要求。（Android 5.0 支持应用锁定）			
61	禁止和允许返回键	是否允许使用返回键，如移动营销，移动课堂等场景不允许切换要求。（Android 5.0 支持应用锁定）	setEnableBackKey	*	77
62	通过包名指定哪些 APP 为单应用模式	仅允许使用指定的应用并不允许退出当前应用界面，可通过 Home 键、返回键控制或者应用锁定等接口封装实现单应用模式。	setKioskApp		49
63	设置系统时间	（CDMA 终端默认网络同步，不在此列）	setDataTime		50
64	设置默认 launcher		setDefaultLauncher	*	50
65	隐藏任务栏		hideNavigationBar	*	51
	通信管理 API	SIM 卡电话号码、状态信息、ISO 信息、可用网络类型信息获取（双卡版本获取默认卡信息）。			51
66	禁止和允许收发短信		setEnableSMS		51
67	关闭和打开移动通信天线	关闭和打开移动通信天线功能, 关闭天线后，不能打电话和发短信。	setEnableRadio		81
68	短信删除	删除全部联系人(电话号码)或者内容关键字的短信,包括收件箱和发件箱。	deleteSMS		52
69	删除联系人	通过匹配姓名或电话号码，删除全部联系人。	deleteContact		53
70	通话记录删除	删除所有通话记录。	deleteCallHistory		53
71	挂电话	可远程挂断电话。	endCall		53

72	仅允许紧急呼叫控制		setEnableOnlyEmergencyCall		54
73	数据上网控制	禁止和允许移动数据上网功能。	setEnableMobileData		54
	设备控制 API				55
74	远程重启终端	可远程重启终端。	reboot		55
75	飞行模式开启		setAirplaneOn		88
76	飞行模式关闭		setAirplaneOff		56
77	关闭终端	可以远程关机。	shutDown		56
78	屏幕锁定	锁定屏幕，供设备安全管控场景使用。	lockScreen		56
79	屏幕解锁		unlockScreen		57
80	锁屏密码修改	远程密码修改。	setPassword		57
81	恢复出厂设置	可远程恢复出厂设置，供移动设备安全管控场景使用。	factoryReset		58
	数据擦除和备份 API				58
82	邮件删除	指终端预装的邮件客户端的删除。	uninstallEmail		58
83	应用临时数据删除	可远程删除应用临时数据。	wipeApp		58
84	设备擦除	设备数据擦除。	wipeData		59
85	视频删除	可删除视频，供企业数据安全管控场景使用。	removeVideo		59
86	图片删除	可删除图片，供企业数据安全管控场景使用。	removePicture		60
87	日历计划删除	可删除日历计划，供企业数据安全管控场景使用。	removeSchedule		60
	拷贝、粘贴控制 API				60
88	剪贴板控制	指允许和禁止启用剪贴板。	setEnableClipboard		60

89	截屏控制	指允许和禁止截屏。	setEnableCaptureScreen		61
	GPS 控制和管理子类 API		子类 MtmGPSManager		61
90	GPS 控制	指允许和禁止启用 GPS。	setEnableGps		61
	开放接口控制 API	指对终端对外开放供应用调用接口的使用限制。			62
91	防火墙接口（iptables）调用配置管理	允许或禁止配置开发防火墙接口，供应用调用。	setEnableIptables		62
	外设控制 API	指对终端外设进行控制。			62
	USB 管理子类 API		子类 MtmUSBManager		62
92	USB 调试模式控制	可开启和关闭 USB 调试。	setEnableUsbAdb		62
93	USB 控制	是否允许启用 USB，禁用 USB 后，只保留 USB 充电。	setEnableUsbFunction		63
94	USB-MTP 控制	指允许和禁止启用 USB 媒体流传输。	setEnableUsbMTP		63
95	USB-PTP 控制	指允许和禁止启用 USB 图片传输。	setEnableUsbPTP	*	64
96	USB 大容量存储控制	指允许和禁止使用 USB 大容量存储。	setEnableUsbMassStorage	*	64
97	USB 转网卡控制	指允许和禁止使用 USB 转网卡。	setEnableUsbRNDIS		65
98	USB 网络共享控制	可开启和关闭 USB 网络共享。	setEnableUsbTethering		65
	存储管理 API				65
99	存储访问控制(SD 卡或 TF 卡)	指允许和禁止访问 SD 卡存储。	setEnableSDCard		65
100	挂载 SD 卡	挂载 SD 卡（TF 卡），供设备安全管控场景使用，或支持系统和 App 访问 SDCard。	mountSDCard		66
101	卸载 SD 卡	卸载 SD 卡（TF 卡），禁止系统和 App 访问 SDCard。	umountSDCard		66

102	SD 卡擦除	可远程擦除内、外置 SD 卡数据，供企业数据安全管控场景使用。	eraseSDCard		67
	摄像头管理子类 API		子类 MtmCamaraManager		67
103	摄像头控制	可开启和关闭摄像头，供移动设备安全管控场景使用。	setEnableCamera		67
	录音控制子类 API		子类 MtmRecorderManager		68
104	麦克风控制	指允许和禁止启用麦克风进行录音等操作。	setEnableMicrophoneMute		68
105	录音控制	是否允许启用录音。	setEnableAudioRecorder		68
	应用软件管理和控制 API	指对终端上安装的应用软件进行管理和自启动、关闭等控制。			69
	应用配置 API	指对邮件等终端自带应用参数的配置。			69
106	配置添加邮件	向终端系统添加邮件配置，支持 Exchange、POP3、IMAP 等帐号。	addMailConfig		69
107	删除邮件配置	删除终端系统邮件配置信息。	deleteMailConfig		69
108	更新邮件配置	向终端系统更新覆盖邮件配置，支持 Exchange、POP3、IMAP 等帐号。	updateMailConfig		70
109	获取设备当前运行中的软件列表	可查看到当前正在运行的应用。	getRunningAppList		70
110	软件安装的历史记录		getAppInstallHistory		70
111	启动指定应用进程	远程启动某应用进程。	startApp		71
112	停止指定应用进程	远程禁用某应用进程。	endApp		71
113	新增或删除开机自启动控制	通过包名配置，允许指定 apk 开机自启动，可设置允许指定应用开机自启动。	setEnableAppAutoRun		72
	原生浏览器控制子类 API		子类 MtmBrowserManage		72

			r		
114	浏览器访问控制	可禁止和允许用户使用原生浏览器。	setEnableNativeBrowser		72
115	浏览器 cookie 控制		setEnableBrowserCookie		73
116	浏览器 javascript 控制		setEnableBrowserJavascript		73
117	浏览器弹出窗口控制		setEnableBrowserPopWindow		73
118	浏览器自动填充控制		setEnableBrowserAutoFill		74
119	浏览器 https 安全警告控制		setEnableBrowserSafetyWarning		74
120	URL 访问控制	通过黑白名单限制可访问的网站。	setBrowserUrlBlockingMode		75
121	添加浏览器书签配置	可远程添加自定义书签。	addBrowserBookmark		75
	应用安装、运行和删除子类 API		子类 MtmAppManager		76
122	静默安装应用	强制并静默安装指定路径 App。	forceInstallApp		76
123	静默卸载应用	强制并静默卸载指定包名 App。	forceUninstallApp		76
124	禁止或允许用户安装 App	禁止存储在 SD/TF 卡里面的 apk 安装；禁止通过 adb 命令安装 apk。	setEnableAppInstallation		77
125	禁止或允许用户卸载 App	禁止在 settings-->Apps 中卸载 apk；禁止通过 adb 命令卸载 apk。	setEnableAppUninstallation		77
126	禁止或允许用户强行关闭 App	指禁止强行关闭指定程序或服务。	setEnableKillApp		78
127	禁止或允许 adb 控制	可禁止或启用 adb install 的 APK 安装方式。	setEnableADB		78
128	禁止或允许指定 App 运行	禁止用户运行指定应用。	setEnableExecuteApp		78
129	删除指定应用数据	远程删除应用数据。	eraseAppData		79
130	禁止或允许“最近应用”功能	是否允许使用最近应用键，可以清空最近应用，可检测到有最近应用后直接清空。	setEnableRecentApp		79

	应用黑白名单子类 API		子类 MtmAppFilterManager		80
131	设置应用运行管控模式	设置系统启动应用运行管控模式：黑名单、白名单及普通模式。	setAppRunningBlockMode		80
132	设置应用运行管控配置信息	设置应用运行管控配置信息，内容一般同时包含黑名单列表和白名单列表信息，通过使用 setAppRunningBlockMode 激活黑名单管控或白名单管控。	setAppRunningBlockConfig		80
133	设置应用安装管控模式	设置系统启动应用安装管控模式：黑名单、白名单及普通模式。	setAppInstallBlockMode		81
134	设置应用安装管控配置信息	设置应用安装管控配置信息，内容一般同时包含黑名单列表和白名单列表信息，通过使用 setAppRunningBlockMode 激活黑名单管控或白名单管控。	setAppInstallBlockConfig		82
135	强制指定应用运行	设置强制运行的指定应用列表信息，在该信息中包含的应用将被操作系统强制启动并保持运行。	forceAppKeepRunning		83
136	卸载黑名单的应用		uninstallAppInBlackList		84
137	设置禁止卸载的应用名单(必装应用)	必须强制安装该列表（应用卸载黑名单列表）中的应用且不允许卸载，安装通过包名匹配。	setForbitUninstallationList		84
138	设置系统可以访问的 URL 黑白名单模式	设置系统可以访问的 URL 管控模式：黑名单、白名单及普通模式。	setURLBlockingMode		85
139	设置系统可以访问的 URL 黑白名单配置信息	设置系统可以访问的 URL 黑白名单配置信息，内容一般同时包含黑名单列表和白名单列表信息，通过使用 setURLBlockingMode 激	setURLBlockingConfig	*	85

		活黑名单管控或白名单管控。			

高级API函数详细定义见附录B。

6.3 必备与可选API

在本标准规范的函数集中有些函数是企业管理常用的，有些是不常用的，对于不常用的函数本规范作为“可选”函数，在表1 EMCG-MTM-API基本函数集、表2 EMCG-MTM-API高级函数集中用星号（“*”）标注的为“可选”函数，未用星号标注的为执行本标准的“必备”函数。

7 授权机制

为保证移动终端的安全，确保本规范定义的高级API（3.4高级API）调用合法性，调用高级API的终端管理客户端（MTM-App）须经授权机制认证授权方可调用。

7.1 授权要素

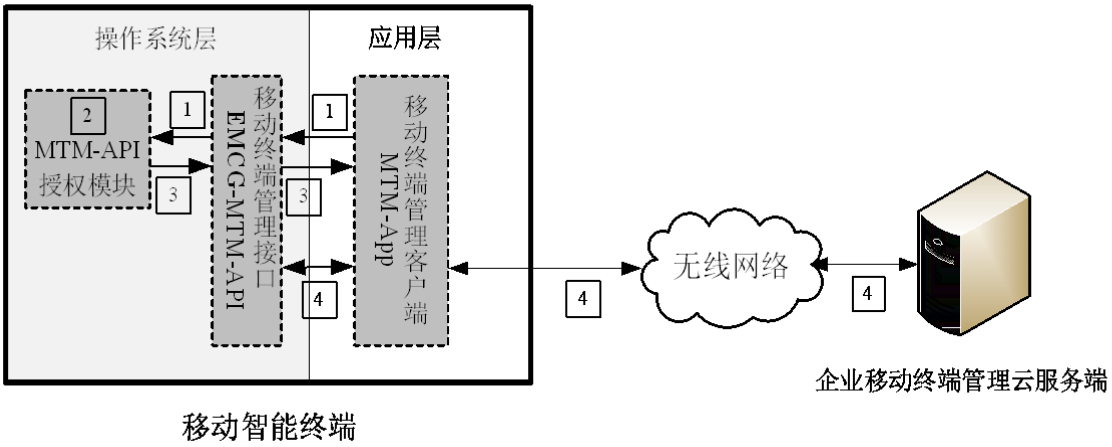
1）调用高级API的MTM-App是遵照《企业移动智能终端应用开发、安装、运行管控机制(T/ZSA 3001.01-2016）》标准开发的EMCG-App，并且MTM-App是采用EMCG企业移动化管理（EMM）属性证书进行签名的。

2）移动智能终端须具有验证EMCG-App合法性的功能，包括EMCG-App安装、运行签名验证机制。

3）移动智能终端须预置EMCG官方机构证书。

4）移动智能终端须预置MTM-App授权模块（授权模块可以是独立的App，也可是一个系统库函数）。

7.2 授权流程



1 **授权请求** **终端管理客户端（MTM-App）**在调用高级功能 API 之前调用本规范定义的**授权函数(**Authenticate)向终端系统发起授权请求。(系统将 MTM-App 包名和开发者签名证书传递给 **MTM-App 授权模块）**

2 **授权验证** **MTM-App 授权模块**对 **MTM-App** 签名进行验证。 验证 MTM-App 是否是用 **EMCG 证书**(“EMM”属性证书)签名的 EMCG-App。

3 **验证返回** 如果 **MTM-App** 是 EMCG-App，并且是用 EMM 属性证书签名的，则 **MTM-App 授权模块**返回授权许可，否则返回禁止授权。

4 **API 调用** MTM-App 获取到授权许可后，可接受企业终端管理云服务器发出的管理控制指令，调用高级功能 API；后续的调用无需再授权，直至 **MTM-App** 退出运行。

图 2 终端管理客户端（MTM-App）授权流程

附录 A

EMCG-MTM-API 基本函数集定义

(Android OS)

版本	作者	时间	备注
1.0.	王克、易平平、刘长山、宋卓、张建国	2015 年 3 月~4 月	初稿分工设计、讨论、编写、完善
1.1	张建国	2015 年 4 月 26 日	全面整合
1.2	王克	2015 年 5 月 7 日	根据易平平、张建国、刘长山、宋卓提供内容整理
1.3	刘长山、胡文专	2015 年 5 月 22 日	根据 5 月 8 日会后王克整理

A.1 概述

本附录是**移动终端管理软件调用接口(EMCG-MTM-API)**基本函数集的程序调用说明文档。移动智能终端制造商依据本附录进行产品开发，**企业移动管理系统（EMMS）提供者**依据本附录开发**终端管理客户端（MTM-App），**实现对所有符合 EMCG 标准的硬件终端设备的统一管理。

本文档定义的基本 API 集可完成企业移动终端管理需要的设备相关信息获取等功能。一般移动终端管理应用软件不需授权即可调用基本 API 集。

本文档版本包含 34 个 API 调用方法。

A.2 Android 调用规范

移动终端厂商依据 EMCG-MTM-API 规范开发并提供给**终端管理客户端**（MTM-App）调用。EMCG-MTM-API 以 Android JAR 包方式封装并部署在 Android 的 Framework 层。EMMS 开发 MTM-App 时需要在开发环境中包含 EMCG-MTM-API 的引用文件，名称为：EMCG_MDM_SDK.jar（注意大小写）。具体使用方法参见 **Error! Reference source not found.**。

EMCG-MTM-API 定义了 MTM 管理类，名称为：EMCG_MTM_Manager，开发调用时须首先进行类的实例化，然后即可调用各成员函数，如：

```
//实例化对象
EMCG_MTM_Manager    mtmManager = new EMCG_MTM_Manager();
…
//如果需要调用高级权限 API，App 须认证权限；如只需调用基本权限 API 则无需认证
//retVal = mdmManager.authenticate();
…
//调用 EMCG-MTM-API 中函数的具体方法，如：强制静默安装指定 apk 软件
int retVal = mtmManager.forceInstallApp("/SDCard/apks/qq.apk");
…
```

A.3 网络配置和管理 API

A.3.1 蓝牙管理子类 API

Class MtmBluetoothManager

类说明：

蓝牙管理子类，负责管理智能终端设备的各项蓝牙功能。MtmBluetoothManager 属于 EMCG_MTM_Manager 类的子类。

Class MtmBluetoothDevice

类说明：

蓝牙设备类，用于描述外部的蓝牙设备，包含蓝牙设备的属性、特征和状态，如名称、mac 地址和绑定状态等；

A.3.1.1 蓝牙状态

```
int getState()
```

函数说明：

获取蓝牙设置状态，属于蓝牙管理子类 MtmBluetoothManager 的方法。

参数说明：

- [输入]

  无。

- [返回值]

  - BLUETOOTH_STATE_OFF

    蓝牙设备处于已关闭状态；

  - BLUETOOTH_STATE_TURNING_ON

    蓝牙设备正在打开；

  - BLUETOOTH_STATE_ON

    蓝牙设备处于已开启状态；

  - BLUETOOTH_STATE_TURNING_OFF

    蓝牙设备正在关闭。

A.3.1.2 蓝牙历史配对记录

```
Set<MtmBluetoothDevice> getBondedDevices()
```

函数说明：

可获得蓝牙配对设备信息，供设备安全管控场景使用。

参数说明：

- [输入]

  无。

- [返回值] Set<MtmBluetoothDevice>

  已配对过的蓝牙设备信息，如发生错误，则返回为 null。

注意事项：

如果蓝牙状态不处于 BLUETOOTH_STATE_ON 状态，该 API 将返回 empty set。

A.3.1.3 蓝牙 MAC 地址信息获取

```
String getAddress()
```

函数说明：

可获得蓝牙 MAC 地址设备信息，供设备安全管控场景使用。

参数说明：

- [输入]

  无。

- [返回值] String

  蓝牙硬件 MAC 地址，如“00:11:22:AA:BB:CC”。

注意事项：

无。

A.3.2 WiFi 管理子类 API

Class MtmWifiManager

类说明：

WiFi 管理子类，负责管理智能终端设备的各项 WiFi 功能。MtmWifiManager 属于 EMCG_MTM_Manager 类的子类。

A.3.2.1 SSID 信息获取

String getSSID()

函数说明：

可获得当前 Wifi 连接的 SSID（无线网络名称）信息，供身份认证和设备标识。

参数说明：

- [输入]

  无。

- [返回值] String

  当前 WiFi 连接的 SSID（无线网络名称）信息。

注意事项：

仅当已建立 WiFi 连接或者正在建立 WiFi 连接时，才能获取到，否则返回为 null。

A.3.2.2 Wifi 状态

int getWifiState()

函数说明：

获取 WiFi 状态。

参数说明：

- [输入]

  无。

- [返回值]

  - WIFI_STATE_DISABLED

  WiFi 处于已关闭状态；

- WIFI_STATE_DISABLING

WiFi 正在关闭；

- WIFI_STATE_ENABLED

WiFi 处于已开启状态；

- WIFI_STATE_ENABLING

WiFi 正在开启；

- WIFI_STATE_UNKNOWN

WiFi 处于不可知的状态，一般在开启或者关闭过程中发生错误时才出现。

注意事项：

无。

A.4 系统功能控制 API

指对终端提供的系统功能进行控制。

A.4.1 电话信息管理子类 API

Class MtmTelephonyManager

类说明：

电话信息管理子类，负责管理智能终端设备的各项电话信息功能。MtmTelephonyManager 属于 EMCG_MTM_Manager 类的子类。

SIM 卡电话号码、状态信息、ISO 信息、可用网络类型信息获取（双卡版本获取默认卡信息）。

A.4.1.1 SIM 卡电话号码信息获取

String getLine1Number()

函数说明：

可获得 SIM 卡电话号码信息，供身份认证和设备标识。

（备注：手机号码不是所有的都能获取。只是有一部分可以拿到，这个是由于移动运营商没有把手机号码的数据写入到 SIM 卡中。）

参数说明：

- [输入]

  无。

- [返回值] String

  SIM 卡电话号码。

注意事项：

无。

A.4.1.2 SIM卡可用网络类型信息获取

```
int getNetworkType()
```

函数说明：

可获得SIM卡当前数据传输的网络类型信息，供统计和审计使用。

参数说明：

- [输入]

  无。

- [返回值]

  - NETWORK_TYPE_UNKNOWN

    当前数据网络类型未知；

  - NETWORK_TYPE_GPRS

    当前数据网络类型为GPRS；

  - NETWORK_TYPE_EDGE

    当前数据网络类型为EDGE；

  - NETWORK_TYPE_UMTS

    当前数据网络类型为UMTS；

  - NETWORK_TYPE_HSDPA

    当前数据网络类型为HSDPA；

  - NETWORK_TYPE_HSUPA

    当前数据网络类型为HSUPA；

  - NETWORK_TYPE_HSPA

    当前数据网络类型为HSPA；

  - NETWORK_TYPE_CDMA

    当前数据网络类型为CDMA：Either IS95A or IS95B；

  - NETWORK_TYPE_EVDO_0

    当前数据网络类型为EVDO revision 0；

  - NETWORK_TYPE_EVDO_A

    当前数据网络类型为EVDO revision A；

  - NETWORK_TYPE_EVDO_B

    当前数据网络类型为EVDO revision B；

  - NETWORK_TYPE_1xRTT

    当前数据网络类型为1xRTT；

  - NETWORK_TYPE_IDEN

    当前数据网络类型为iDen；

  - NETWORK_TYPE_LTE

当前数据网络类型为 LTE；

- NETWORK_TYPE_EHRPD

当前数据网络类型为 eHRPD；

- NETWORK_TYPE_HSPAP

当前数据网络类型为 HSPA+。

注意事项：

无。

A.4.1.3 SIM 卡 ISO 信息获取

```
String getSimCountryIso()
```

函数说明：

可获得 SIM 卡 ISO（SIM 卡运营商的国家代码）信息，供统计和审计使用。

参数说明：

- [输入]

  无。

- [返回值] String

  SIM 卡 ISO（SIM 卡运营商的国家代码）信息。

注意事项：

无。

A.4.1.4 SIM 卡状态信息获取

```
int getSimState()
```

函数说明：

可获得 SIM 卡状态信息，如无卡、未知状态、需要 pin 解锁、需要 puk 解锁、需要 networkpin 解锁和良好状态。

参数说明：

- [输入]

  无。

- [返回值]

  - SIM_STATE_UNKNOWN

    SIM 卡状态未知;

  - SIM_STATE_ABSENT

    SIM 卡缺失状态，没有可用的 SIM 卡在终端设备中;

  - SIM_STATE_PIN_REQUIRED

SIM 卡处于 PIN 码锁定状态，需要用户输入 SIM 卡 PIN 码解锁;

- SIM_STATE_PUK_REQUIRED

SIM 卡处于 PUK 码锁定状态，需要用户输入 SIM 卡 PUK 码解锁;

- SIM_STATE_NETWORK_LOCKED

SIM 卡处于网络锁定状态，需要网络 PIN 码解锁;

- SIM_STATE_READY

SIM 卡处于 Ready 状态;

- SIM_STATE_NOT_READY

SIM 卡错误;

- SIM_STATE_PERM_DISABLED

SIM 卡错误;

- SIM_STATE_CARD_IO_ERROR

SIM 卡错误。

注意事项：

无。

A.4.1.5 SIM 卡运营商信息获取

```
String getSimOperatorName()
```

函数说明：

可获得 SIM 卡的运营商名。

参数说明：

- [输入]

  无。

- [返回值] String

  SIM 卡的运营商名称。

注意事项：

SIM 卡必须处于 SIM_STATE_READY 状态。

A.4.1.6 网络运营商信息获取

```
String getNetworkOperatorName()
```

函数说明：

可获得注册的移动网络运营商名。

参数说明：

- [输入]

  无。

- [返回值] String

  当前注册的移动网络运营商名称。

注意事项：

仅当用户已经注册了移动网络才有效。

A.4.1.7 移动网络运营商的序列标识

```
String getNetworkOperator()
```

函数说明：

获取当前注册的移动网络运营商的序列标识（MCC+MNC）。

参数说明：

- [输入]

  无。

- [返回值] String

  当前注册的移动网络运营商的序列标识（MCC+MNC）。

注意事项：

仅当用户已经注册了移动网络才有效。

A.4.1.8 漫游状态获取

```
boolean isNetworkRoaming()
```

函数说明：

可获得漫游信息，判断是否发生漫游，供漫游控制场景使用。如发生漫游可设置启动/禁用推送服务、数据同步等策略。

参数说明：

- [输入]

  无。

- [返回值] boolean

  true：表示处于漫游状态；

  false：表示处于非漫游状态。

注意事项：

仅当用户已经注册了移动网络才有效。

A.4.2 设备信息获取 API

A.4.2.1 IMEI（移动通讯设备身份标识）信息获取

```
String getIMEI()
```

函数说明：

获取设备的移动通讯设备身份标识码，供身份认证和设备标识。

注：无移动通讯模块的 PAD 设备无此标识。

参数说明：

- [输入]

  无。

- [返回值] String

  DeviceId 信息。

注意事项：

A.4.2.2 ICCID（IC 卡的唯一识别号码）信息获取

```
String getICCID()
```

函数说明：

获取 ICCID(IC 卡的唯一识别号码)信息，供身份认证和设备标识。

参数说明：

- [输入]

  无。

- [返回值] String

  ICCID 信息。

注意事项：

无。

A.4.2.3 IMSI（SIM 卡的唯一用户识别号码）信息获取

```
String getIMSI()
```

函数说明：

获取设备的 IMSI 标识信息。

参数说明：

- [输入]

  无。

- [返回值] String

  包含 IMSI 信息。

注意事项：

无。

A.4.2.4 MAC 地址信息获取

```
String getMacAddress()
```

函数说明：

获取 MAC 地址信息，供身份认证和设备标识。

参数说明：

- [输入]

  无。

- [返回值] String

  MAC 地址信息。

注意事项：

无。

A.4.2.5 IP 地址信息获取

```
String getIpAddress()
```

函数说明：

获得移动设备的 IP 地址信息，供统计和审计使用。

参数说明：

- [输入]

  无。

- [返回值] String

  IP 地址信息。

注意事项：

无。

A.4.2.6 获取设备唯一序列号 SN

```
String getDeviceSerialNumber()
```

函数说明：

获取设备唯一序列号。

参数说明：

- [输入]

  无。

- [返回值] String

  DeviceSerialNumber 号信息。

注意事项：

无。

A.4.2.7 获取 ROM 版本号信息

```
String getRomVersion()
```

函数说明：

获得移动设备的 ROM 版本号，供设备管理场景使用。

参数说明：

- [输入]

  无。

- [返回值] String

  ROM 版本号信息。

注意事项：

无。

A.4.2.8 CPU 使用率、个数、频率和型号信息获取

```
String getCpuInfo()
```

函数说明：

获得 CPU 使用率等信息，供设备检查场景使用。

参数说明：

- [输入]

  无。

- [返回值] String

  CPU 使用率、个数、频率和型号信息。

注意事项：

无。

A.4.2.9 电量信息获取

```
String getBatteryLevel()
```

函数说明：

获得电量信息，供设备检查场景使用。

参数说明：

- [输入]

  无。

- [返回值] String

  电量信息。

注意事项：

无。

A.4.2.10 内存容量和可用空间信息获取

```
String getMemoryInfo()
```

函数说明：

获得 ROM 总容量和可用空间信息，供设备检查场景使用。

参数说明：

- [输入]

  无。

- [返回值] String

  内存容量和可用空间信息。

注意事项：

无。

A.4.2.11 内部存储器信息获取

```
String getInternalStorageInfo()
```

函数说明：

返回设备内部存储空间大小，可用空间。

参数说明：

- [输入]

  无。

- [返回值] String

  返回设备的存储空间大小和设备上剩余的可用空间信息。

注意事项：

使用空间可以通过计算得知。

A.4.2.12 SD 卡信息获取

```
String getSDCardInfo()
```

函数说明：

返回设备外部存储设备的空间大小，可用空间。

参数说明：

- [输入]

  无。

- [返回值] String

  返回设备的存储空间大小和设备上剩余的可用空间信息。

注意事项：

使用空间可以通过计算得知。

A.4.2.13 OS 版本信息获取

```
String getOSVersion()
```

函数说明：

获得移动设备的系统信息，供设备管理场景使用。

参数说明：

- [输入]

  无。

- [返回值] String

  设备的系统信息。

注意事项：

无。

A.4.2.14 设备型号

```
String getDeviceModel()
```

函数说明：

获得移动设备的 Model 属性信息，供设备安全管控场景使用。

参数说明：

- [输入]

  无。

- [返回值] String

  DeviceModel 属性信息。

注意事项：

无。

A.4.3 GPS 信息获取 API

A.4.3.1 GPS 当前位置信息获取

```
String getCurrentLocation()
```

函数说明：

获取当前设备的经纬度信息。

参数说明：

- [输入] String provider

  输入的 provider。

- [返回值] Location

  返回用户的位置信息，包含精度和纬度。

注意事项：

无。

A.4.3.2 GPS 最后位置信息获取

```
String getLastKnownLocation()
```

函数说明：

获得用户位置信息如获取经纬度、时间。

参数说明：

- [输入] String provider

  输入的 provider。

- [返回值] Location

  返回用户的位置信息，包含精度和纬度。

注意事项：

无。

A.5 外设信息获取 API

A.5.1 摄像头个数信息获取

```
int getNumberOfCameras()
```

函数说明：

获得摄像头个数，供设备管理场景使用。

参数说明：

- [输入]

  无。

- [返回值] int

  返回摄像头个数。

注意事项：

无。

A.6 应用软件信息获取 API

A.6.1 获取应用流量统计

```
String getAppDataTrafficUsage(String appID);
```

函数说明：

获取指定包名的 App 数据流量用量。

参数说明：

- [输入]String appID

  输入应用程序包名。

- [返回值] String

  返回该 App 网络用量信息，包含距离起始点的日期时间， 从起始点到现在消耗的网络流量等。详细格式参见附件。

注意事项：

无。

A.6.2 设备当前安装的软件列表

```
int getInstalledAppList(String []returnNames);
```

函数说明：

获取设备当前安装的软件包名列表，如 com.tencent.qq、com.baidu.map 等。

参数说明：

- [输出]String [] returnNames

  返回一个字符串数组，包含多个 App 包名字符串。

- [返回值]

  ➢ ERRORCODE_SUCCESS

  执行正确；

  ➢ 其他值

  错误，详见“错误码定义”章节。

注意事项：

无。

A.6.3 获取指定包名的已安装 App 详细信息

```
String getAppInfo(String appID);
```

函数说明：

获取指定包名的已安装 App 详细信息，包括：程序名、版本号、大小、使用时长、启动次数，供设备应用管理场景使用。

参数说明：

- [输入]String appID

  输入应用程序包名。

- [返回值] String

  返回指定包名的已安装 App 详细信息，XML 结构，详见附录格式。

注意事项：

无。

A.7 高级 API 调用授权

```
int authenticate()
```

函数说明：

MTM App 申请 EMCG 设备授权使用 MTM 高级权限。

参数说明：

- [输入]

  无。

- [返回值]

  - ERRORCODE_SUCCESS

    授权成功；

  - ERRORCODE_INVALIDSIGNATURE

    APP 签名证书无效。

注意事项：

无。

A.8 错误码定义

函数调用返回错误码定义表

错误码定义	值	含义
ERRCODE_SUCCESS	0	函数执行正确
ERRCODE_UNKNOWN	-1	未知异常错误
ERRCODE_EXECUTEERR	-2	执行失败
ERRCODE_PERMISSIONERR	-3	权限错误
ERRCODE_OBJECTNOTFOUNDERR	-4	未找到对象（如 SIM 卡、WIFI 设备、蓝牙等）

ERRORCODE_INVALIDSIGNATURE	-11	APP 签名证书无效

附录 B

EMCG-MTM-API 高级函数集定义

(Android OS)

版本	作者	时间	备注
1.0.	王克、易平平、刘长山、宋卓、张建国	2015 年 3 月~4 月	初稿分工设计、讨论、编写、完善
1.1	张建国	2015 年 4 月 26 日	全面整合
1.2	王克	2015 年 5 月 7 日	根据易平平、张建国、刘长山、宋卓提供内容整理
1.3	刘长山、胡文专、张建国	2015 年 6 月 14	5 月 8 日会后，王克整理
1.4	王克	2015 年 6 月 21 日	增加晋艳伟建议函数：4.3.7 锁屏密码修改 setPassword
1.5	王克	2015 年 6 月 23 日	根据长山、建国修改整理

B.1 概述

本附录是**移动终端管理软件调用接口(EMCG-MTM-API)**高级函数集的程序调用说明文档。移动智能终端制造商依据本附录进行产品开发，**企业移动管理系统（EMMS）提供者**依据本附录开发**终端管理客户端（MTM-App)**，实现对所有符合 EMCG 标准的硬件终端设备的统一、全面、深度的管理和控制。

调用高级 API 的 MTM-App 必须是 EMCG 签名的 App，且须经授权机制授权后方可调用高级 API。

本文档版本包含 105 个 API 调用方法。

B.2 Android 调用规范

EMCG-MTM-API 以 Android JAR 包方式封装并部署在 Android 的 Framework 层。EMMS 开发 MTM-App 时需要在开发环境中包含 EMCG-MTM-API 的引用文件，名称为：EMCG_MDM_SDK.jar（注意大小写）。具体使用方法参见 **Error! Reference source not found.**。

EMCG-MTM-API 定义了 MTM 管理类，名称为：EMCG_MTM_Manager，开发调用时需要首先进行类的实例化，然后即可调用各成员函数，如：

```
//实例化对象
EMCG_MTM_Manager    mtmManager = new EMCG_MTM_Manager();
…
//如果需要调用高级权限 API，App 须认证权限；
retVal = mdmManager.authenticate();
…
//调用 EMCG-MTM-API 函数的具体方法，如：强制静默安装指定 apk 软件
int retVal = mtmManager.forceInstallApp("/SDCard/apks/qq.apk");
…
```

B.3 网络配置和管理 API

B.3.1 APN 管理子类 API

```
Class MtmAPNManager
```

类说明：

APN 管理子类，负责管理智能终端设备的各项 APN 功能。MtmAPNManager 属于 EMCG_MTM_Manager 类的子类。

B.3.1.1 查询指定 APN

```
List<String> queryApn()
```

函数说明：

查询指定 APN。

参数说明：

- [输入] Map<String, String>apnparam

  指定查询 APN 所需匹配的关键参数，可一个或者多个关键参数，主要包括 name、apn、

mcc、mnc、numeric、user、password、server、proxy、port、mmsport、mmsproxy、mmsc、authtype、current、type。

- [返回值]
  - 返回所查询到 APN 的 id 列表

  执行正确；
  - null

  未查找到。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.3.1.2 添加 APN

```
String addApn()
```

函数说明：

添加 APN。

参数说明：

- [输入] Map<String, String>apnparam

  APN 配置参数，主要包括 name、apn、mcc、mnc、numeric、user、password、server、proxy、port、mmsport、mmsproxy、mmsc、authtype、current、type。
- [返回值]
  - id

  执行正确，返回所添加 APN 的 id 标识；
  - 其他值

  错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.3.1.3 删除指定 APN

```
int deleteApn()
```

函数说明：

删除指定 APN。

参数说明：

- [输入] String id

  所需删除 apn 的标识，该字段可以查询 apn 获取。
- [返回值]
  - ERRORCODE_SUCCESS

  执行正确；
  - 其他值

错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.3.1.4 更新 APN

```
int updateApn()
```

函数说明：

更新 APN。

参数说明：

- [输入] String id

  所需更新 apn 的标识，该字段可以查询 apn 获取；

- [输入] Map<String, String> apnparam

  所需更新 apn 的配置参数，主要包括 name、apn、mcc、mnc、numeric、user、password、server、proxy、port、mmsport、mmsproxy、mmsc、authtype、current、type。

- [返回值]

  ➢ ERRORCODE_SUCCESS

  执行正确；

  ➢ 其他值

  错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.3.1.5 设置默认 APN

```
int setPreferApn()
```

函数说明：

设置默认 APN。

参数说明：

- [输入] String id

  所需设置为默认 apn 的标识，该字段可以查询 apn 获取。

- [返回值]

  ➢ ERRORCODE_SUCCESS

  执行正确；

  ➢ 其他值

  错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.3.2 VPN 管理 API

添加、删除和更新 VPN，连接和断开 VPN；可自动配置 VPN 参数，供移动设备安全管控场景使用。

B.3.2.1 查询 VPN

List<String> queryVpn()

函数说明：

查询 VPN。

参数说明：

- [输入] Map<String, String>vpnparam

  指定查询 VPN 所需匹配的关键字，可一个或多个，详见 androd 源码 VpnProfile 类；如果 vpnparam 为 null，则返回所有 vpn 列表。

- [返回值]

  ➢ 返回所查询到 VPN 的 id 列表

  执行正确；

  ➢ null

  未查找到。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.3.2.2 添加 VPN

String addVpn()

函数说明：

添加 VPN，VPN 参数详见 androd 源码 VpnProfile 类。

参数说明：

- [输入] Map<String, String>vpnparam

  VPN 配置参数，VPN 参数详见 androd 源码 VpnProfile 类。

- [返回值]

  ➢ id

  执行正确，返回 VPN 的 id 标识；

  ➢ 其他值

  错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.3.2.3 删除指定 VPN

int deleteVpn()

函数说明：

删除指定 VPN。

参数说明：

- [输入] String id

  所需删除 vpn 的标识，该字段可以查询 vpn 获取。

- [返回值]

  ➢ ERRORCODE_SUCCESS

  执行正确；

  ➢ 其他值

  错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.3.2.4 更新指定 VPN 配置

```
int updateVpn()
```

函数说明：

更新指定 VPN 配置。

参数说明：

- [输入] String id

  所需更新 vpn 的标识，该字段可以查询 vpn 获取；

- [输入] Map<String, String> vpnparam

  更新 VPN 的配置参数，VPN 参数详见 androd 源码 VpnProfile 类。

- [返回值]

  ➢ ERRORCODE_SUCCESS

  执行正确。

  ➢ 其他值

  错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.3.2.5 连接指定 VPN

```
int connectVpn()
```

函数说明：

连接指定 VPN。

参数说明：

- [输入] String id

  所需连接 vpn 的标识，该字段可以查询 vpn 获取。

- [返回值]

  ➢ ERRORCODE_SUCCESS

  执行正确；

➢ 其他值

错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.3.2.6 断开 VPN

int disConnectVpn()

函数说明：

断开 APN。

参数说明：

- [输入]

  无。

- [返回值]

➢ ERRORCODE_SUCCESS

执行正确；

➢ 其他值

错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.3.3 NFC 管理子类 API

➢ Class MtmNFSManager

类说明：

NFS 管理子类，负责管理智能终端设备的各项 NFS 功能。MtmNFSManager 属于 EMCG_MTM_Manager 类的子类。

B.3.3.1 禁止和允许 NFC

int setEnableNFC()

函数说明：

禁止和允许 NFC 近距离传输数据。

参数说明：

- [输入]boolean enable

  true：允许；

  false：禁止。

- [返回值]

➢ ERRORCODE_SUCCESS

执行正确；

➢其他值

错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B. 3. 4 蓝牙管理子类 API

Class MtmBluetoothManager

类说明：

蓝牙管理子类，负责管理智能终端设备的各项蓝牙功能。MtmBluetoothManager 属于 EMCG_MTM_Manager 类的子类。

B. 3. 4. 1 只允许和指定蓝牙设备类型通信

int setWhiteBluetoothType()

函数说明：

只允许和指定蓝牙设备类型通信，如与其他蓝夜设备、音频/视频设备、PC 的连接控制。

参数说明：

- [输入] List<int>bluetoothclass

  指定允许通信的蓝牙设备类型列表，其类型取值详见 BluetoothClass.Device 内的常量。

- [返回值]

  ➢ERRORCODE_SUCCESS

  执行正确；

  ➢其他值

  错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B. 3. 4. 2 禁止和允许蓝牙功能

int setEnableBluetooth()

函数说明：

禁止和允许蓝牙功能。

参数说明：

- [输入]boolean enable
  true：允许；
  false：禁止。
- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；
  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B. 3. 4. 3　禁止和允许蓝牙共享功能

int setEnableBluetoothShare()

函数说明：

禁止和允许蓝牙共享功能。

参数说明：

- [输入]boolean enable

  true：允许；

  false：禁止。

- [返回值]

  ➢ ERRORCODE_SUCCESS

  执行正确；

  ➢ 其他值

  错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B. 3. 5　WiFi 管理子类 API

Class MtmWifiManager

类说明：

WiFi 管理子类，负责管理智能终端设备的各项 WiFi 功能。MtmWifiManager 属于 EMCG_MTM_Manager 类的子类。

Class MtmWifiConfiguration

类说明：

WiFi 网络配置信息类，用于描述 WiFi 网络的信息，如包含 SSID、认证协议、状态信息和 WiFi 网络在操作系统中的唯一 Id 标识（netId）等。

B. 3. 5. 1　WiFi 连接历史记录

List<MtmWifiConfiguration> getConfiguredNetworks()

函数说明：

获取 WiFi 连接历史记录。

参数说明：

- [输入]

  无。

- [返回值]List<MtmWifiConfiguration>

  WiFi 连接历史记录；WiFi 关闭或者失败时，返回 null。

注意事项：

无。

B.3.5.2 禁止和允许 Wifi

```
int setEnableWifi()
```

函数说明：

禁止和允许 WIFI 功能。

参数说明：

- [输入]boolean enable

  true：允许；

  false：禁止。

- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；

  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.3.5.3 禁止和允许 WiFi 共享

```
int setEnableWifiShare()
```

函数说明：

禁止和允许 WIFI 共享功能。

参数说明：

- [输入]boolean enable

  true：允许；

  false：禁止。

- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；

  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.3.5.4 增加 WiFi 网络

```
int addNetwork()
```

函数说明：

添加一条 WiFi 网络。

参数说明：

- [输入]MtmWifiConfiguration config

WiFi 网络的配置参数。

- [返回值]
  - netId

    执行正确，返回 WiFi 网络配置 id；

  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.3.5.5 删除 WiFi 网络

```
int removeNetwork()
```

函数说明：

删除指定的 WiFi 网络。

参数说明：

- [输入]int netId

  所需删除 WiFi 网络配置的 id。netId 是 WiFi 网络在操作系统中的唯一标识，属于 MtmWifiConfiguration 中的一个字段。添加和更新 WiFi 网络时，都会返回 netid，另外也可以通过 getConfiguredNetworks 获取所有的已配置过的 WiFi 网络列表，在 WiFi 网络配置类 MtmWifiConfiguration 中获取。

- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；

  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.3.5.6 修改 WiFi 网络

```
int updateNetwork()
```

函数说明：

更新指定的 WiFi 网络。

参数说明：

- [输入]MtmWifiConfiguration config

  所需更新的 WiFi 网络的配置参数。

- [返回值]
  - netId

    执行正确，WiFi 网络配置 id；

  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.3.5.7 设置优先连接的 WiFi 网络

```
int enableNetwork()
```

函数说明：

设置优先连接的 WiFi 网络。

参数说明：

- [输入]int netId

  指定优先连接的 WiFi 网络标识。

- [输入]boolean disableOthers

  是否禁用其它 WiFi 网络：
  true 表示禁用其它 WiFi 网络，只与指定优先连接的 WiFi 网络建立连接；
  false 表示不禁用其它网络，未与优先连接的 WiFi 网络建立连接的情况下，可与其它 WiFi 建立连接。

- [返回值]

  ➢ ERRORCODE_SUCCESS

  执行正确；

  ➢ 其他值

  错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.3.5.8 禁用指定的 WiFi 网络

```
int disableNetwork()
```

函数说明：

禁用指定的 WiFi 网络，在 Wifi 连接的时候，该 WiFi 网络不在连接的选择以内。

参数说明：

- [输入]int netId

  所需禁用 WiFi 网络的标识。

- [返回值]

  ➢ ERRORCODE_SUCCESS

  执行正确；

  ➢ 其他值

  错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.3.5.9 断开 WiFi 连接

int disconnect()

函数说明：

断开 WiFi 连接。

参数说明：

- [输入]

  无。

- [返回值]

  ➢ ERRORCODE_SUCCESS

  执行正确；

  ➢ 其他值

  错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.3.5.10 重新建立 WiFi 连接

int reconnect()

函数说明：

重新与当前活跃的接入点(AP)建立 WiFi 连接。

参数说明：

- [输入]

  无。

- [返回值]

  ➢ ERRORCODE_SUCCESS

  执行正确；

  ➢ 其他值

  错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.4 系统功能控制 API

B.4.1 系统设置 API

B.4.1.1 禁止和允许 home 键

int setEnableHomeKey()

函数说明：

禁止和允许 home 键功能。如移动营销，移动课堂等场景不允许切换要求。（Android 5.0 支持应用锁定）

参数说明：

- [输入]boolean enable

true：允许；

false：禁止。

- [返回值]
  - ERRORCODE_SUCCESS

  执行正确；

  - 其他值

  错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.4.1.2 禁止和允许返回键

```
int setEnableBackKey()
```

函数说明：

禁止和允许返回键功能。如移动营销，移动课堂等场景不允许切换要求。（Android 5.0 支持应用锁定）

参数说明：

- [输入]boolean enable

  true：允许；

  false：禁止。

- [返回值]
  - ERRORCODE_SUCCESS

  执行正确；

  - 其他值

  错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.4.1.3 通过包名指定哪些 APP 为单应用模式

```
int setKioskApp()
```

函数说明：

仅允许使用指定的应用并不允许退出当前应用界面，可通过 Home 键、返回键控制或者应用锁定等接口封装实现单应用模式。

参数说明：

- [输入] String pkgName

  指定单应用模式的 APP 列表。

- [返回值]
  - ERRORCODE_SUCCESS

  执行正确；

➢其他值

错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.4.1.4 设置系统时间

```
int setDataTime()
```

函数说明：

设置系统时间（CDMA 终端默认网络同步，不在此列）。

参数说明：

- [输入] String year

  年

- [输入] String month

  月

- [输入] String date

  日

- [输入] String hour

  时

- [输入] String min

  分

- [输入] String sec

  秒

- [返回值]

➢ERRORCODE_SUCCESS

执行正确；

➢其他值

错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.4.1.5 设置默认 launcher

```
int setDefaultLauncher()
```

函数说明：

设置默认 Launcher。

参数说明：

- [输入] String pkg

  Launcher App 的包名；

- [输入] String className

Launcher App 的 Launcher 界面 activity 对应的类名。

- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；

  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.4.1.6 隐藏任务栏

```
int hideNavigationBar()
```

函数说明：

隐藏和显示任务栏。

参数说明：

- [输入]boolean enable

  true：隐藏任务栏；

  false：显示任务栏。

- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；

  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.4.2 通信管理 API

B.4.2.1 禁止和允许收发短信

```
int setEnableSMS()
```

函数说明：

禁止和允许收发短信功能。

参数说明：

- [输入]boolean enable

  true：允许；

  false：禁止。

- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；

➢其他值

错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.4.2.2 关闭和打开移动通信天线

```
int setEnableRadio()
```

函数说明：

关闭和打开移动通信天线功能，关闭天线后，不能打电话和发短信。

参数说明：

- [输入]boolean enable

  true：打开；

  false：关闭。

- [返回值]

➢ERRORCODE_SUCCESS

执行正确；

➢其他值

错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.4.2.3 短信删除

```
int deleteSMS()
```

函数说明：

删除指定联系人(电话号码)或者内容关键字的短信，包括收件箱和发件箱。

参数说明：

- [输入] String address

  联系人号码。

- [输入] String bodyKey

  短信内容关键字。

  如果 address 和 bodyKey 均不为 null，则两者都匹配；如果 address 为 null， bodyKey 不为 null，则匹配 bodyKey；如果 address 不为 null，bodyKey 为 null，则匹配 address；如果 address 和 bodyKey 均为 null，则删除所有的短信。

- [返回值]

➢ERRORCODE_SUCCESS

执行正确；

➢其他值

错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.4.2.4 删除联系人

```
int deleteContact()
```

函数说明：

通过匹配姓名或电话号码，删除指定联系人。

参数说明：

- [输入] String name

  联系人姓名。

- [输入] String phone

  联系人电话号码。

  如果 name 和 phone 均不为 null，则两者都匹配；如果 name 为 null，phone 不为 null，则匹配 phone；如果 name 不为 null，phone 为 null，则匹配 name；如果 name 和 bodyKey 均为 null，则删除所有的联系人。

- [返回值]

  ➢ ERRORCODE_SUCCESS

  执行正确；

  ➢ 其他值

  错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.4.2.5 通话记录删除

```
int deleteCallHistory()
```

函数说明：

删除所有通话记录。

参数说明：

- [输入]

  无。

- [返回值]

  ➢ ERRORCODE_SUCCESS

  执行正确；

  ➢ 其他值

  错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.4.2.6 挂电话

int endCall()

函数说明：

挂断电话。

参数说明：

- [输入]

  无。

- [返回值]

  ➢ ERRORCODE_SUCCESS

  执行正确；

  ➢ 其他值

  错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B. 4. 2. 7 仅允许紧急呼叫控制

int setEnableOnlyEmergencyCall()

函数说明：

设置仅允许紧急呼叫。

参数说明：

- [输入]boolean enable

  true：限制通话，仅允许紧急呼叫；

  false：不限制通话。

- [返回值]

  ➢ ERRORCODE_SUCCESS

  执行正确；

  ➢ 其他值

  错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B. 4. 2. 8 数据上网控制

int setEnableMobileData()

函数说明：

禁止和允许移动数据上网功能。

参数说明：

- [输入]boolean enable

  true：允许；

  false：禁止。

- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；
  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.4.3 设备控制 API

B.4.3.1 远程重启终端

```
int reboot()
```

函数说明：

可远程重启终端。

参数说明：

- [输入]

  无。
- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；
  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.4.3.2 飞行模式开启

```
int setAirplaneOn()
```

函数说明：

可远程设置启动飞行模式。

参数说明：

- [输入]

  无。
- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；
  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B. 4. 3. 3 飞行模式关闭

```
int setAirplaneOff()
```

函数说明：

可远程设置关闭飞行模式。

参数说明：

- [输入]

  无。

- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；

  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B. 4. 3. 4 关闭终端

```
int shutDown()
```

函数说明：

远程关机。

参数说明：

- [输入]

  无。

- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；

  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B. 4. 3. 5 屏幕锁定

```
int lockScreen()
```

函数说明：

锁定设备，供设备安全管控场景使用。

参数说明：

- [输入]

无。

- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；
  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.4.3.6 屏幕解锁

```
int unlockScreen()
```

函数说明：

解锁设备，供设备安全管控场景使用。

参数说明：

- [输入]

  无。
- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；
  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.4.3.7 锁屏密码修改

```
int setPassword()
```

函数说明：

远程设置锁屏密码。

参数说明：

- [输入] String newPassword

  新密码值。
- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；
  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）

B.4.3.8 恢复出厂设置

```
int factoryReset()
```

函数说明：

可远程恢复出厂设置，供移动设备安全管控场景使用。

参数说明：

- [输入] int flags

  0：擦除用户数据；

  1：不擦除用户数据。

- [返回值]

  - ERRORCODE_SUCCESS

    执行正确；

  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.4.4 数据擦除和备份 API

B.4.4.1 邮件删除

```
int uninstallEmail()
```

函数说明：

指终端预装的邮件客户端的删除。

参数说明：

- [输入]

  无。

- [返回值]

  - ERRORCODE_SUCCESS

    执行正确；

  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.4.4.2 应用临时数据删除

```
int wipeApp()
```

函数说明：

远程删除应用的临时数据。

参数说明：

- [输入] String pName

  路径+文件名。

- [返回值]

  - ERRORCODE_SUCCESS

    执行正确；

  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B. 4. 4. 3 设备擦除

```
void wipeData()
```

函数说明：

设备数据擦除。

参数说明：

- [输入] int flags

  0： 擦除内置空间数据；

  1： 擦除外置空间数据-SD 卡。

- [返回值]

  - ERRORCODE_SUCCESS

    执行正确；

  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B. 4. 4. 4 视频删除

```
int removeVideo()
```

函数说明：

删除视频，供企业数据安全管控场景使用。

参数说明：

- [输入] String pName

  路径+文件名。(Video_Remove_All 表示删除所有视频。)

- [返回值]

  - ERRORCODE_SUCCESS

    执行正确；

  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B. 4. 4. 5 图片删除

```
int removePicture()
```

函数说明：

删除图片，供企业数据安全管控场景使用。

参数说明：

- [输入] String pName

  路径+文件名。(Picture_Remove_All 表示删除所有图片。)

- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；
  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B. 4. 4. 6 日历计划删除

```
int removeSchedule()
```

函数说明：

删除日历计划，供企业数据安全管控场景使用。

参数说明：

- [输入] String pName

  路径+文件名。(Schedule_Remove_All 表示删除所有日历计划。)

- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；
  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B. 4. 5 拷贝、粘贴控制 API

B. 4. 5. 1 剪贴板控制

```
int setEnableClipboard()
```

函数说明：

允许和禁止启用剪贴板。

参数说明：

- [输入]boolean enable

  true：允许；

  false：禁止。

- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；

  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.4.5.2 截屏控制

```
int setEnableCaptureScreen()
```

函数说明：

允许和禁止启用截屏功能。

参数说明：

- [输入]boolean enable

  true：允许；

  false：禁止。

- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；

  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.4.6 GPS 控制和管理子类 API

```
Class MtmGPSManager
```

类说明：

GPS 控制和管理子类，负责管理智能终端设备的各项 GPS 功能。MtmGPSManager 属于 EMCG_MTM_Manager 类的子类。

B.4.6.1 GPS 控制

```
int setEnableGps()
```

函数说明：

允许 APP 获得用户位置信息如获取经纬度等。

参数说明：

- [输入] boolean enable

  true：允许；

  false：禁止。

- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；

  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.4.7 开放接口控制 API

B.4.7.1 防火墙接口(iptables)调用配置管理

int setEnableIptables ()

函数说明：

允许或禁止配置开发防火墙接口，供应用调用。

参数说明：

- [输入]boolean enable

  true：允许；

  false：禁止。

- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；

  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.5 外设控制 API

B.5.1 USB 管理子类 API

Class MtmUSBManager

类说明：

USB 管理子类，负责管理智能终端设备的各项 USB 功能。MtmUSBManager 属于 EMCG_MTM_Manager 类的子类。

B.5.1.1 USB 调试模式控制

int setEnableUsbAdb()

函数说明：

开启和关闭 USB 调试。

参数说明：

- [输入]boolean enable

  true：开启；

  false：关闭。

- [返回值]

  - ERRORCODE_SUCCESS

    执行正确；

  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.5.1.2 USB 控制

```
int setEnableUsbFunction()
```

函数说明：

是否允许启用 USB 通讯；禁用 USB 通讯后，只保留 USB 充电功能。

参数说明：

- [输入]boolean enable

  true：允许；

  false：禁止。

- [返回值]

  - ERRORCODE_SUCCESS

    执行正确；

  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.5.1.3 USB-MTP 控制

```
int setEnableUsbMTP()
```

函数说明：

允许和禁止启用 USB 媒体流传输。

参数说明：

- [输入]boolean enable

  true：允许；

  false：禁止。

- [返回值]

- ERRORCODE_SUCCESS

  执行正确；

- 其他值

  错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.5.1.4 USB-PTP 控制

```
int setEnableUsbPTP()
```

函数说明：

允许和禁止启用 USB 图片传输。

参数说明：

- [输入]boolean enable

  true：允许；

  false：禁止。

- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；

  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.5.1.5 USB 大容量存储控制

```
int setEnableUsbMassStorage()
```

函数说明：

允许和禁止使用 USB 大容量存储。

参数说明：

- [输入]boolean enable

  true：允许；

  false：禁止。

- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；

  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.5.1.6 USB 转网卡控制

```
int setEnableUsbRNDIS()
```

函数说明：

允许和禁止使用 USB 转网卡。

参数说明：

- [输入]boolean enable

  true：允许；

  false：禁止。

- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；
  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.5.1.7 USB 网络共享控制

```
int setEnableUsbTethering()
```

函数说明：

开启和关闭 USB 网络共享。

参数说明：

- [输入] boolean enable

  true：开启；

  false：关闭。

- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；
  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.5.2 存储管理 API

B.5.2.1 存储访问控制(SD 卡或 TF 卡)

```
int setEnableSDCard ();
```

函数说明：

指允许和禁止访问 SD 卡存储。

参数说明：

- [输入] boolean enable

  true：允许；

  false：禁止。

- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；

  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.5.2.2 挂载 SD 卡

```
int mountSDCard ();
```

函数说明：

挂载 SD 卡（TF 卡），供设备安全管控场景使用，或支持系统和 App 访问 SDCard。

参数说明：

- [输入]

  无。

- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；

  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.5.2.3 卸载 SD 卡

```
int umountSDCard ();
```

函数说明：

卸载 SD 卡（TF 卡），禁止系统和 App 访问 SDCard。

参数说明：

- [输入]

  无。

- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；

  - 其他值

错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.5.2.4 SD 卡擦除

```
int eraseSDCard();
```

函数说明：

清除 SD 卡中所有的数据，可指定内置存储卡或外置存储卡。

参数说明：

- [输入] int flag

  0：清除内置存储卡；

  1：清除外置存储卡。

- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；

  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.5.3 摄像头管理子类 API

```
Class MtmCamaraManager
```

类说明：

摄像头管理子类，负责管理智能终端设备的各项摄像头功能。MtmCamaraManager 属于 EMCG_MTM_Manager 类的子类。

B.5.3.1 摄像头控制

```
int setEnabledCamera ();
```

函数说明：

启用和禁用摄像头，供移动设备安全管理场景使用。

参数说明：

- [输入] boolean enable

  true：启用；

  false：禁用。

- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；

- 其他值

  错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B. 5. 4 录音控制子类 API

```
Class MtmRecorderManager
```

类说明：

录音控制子类，负责管理智能终端设备的各项录音相关功能。MtmRecorderManager 属于 EMCG_MTM_Manager 类的子类。

B. 5. 4. 1 麦克风控制

```
void setMicrophoneMute();
```

函数说明：

允许和禁止启用麦克风进行录音等操作。

参数说明：

- [输入] boolean enable

  true：启用；

  false：禁用。

- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；

  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B. 5. 4. 2 录音控制

```
int setEnableAudioRecorder（）;
```

函数说明：

是否允许启用录音功能。

参数说明：

- [输入] boolean enable

  true：启用；

  false：禁用。

- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；
  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.6 应用软件管理和控制 API

B.6.1 应用配置 API

本章节描述对邮件等终端自带应用参数的配置。

B.6.1.1 配置添加邮件

```
int addMailConfig ();
```

函数说明：

向终端系统添加邮件配置，支持 Exchange、POP3、IMAP 等帐号。

参数说明：

- [输入] String configstring

  邮箱配置信息，XML 结构，详见附录格式。
- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；
  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.6.1.2 删除邮件配置

```
int deleteMailConfig ();
```

函数说明：

删除终端系统邮件配置信息。

参数说明：

- [输入]

  无。
- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；

- 其他值

  错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.6.1.3 更新邮件配置

```
int updateMailConfig();
```

函数说明：

向终端系统更新覆盖邮件配置，支持 Exchange、POP3、IMAP 等帐号。

参数说明：

- [输入] String configstring

  邮箱配置信息，XML 结构，详见附录格式。

- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；

  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.6.1.4 获取设备当前运行中的软件列表

```
int getRunningAppList();
```

函数说明：

获取设备当前运行中的软件包名列表，如 com.tencent.qq、com.baidu.map 等。

参数说明：

- [输入]String returnNames

  返回一个字符串数组，包含多个 App 包名字符串。

- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；

  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.6.1.5 获取软件安装历史纪录

```
String getAppInstallHistory();
```

函数说明：

获取软件安装历史纪录。

参数说明：

- [输入]

  无。

- [返回值] String

  软件安装历史纪录以 XML 格式呈现，包含每个 APP 安装日期和时间、包名、是否安装成功等信息，详细格式参见附件。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.6.1.6 启动指定应用进程

```
int startApp();
```

函数说明：

启动指定包名的应用进程。

参数说明：

- [输入]String appID

  输入应用程序包名，如：“com.baidu.browser”。

- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；

  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.6.1.7 停止指定应用进程

```
int endApp();
```

函数说明：

停止指定包名的应用进程。

参数说明：

- [输入]String appID

  输入应用程序包名

- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；

  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B. 6. 1. 8　新增或删除开机自启动控制

```
int setEnableAppAutoRun();
```

函数说明：

新增或删除指定 APP 开机自启动控制。

参数说明：

- [输入]String appID

  输入应用程序包名，

- [输入]boolean enable

  true：添加；

  false：删除。

- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；

  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B. 6. 2　原生浏览器控制子类 API

```
Class MtmBrowserManager
```

类说明：

原生浏览器控制子类，属于 EMCG_MTM_Manager 类的子类。

B. 6. 2. 1　浏览器访问控制

```
int setEnableNativeBrowser();
```

函数说明：

禁止和允许用户使用原生浏览器。

参数说明：

- [输入] boolean　enable

  true：允许；

  false：禁止。

- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；

  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.6.2.2 浏览器 cookie 控制

```
int setEnableBrowserCookie();
```

函数说明：

禁止和允许浏览器使用 cookie。

参数说明：

- [输入] boolean enable

  true：允许；

  false：禁止。

- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；

  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.6.2.3 浏览器 javascript 控制

```
int setEnableBrowserJavascript();
```

函数说明：

禁止和允许浏览器使用 javascript。

参数说明：

- [输入] boolean enable

  true：允许；

  false：禁止。

- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；

  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.6.2.4 浏览器弹出窗口控制

```
int setEnableBrowserPopWindow();
```

函数说明：

禁止和允许浏览器弹出窗口。

参数说明：

- [输入]boolean enable

  true：允许；

  false：禁止。

- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；

  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.6.2.5 浏览器自动填充控制

```
int setEnableBrowserAutoFill ();
```

函数说明：

禁止和允许浏览器自动填充。

参数说明：

- [输入]boolean enable

  true：允许；

  false：禁止。

- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；

  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.6.2.6 浏览器 https 安全警告控制

```
int setEnableBrowserSafetyWarning ();
```

函数说明：

禁止和允许浏览器针对不安全内容弹出警告。

参数说明：

- [输入]boolean enable

  true：允许；

  false：禁止。

- [返回值]
  - ERRORCODE_SUCCESS

执行正确；

- 其他值

错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.6.2.7 URL 访问控制

```
int setBrowserUrlBlockingMode();
```

函数说明：

设置浏览器 URL 访问控制策略。

参数说明：

- [输入]int blockingMode

  URL 访问控制模式：

  - 0：禁用访问控制；
  - 1：启用黑名单模式；
  - 2：启用白名单模式。

- [返回值]
  - ERRORCODE_SUCCESS

  执行正确；

  - 其他值

  错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.6.2.8 添加浏览器书签配置

```
int addBrowserBookmark();
```

函数说明：

向浏览器添加书签。

参数说明：

- [输入]String bookmarkName

  书签名称；

- [输入]String url

  书签 Url 地址。

- [返回值]
  - ERRORCODE_SUCCESS

  执行正确；

- 其他值

  错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.6.3 应用安装、运行和删除子类 API

```
Class MtmAppManager
```

类说明：

应用安装、运行和删除管理子类，属于 EMCG_MTM_Manager 类的子类。

B.6.3.1 静默安装应用

```
int forceInstallApp ();
```

函数说明：

强制并静默安装指定路径 App。

参数说明：

- [输入]String appFilePath

  待安装 App 文件路径。

- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；

  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.6.3.2 静默卸载应用

```
int forceUninstallApp ();
```

函数说明：

强制并静默卸载指定包名 App。

参数说明：

- [输入]String appID

  待卸载 App 包名。

- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；

  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.6.3.3 禁止或允许用户安装 App

```
int setEnableAppInstallation();
```

函数说明：

禁止和允许用户安装 App，包括安装存储在 SD 卡或内存中的软件包，或者通过以及第三方管理软件安装 App。

参数说明：

- [输入]boolean enable

  true：允许；

  false：禁止。

- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；

  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。禁止用户安装 App 后，系统或 MDM 等程序通过 forceInstallApp()函数调用仍可以安装 App。

B.6.3.4 禁止或允许用户卸载 App

```
int setEnableAppUninstallation();
```

函数说明：

禁止和允许用户卸载 App，包括用户在系统设置中卸载或者通过 adb 以及第三方管理软件卸载 App。

参数说明：

- [输入]boolean enable

  true：允许；

  false：禁止。

- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；

  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。禁止用户安装 App 后，系统或 MDM 等程序通过 forceUninstallApp()函数调用仍可以卸载 App。

B.6.3.5 禁止或允许用户强行关闭 App

```
int setEnableKillApp();
```

函数说明：

禁止和允许用户强行关闭/杀死指定 App 或服务，包括用户在系统设置中关闭或者通过 adb 以及第三方管理软件关闭 App。

参数说明：

- [输入]String appID

  指定包名的 App 或服务。

- [输入]boolean enable

  true：允许；

  false：禁止。

- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；

  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.6.3.6 禁止或允许 adb 控制

```
int setEnableADB();
```

函数说明：

禁止和允许用户通过 ADB 安装或卸载 App，包括依赖 ADB 协议的第三方管理软件的控制功能。

参数说明：

- [输入]boolean enable

  true：允许；

  false：禁止。

- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；

  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.6.3.7 禁止或允许指定 App 运行

```
int setEnableExecuteApp();
```

函数说明：

禁止和允许系统执行指定包名的 App。

参数说明：

- [输入]String appID

  指定包名的 App 或服务。

- [输入]boolean enable

  true：允许；

  false：禁止。

- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；

  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B. 6. 3. 8 删除指定应用数据

```
int eraseAppData();
```

函数说明：

删除指定应用数据，同时关闭该应用。

参数说明：

- [输入] String appID

  指定 App 包名。

- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；

  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B. 6. 3. 9 禁止或允许“最近应用”功能

```
int setEnableRecentApp();
```

函数说明：

禁止或允许用户使用系统的“最近应用”功能。

参数说明：

- [输入]boolean enable

  true：允许；

false：禁止。

- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；

  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B. 6. 4 应用黑白名单子类 API

```
Class MtmAppFilterManager
```

类说明：

应用黑白名单过滤功能子类，属于 EMCG_MTM_Manager 类的子类。

B. 6. 4. 1 设置应用运行管控模式

```
int setAppRunningBlockMode();
```

函数说明：

设置 App 黑/白名单运行控制模式。

参数说明：

- [输入]int blockingMode

  App 控制模式：

  - 0：禁用黑白名单控制；
  - 1：启用黑名单模式；
  - 2：启用白名单模式。

- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；

  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B. 6. 4. 2 设置应用运行管控配置信息

```
int setAppRunningBlockConfig();
```

函数说明：

设置应用运行管控配置信息，内容一般同时包含黑名单列表和白名单列表信息，通过使用 setAppRunningBlockMode 激活黑名单管控或白名单管控。

参数说明：

- [输入] String blockingConfigInfo

  应用运行管控配置信息。

  应用运行管控配置文件格式示例如下：

```
<?xml version="1.0" encoding="UTF-8"?>
<APP_RUNNING_BLOCK_CONFIG>
    <BLACKLIST>
      <APPITEM>
        <NAME>爱消除</NAME>
        <APPID>com.test.aixiaochu</APPID>
      </APPITEM>
      <APPITEM>
        <NAME>快跑</NAME>
        <APPID>com.unknown.kuaipao</APPID>
      </APPITEM>
    </BLACKLIST >
    <WHITELIST>
      <APPITEM>
        <NAME>微信</NAME>
        <APPID>com.tencent.weixin</APPID>
      </APPITEM>
      <APPITEM>
        <NAME>百度输入法</NAME>
        <APPID>com.baidu.ime</APPID>
      </APPITEM>
    </WHITELIST>
</APP_RUNNING_BLOCK_CONFIG>
```

- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；

  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B. 6. 4. 3 设置应用安装管控模式

```
int setAppInstallBlockMode();
```

函数说明：

设置 App 黑/白名单安装控制模式。

参数说明：

- [输入]int blockingMode

  App 控制模式：

- 0：禁用黑白名单控制；
- 1：启用黑名单模式；
- 2：启用白名单模式。

- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；
  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B. 6. 4. 4 设置应用安装管控配置信息

```
int setAppInstallBlockConfig();
```

函数说明：

设置应用安装管控配置信息，内容一般同时包含黑名单列表和白名单列表信息，通过使用 setAppRunningBlockMode 激活黑名单管控或白名单管控。

参数说明：

- [输入] String blockingConfigInfo

  应用安装管控配置信息。

  应用安装管控配置文件格式示例如下：

```
<?xml version="1.0" encoding="UTF-8"?>
<APP_INSTALL_BLOCK_CONFIG>
    <BLACKLIST>
      <APPITEM>
        <NAME>爱消除</NAME>
        <APPID>com.test.aixiaochu</APPID>
      </APPITEM>
      <APPITEM>
        <NAME>快跑</NAME>
        <APPID>com.unknown.kuaipao</APPID>
      </APPITEM>
    </BLACKLIST >
    <WHITELIST>
      <APPITEM>
        <NAME>微信</NAME>
        <APPID>com.tencent.weixin</APPID>
      </APPITEM>
      <APPITEM>
        <NAME>百度输入法</NAME>
        <APPID>com.baidu.ime</APPID>
```

```
        </APPITEM>
    </WHITELIST>
</APP_INSTALL_BLOCK_CONFIG>
```

- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；
  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.6.4.5 强制指定应用运行

```
int forceAppKeepRunning();
```

函数说明：

设置强制运行的指定应用列表信息，在该信息中包含的应用将被操作系统强制启动并保持运行。

参数说明：

- [输入] String keepRunningAppInfo

  应用强制运行管控配置信息。

  应用强制运行管控配置文件格式示例如下：

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

<APPITEM>
    <NAME>爱消除</NAME>
    <APPID>com.test.aixiaochu</APPID>
</APPITEM>
<APPITEM>
    <NAME>快跑</NAME>
    <APPID>com.unknown.kuaipao</APPID>
</APPITEM>
</KEEP_RUNNING_APP_CONFIG>
```

- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；
  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B. 6. 4. 6 卸载黑名单中枚举的应用

```
int uninstallAppInBlackList ();
```

函数说明：

卸载应用安装黑名单中枚举的所有应用。

参数说明：

- [输入]

  无。

- [返回值]

  - ERRORCODE_SUCCESS

    执行正确；

  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MDM 客户端身份认证的函数调用。

关于应用安装黑名单参见 setAppInstallBlockConfig 函数说明。

B. 6. 4. 7 设置禁止卸载的应用名单(必装应用)

```
int setForbitUninstallationList();
```

函数说明：

设置禁止卸载的应用名单，即必装应用。设置后这些应用将无法被用户自行卸载。

参数说明：

- [输入] String forbitUninstallationAppInfo

  应用运行管控配置信息。

  应用运行管控配置文件格式示例如下：

```
<?xml version="1.0" encoding="UTF-8"?>
<FORBIT_UNINSTALL_APP_CONFIG>
<APPITEM>
    <NAME>爱消除</NAME>
    <APPID>com.test.aixiaochu</APPID>
</APPITEM>
<APPITEM>
    <NAME>快跑</NAME>
    <APPID>com.unknown.kuaipao</APPID>
</APPITEM>
</FORBIT_UNINSTALL_APP_CONFIG>
```

- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；

- 其他值

  错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.6.4.8 设置系统可以访问的 URL 黑白名单模式

```
int setURLBlockingMode();
```

函数说明：

设置系统可以访问的 URL 管控模式：黑名单、白名单及普通模式。

参数说明：

- [输入]int blockingMode

  URL 控制模式：

  - 0：禁用黑白名单控制；
  - 1：启用黑名单模式；
  - 2：启用白名单模式。

- [返回值]
  - ERRORCODE_SUCCESS

  执行正确；

  - 其他值

  错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.6.4.9 设置系统可以访问的 URL 黑白名单配置信息

```
int setURLBlockingConfig();
```

函数说明：

设置系统可以访问的 URL 黑白名单配置信息，内容一般同时包含黑名单列表和白名单列表信息，通过使用 setURLBlockingMode 激活黑名单管控或白名单管控。

参数说明：

- [输入] String blockingConfigInfo

  URL 运行管控配置信息。格式示例如下：

```
<?xml version="1.0" encoding="UTF-8"?>
<URL_BLOCK_CONFIG>
    <BLACKLIST>
      <URLITEM>
          <NAME>Bad Url#1</NAME>
          <URL>fakebank.com</ URL >
      </URLITEM>
      <URLITEM>
```

```
            <NAME> Bad Url#1</NAME>
            <URL>www.getsomemoney.com</URL>
          </URLITEM>
        </BLACKLIST>
        <WHITELIST>
          <URLITEM>
            <NAME>企业内网</NAME>
            <URL>192.168.1.112</URL>
          </URLITEM>
        </WHITELIST>
    </URL_BLOCK_CONFIG>
```

- [返回值]
  - ERRORCODE_SUCCESS

    执行正确；
  - 其他值

    错误，详见“错误码定义”章节。

注意事项：

需要事先完成 MTM-App 身份认证的函数调用（调用 Authenticate）。

B.7 错误码定义

函数通用错误码定义表

错误码定义	值	含义
ERRCODE_SUCCESS	0	函数执行正确
ERRCODE_UNKNOWN	-1	未知异常错误
ERRCODE_EXECUTEERR	-2	执行失败
ERRCODE_PERMISSIONERR	-3	权限错误
ERRCODE_OBJECTNOTFOUNDERR	-4	未找到对象（如 SIM 卡、WIFI 设备、蓝牙等）
ERRORCODE_INVALIDSIGNATURE	-11	APP 签名证书无效

参考文献

《企业移动智能终端应用开发、安装、运行管控机制（ZISIA/EMCG 2014-3）》

《可信云服务认证评估方法 第十二部分：企业移动化管理云服务(EMM)》

《信息安全技术 信息系统安全等级保护基本要求-第四部分 移动互联要求（GB/T 22239-4-2015）》（草案）

Association Standard

T/ZSA 52-2018

Enterprise Mobile Terminal Management API Specification

Issue Date 12-19-2018 Implementation Date 3-1-2019

Issued by Zhongguancun Standard Association

Contents

Foreword

This Standard is used by Android system MDM and API transfering operation system, it norms the EMCG-MTM-API which is needed by Mobile Intelligent Management. It could solve the problem of MDM API differences, the difficulties of the controlling various enterprise MDM devices, and etc.

The Standard can be applied to mobile intelligent terminal manufactures, EMMS suppliers, mobile terminal operating system providers and evaluation agency, etc.

The Standard is much pratical, it applies “Real-name Calculation” theory and it orients with the market trends of enterprise mobile informatization. It aims at the weakness of controlling Android system and improves the development of Android APP, but not influenced the sales. It also improves the current situation of lack of effective regulations and the increased of software security threats .

Please note that some contents in this document may involve patents. Zhongguancun Standardization Association shall not be held responsible for identifying such patents.

The standard was proposed and under the jurisdiction of Zhongguancun Standardization Association - Technical Committee.

This standard was drafted by: Zhongguancun Cyberspace Affairs Industry Alliance; Huawei Technologies Co. LTD; Lenovo Group; Trustmobi Co. LTD;Qihoo of Beijing Science and Technology Co. LTD; Beijing Xiaomi Technologies Co. LTD;China Institute of Information;Beijing Syberos Technology Co.LTD; Beijing Guoxin Networks Technology Co. LTD;

Chief drafters of this standard are:：Wang Ke Zhongguancun Cyberspace Affairs Industry Alliance EMCG; Yi Pingping Huawei Technologies Co. LTD; Wang Pan Lenovo Group; Hu Wenzhuang ZTE Corp; Liu Changshan ZTE Corp.; Zhang Jianguo Trustmobi Co. LTD; Ma Shaobu Beijing Syberos Technology Co.LTD; Li Yiming Qihoo of Beijing Science and Technology Co. LTD; Nie Juhu Beijing Xiaomi Technologies Co. LTD; Zhou Hongbin Beijing Syberos Technology Co.LTD; Min Dong China Institute of Information.

Introduction

With the rapid development of Internet and portable intelligent terminal technology, more and more enterprise and governments start a new work pwttern and use the mobile phone and tablet computer handeling their works. On this case, the related technical products enjoy the vast markets prospects. At present, the mobile terminal on the markets could not meet the security requirement of the enterprise information from equipment control and the software security. In order to sovle the problem of mobile terminal security, the demostic and oversea manufacturers have proceeded to think out various solutions.

The openness of Android system could support all kinds of manufacturers to push out different Android mobile terminal brands, models and versions, but it is hard for MDM platform manage so many different brands, models and versions at the same time as the lack of unified software interface.

In order to meet the requirement of the compatibility for so many Android MDM brands and versions, the EMCG organize a lot of manufacturers to research the standard of mobile terminal management interface.

The standard could provide the unified terminal resource control software interface, improving MDM compatibility and reducing system development and promotion costs.

Enterprise Mobile Terminal Management API Specification

1 Scope

EMCG-MTM-API is standard based on this document, which is required by the management of mobile intelligent terminal. Mobile intelligent terminal manufacturers offer the general mobile terminal equipments based on this document. EMMS software programming staff develops the MTM-App per the document to achieve the mobile intelligent terminal compatibility of various brands and models.

The standard can be applied to mobile intelligent terminal manufactures, EMMS suppliers, mobile terminal operating system providers and evaluation agency, etc.

2 Normative references

There are no normative references in this document.

3 Terms and Definition

3.1 EMM

EMM is to achieve the management of the enterprises employees to handle their mobile terminal works by means of Cloud, which includes MDM, MAM and MCM, etc.

3.2 EMMS

EMMS is to achieve the function of EMM, which consists of mobile intelligent terminal, date transmission channel and Cloud service of enterprise mobile terminal management.

3.3 MTM-App

MTM-App is an App installed on the mobile terminal; normally it is developed by the EMMS suppliers and charge for the receiving & carries out the monitoring instruction which is sent out by Cloud service of enterprise mobile terminal management

3.4 Cloud service of enterprise mobile terminal management

Cloud service of enterprise mobile terminal management is developed by EMMS supplier. It is established on the enterprise of mobile terminal management or it could be served as a platform

which can be offered the related information service. Cloud service of enterprise mobile terminal management is to achieve the management of MDM, MAM and MCM by wireless network.

3.5 EMCG-MTM-API Transfering Authority Mechanism

The reason why designed the authority mechanism is to ensure the moobile security and standard the superior API.(Please see No 7. Authority Mechanism)

3.6 App

APP is the abbreviation of application, it refers mobile intelligent terminal.

3.7 EMCG Official Certificate

EMCG Official Certificate is used for the signature release of EMCG-APP. (Please reference 《The regulation mechanisms of enterprise mobile terminal development, installation and operation (T/ZSA 3001.01-2016)》)

3.8 EMCG-App

EMCG-App is released or signed by official institute. (Please reference《The regulation mechanisms of enterprise mobile terminal development, installation and operation (T/ZSA 3001.01-2016)》)

3.9 Official Authority

Official Authority is an social organization defined by EMCG and provides standard service assurance to EMCG. (Please reference 《The regulation mechanisms of enterprise mobile terminal development, installation and operation (T/ZSA 3001.01-2016)》)

3.10 EMCG(Enterprise Mobile Computing Group)

EMCG is established by Zhongguancun Cyberspace Affairs Industry Alliance

4 Abbreviation

API：Application programming interface

App：Application

BYOD：Bring Your Own Device

EMCG：Enterprise Mobile Computing Group

EMM：Enterprise Mobile Management

GPS：Global Positioning System

ICCID：Integrate Circuit Card Identity

IMEI：International Mobile Equipment Identity

MAC：Media Access Control（Medium Access Control）

MDM：Mobile Device Management

MTM-API：Mobile Terminal Management-API

MTM-App：Mobile Terminal Management-App

SIM：Subscriber Identity Module

SSID：Service Set Identifier

WiFi：Wireless Fidelity

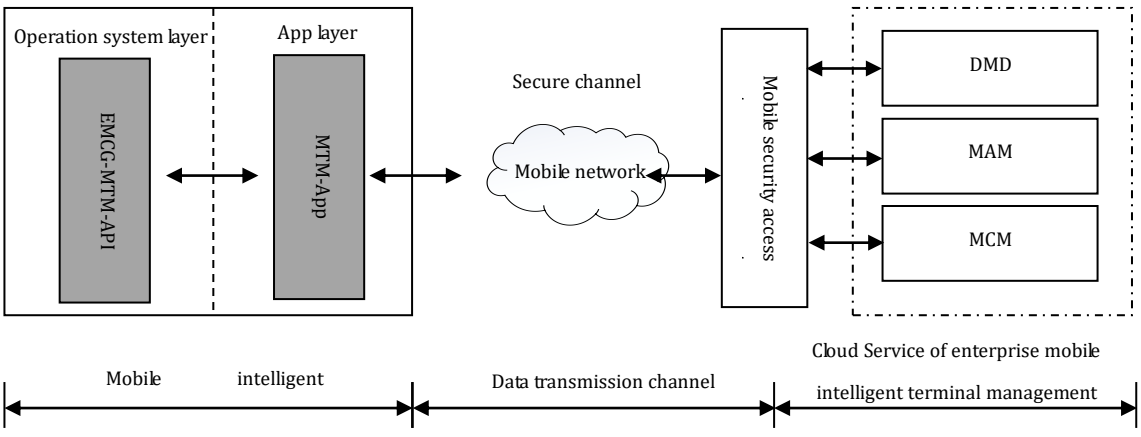
5 System Structure

To achieve the EMCG-MTM-API mobile terminal intelligence, configuration and monitoring under the management of EMMS.

EMMS basic structure is showed as picture 1, which includes mobile intelligent terminal, data transmission channel and cloud Service of Enterprice Mobile Management.

Mobile intelligent terminal can be enterprise specific equipment (only for enterprise mobile business) or BYOD equipment (mix installation for enterprise mobile business and personal APP). The mobile intelligent terminal should have EMCG-MTM-API calling mechanism (Please see No 7. Authority Mechanism). Also it could see the reference of EMCG 《The regulation mechanisms of enterprise mobile terminal development, installation and operation (T/ZSA 3001.01-2016)》) to design.

EMCG-MTM-API is one of the components of operating system, it could receive system calls which is sent out by MTM-App to achieve the control of mobile intelligent terminal.



Picture 1 EMMS Structure

Cloud Service of Enterprice Mobile Management gives commands and instructions to MTM-App by means of mobile network, which could achieve the management of mobile devices, status monitoring, security policy and so on.

EMMS data transmission channel should use necessary security measures to ensure the security of data transmission.

6 EMCG-MTM-API Classification

6.1 Basic API

Basic API (34 kinds of functions) could acquire the related devices equipment information that MTM-App needs. Normally, MTM-App can call the basic API functions without authorization. The basic API function is shown as below.

Table 1 EMCG-MTM-API Basic Functions

No	Function	Description	Function Name	Optinal Function	Appendix A Page
	Network configuration and management API	Refering the configuration management of APN、VPN, such as Account No, password, SSID.			错误！未定义书签。
	Subclass of Bluetooth API		Subclass Bluetooth Manager		错误！未定义书签。
	Bluetooth State		getState		错误！未定义书签。
	Get Bluetooth connected history	Get Bluetooth connected history information, used in device security control setting.	getBondedDevices		错误！未定义书签。
	Get bluetooth MAC address information	Getting bluetooth MAC address information, used in device security control setting	getAddress		错误！未定义书签。
	WiFi Subclass Managememt API		Subclass WifiManager		错误！未定义书签。
	Get SSID Information	Getting SSID (Wireless network name) information , provide identity authentication and device identification.	getSSID		错误！未定义书签。
	WiFi State		getWifiState		错误！未定义

					书签。
	System Function Control API	It refers to the control of system functions which is provided by the terminal.			错 误 ！ 未 定 义 书签。
	Subclass of Telephone Information Management API	Telephone No information of SIM card, information state, ISO information could receive the information by type of network information.	Subclass Telephony Manager		错 误 ！ 未 定 义 书签。
	Get SIM card Telephone No Information	Getting SIM card Tel No information, provide identity authentication and device identification. (Please noted that only parts of telephone No could receive the info, if the tel No data isn't loaded into SIM card, it could not received.)	getLine1Number		错 误 ！ 未 定 义 书签。
	SIM card could get info by network type	SIM card could get info by network type to calculate and audit.	getNetworkType		错 误 ！ 未 定 义 书签。
	Get SIM Card's ISO Info	Getting SIM card's info to calculate and audit.	getSimCountryIso		26
	Get SIM Card State	It could not get SIM card's state info in the case of without card or unknown state, it will be needed pin, puk and network pin to unlock and recover well state.	getSimState		27
	Get SIM Operator	Get SIM Operator Name	getSimOperatorNa		27

	Name		me		
	Get Network Operator Name	Get Network Operator Name	getNetworkOperatorName		错误！未定义书签。
	Get Network Operator Sequence Identity	Get network operator registered sequence identity（MCC+MNC）	getNetworkOperator		28
	Get Network Roaming State Info	Getting network roaming state info to judge if it is roaming state. If yes, it could use the method of setting open/disable and data synchronization.	isNetworkRoaming		28
	Get equipment device API				28
	Get IMEI Info (Mobile Communication Equipment Identity)	Get IMEI Info (Mobile Communication Equipment Identity). Notice: If it is without mobile communication PAD, there is without the identification	getIMEI		错误！未定义书签。
	Get ICCID Info (IC card's unique identification No)	Getting ICCID Info (IC card's unique identification No), provide identity authentication and device identification.	getICCID		错误！未定义书签。
	Get IMSI (SIM card's unique identification)	Getting IMSI (SIM card's unique identification), provide identity authentication and device identification.	getIMSI		29

	Get Mac Address Info	Getting Mac address info, provide identity authentication and device identification.	getMacAddress		错误！未定义书签。
	Get IP Address	Getting IP address to calculate and audit.	getIpAddress		错误！未定义书签。
	Get Device Serial Number	Getting Device Serial Number	getDeviceSerialNumber		错误！未定义书签。
	Get ROM Version	Getting ROM version, used in device management setting.	getRomVersion		错误！未定义书签。
	Get CPU usage rate, number, frequency and item No info	Getting CPU usage rate, number, frequency and item No info, used in device security control setting.	getCpuInfo		错误！未定义书签。
	Get Battery Level	Getting battery level, used in device security control setting.	getBatteryLevel		错误！未定义书签。
	Get Memory Info	Get total memory capacity info and available info.	getMemoryInfo		错误！未定义书签。
	Get Internal Storage Info	Going back inside memory capacity and available space.	getInternalStorageInfo		错误！未定义书签。
	Get SD Card Info	Going back outside memory capacity and available space.	getSDCardInfo		错误！未定义书签。
	OS version	Getting OS version, used in device management setting.	getOSVersion		32
	Device model	Getting device model, used in device	getDeviceModel		错误！未定义

		management setting.			书签。
	Get GPS Info API				错误！未定义书签。
	Get GPS current location	Getting current device location info	getCurrentLocation		错误！未定义书签。
	Get GPS Last Known Location	Getting current users location info, such as latitude and longitude and time.	getLastKnownLocation		错误！未定义书签。
	Get Peripheral Info API				34
	Get Number Of Cameras	Getting No of cameras, used in device management setting.	getNumberOfCameras		错误！未定义书签。
	Get APP Info API				34
	App Data Traffic	Getting App data traffic, used in device management setting.	getAppDataTrafficUsage		错误！未定义书签。
	Get Installed App list	Getting installed lsit, such as com.tencent.qq, com.baidu.map and etc.	getInstalledAppList		错误！未定义书签。
	Get specific package App Info	Getting specific installed App info, including program name, version,size, time length and start-up times, used in device management setting.	getAppInfo		35
	High level API calls authorization	Terminal authorization MTM-App calls high-level API.	Authenticate		35

Please see the reference of appendix A & C for the specific definition of basic API.

6.2 High-level API

High-level API (total 105 functions) define another APIs (except 6.1 basic API), which comprehensively and deeply implement the control of enterprise mobile terminal. High-level MTM-App API must be signed by EMCG and authority mechanism before called the high-level API functions. Below table 2 is EMCG-MTM-API high-level function.

Table 2 EMCG-MTM-API High-level Function

No	Function	Description	Function Name	Optinal Function	Appendix B Page
	Network configuration and management API	Refering the configuration management of APN、VPN, such as Account No, password, SSID.			45
	APN Subclass Management API		Sublass MtmAPNManager		**错误!未定义书签。**
1	Query APN		queryApn		**错误!未定义书签。**
2	Add APN		addApn		**错误!未定义书签。**
3	Delete Specific APN		deleteApn		39
4	Update APN		updateApn		40
5	Set Prefer APN		setPreferApn		40
	VPN Management API	Add, delete and update VPN, connect and disconnect VPN; could set parameter automatic,			41

		which used in device security control setting.			
6	Query VPN		queryVpn		错误! 未定义书签。
7	Add VPN		addVpn		41
8	Delete Specific VPN		deleteVpn		42
9	Update Specific VPN Configuration		updateVpn		42
10	Connect Specific VPN		connectVpn		42
11	Disconnect VPN		disConnectVpn		错误! 未定义书签。
	NFC Control Subclass API		Sublass MtmNFSManager		错误! 未定义书签。
12	Set Enable NFC		setEnableNFC		错误! 未定义书签。
	Bluetooth management subclass API		Subclass MtmBluetoothAdapter		错误! 未定义书签。
13	Only Enable Specific type of Bluetooth	Only enable specific type of Bluetooth, such as Bluetooth equipment, audio and video equipment connected with PC.	setWhiteBluetoothType		错误! 未定义书签。
14	Set Enable Bluetooth	Open or off the Bluetooth is acceptable.	setEnableBluetooth		错误! 未定义书签。
15	Set Enable Bluetooth Share Function	Set Enable Bluetooth Share Function	setEnableBluetoothShare		45
	WiFi Subclass Management API		Subclass WifiManager		错误! 未定义书签。
16	WiFi Connected History Record		getConfiguredNetworks		错误! 未定义

					书签。
17	Set Enable WiFi	Open or off WiFi is acceptable.	setEnableWifi		错误! 未定义书签。
18	Set Enable WiFi Share	Seting enable WiFi share	setEnableWifiShare		46
19	Add WiFi Network		addNetwork		47
20	Remove WiFi Network		removeNetwork		47
21	Update WiFi Network		updateNetwork		48
22	Set WiFi Network Priority		enableNetwork		48
23	Disable WiFi Network		disableNetwork		错误! 未定义书签。
24	Disconnect WiFi Network		disconnect		49
25	Reconnect WiFi Network		reconnect		49
	System Function Control API	It refers to the control of system function which provided by terminal.			50
	System Enable API				错误! 未定义书签。
26	Set Enable Home Key	The scene of mobile marketing, mobile classroom are not allowed to swift the Home Key.（Android 5.0 version supports App block.）	setEnableHomeKey	*	错误! 未定义书签。
27	Set Enable Back Key	The scene of mobile marketing, mobile classroom are not allowed to swift the Back Key.（Android 5.0 version supports App block.）	setEnableBackKey	*	错误! 未定义书签。
28	It appoints which package name are single App models.	It only allows specific App and exit the current interface is forbidden. It could achieve single App	setKioskApp		51

		mode by home key, back key or App black.			
29	Set Date Tine	（CDMA terminal default network synchronization is not included.）	setDataTime		51
30	Set Default Launcher		setDefaultLauncher	*	52
31	Hide Navigation Bar		hideNavigationBar	*	52
	Communication Management API	SIM card's tel No , state and ISO could get info from the network type info. (Getting double card version default info).			**错误! 未定义书签。**
32	Set Enable SMS		setEnableSMS		**错误! 未定义书签。**
33	Set Enable Radio	When the radio is on off state, it could not call or send message.	setEnableRadio		**错误! 未定义书签。**
34	Delete SMS	Delete all tel No or related key words message from the in-box and outbox.	deleteSMS		54
35	Delete Contact	Delete all tel No through matching name and contact No.	deleteContact		54
36	Delete Call History	Delete all calls history.	deleteCallHistory		55
37	End Call	Remote end call is acceptable.	endCall		55
38	Set enable only emergency call		setEnableOnlyEmergencyCall		55
39	Set enable mobile data	Set enable mobile data	setEnableMobileData		56
	Equipment control API				56
40	Remote reboot terminal	Remote reboot is acceptable.	reboot		**错误! 未定义书签。**

41	Set Airplane On		setAirplaneOn		错误! 未定义书签。
42	Set Airplane Off		setAirplaneOff		57
43	Shut down	Remote shut down is acceptable.	shutDown		57
44	Lock screen	Lock screen, which used in device security control setting.	lockScreen		58
45	Unlock screen		unlockScreen		58
46	Set password	Remote setting password is acceptable.	setPassword		59
47	Factory reset	Remote factory resetting is acceptable, which used in device security control scene.	factoryReset		59
	Data wiping and backups API				59
48	Delete email	Delete uninstall email.	uninstallEmail		错误! 未定义书签。
49	Wipe App temporary data	Remote wiping App temporary data is acceptable.	wipeApp		错误! 未定义书签。
50	Wipe device data	Wipe devise data.	wipeData		错误! 未定义书签。
51	Remove video	Remove video to provide in the use of enterprise data security control scene.	removeVideo		61
52	Remove picture	Remove picture to provide in the use of enterprise data security control scene.	removePicture		61
53	Remove schedule	Remove schedule to provide in the use of enterprise data security control scene.	removeSchedule		62
	Copy and Paste API				错误! 未定义书签。

54	Set enable clip board	Set enable clip board.	setEnableClipboard		错误! 未定义书签。
55	Set enable capture screen	Set enable capture screen.	setEnableCaptureScreen		62
	Subclass GPS Control and Management API		Subclass MtmGPSManager		错误! 未定义书签。
56	GPS Control	Set enable GPS.	setEnableGps		错误! 未定义书签。
	Open interface control API	It refers to restrict the usage of the open interface terminal.			错误! 未定义书签。
57	Set iptables configuration management	Set iptables configuration management.	setEnableIptables		错误! 未定义书签。
	Peripheral Control API	It refers to the control of terminal peripheral.			64
	Subclass USB Management API		Subclass MtmUSBManager		错误! 未定义书签。
58	USB debug mode	Set enable USB Adb	setEnableUsbAdb		错误! 未定义书签。
59	USB Control	If the USB is forbidden, only keeps battery charges.	setEnableUsbFunction		65
60	USB-MTP Control	Set enable USB media stream transmission.	setEnableUsbMTP		错误! 未定义书签。
61	USB-PTP Control	Set enable USB picture transmission.	setEnableUsbPTP	*	错误! 未定义书签。
62	Set USB mass storage	Set enable USB mass storage.	setEnableUsbMassStorage	*	66
63	Set USB to swift network card control	Set USB to swift network card control	setEnableUsbRNDIS		66
64	USB network share	Open or off the USB network share is acceptable.	setEnableUsbTethering		67

	Storage Management API				67
65	Storage access control (SD or TF card)	Set enable SD card storage access.	setEnableSDCard		错误! 未定义书签。
66	Mount SD Card	Mount SD card (TF card), which used in device security control scene or system support and App access SD card.	mountSDCard		67
67	Unmount SD Card	Unmount SD card (TF card), system and App access is not allowed.	unmountSDCard		错误! 未定义书签。
68	Erase SD Card	Remote erase internal and extra-position SD card data to provide the use of enterprise data security scene.	eraseSDCard		68
	Camera Management Subclass API		Subclass MtmCameraManager		69
69	Set enable camera	Open or off the camera is acceptable, which provides mobile device security scene.	setEnableCamera		错误! 未定义书签。
	Recorder Management Subclass API		Subclass MtmRecorderManager		错误! 未定义书签。
70	Set Enable Microphone Mute	Set Enable Microphone Mute	setEnableMicrophoneMute		错误! 未定义书签。
71	Set Enable Audio Recorder	Set Enable Audio Recorder	setEnableAudioRecorder		70
	App Management and Control API	It refers to the control of the installation App and its function of self-starting and close, etc.			错误! 未定义书签。
	App Configuration API	It refers to the email and App configuration.			错误! 未定义书签。
72	Add Mail Config	It supports Exchange,	addMailConfig		错误!

		POP3 and IMAP accounts.			未定义书签。
73	Delete Mail Config	Delete Mail config	deleteMailConfig		71
74	Update Mail Config	It supports Exchange, POP3 and IMAP accounts.	updateMailConfig		71
75	Get Running App List	Get Running App List	getRunningAppList		72
76	Get App Install History		getAppInstallHistory		72
77	Start App	Remote start App	startApp		错误! 未定义书签。
78	End App	Remote endApp	endApp		错误! 未定义书签。
79	Set Enable App Auto Run	Set enable App auto Run	setEnableAppAutoRun		错误! 未定义书签。
	Native Browser Subclass API		Subclass MtmBrowserManager		74
80	**Browser access**	Set Enable Native Browser	setEnableNativeBrowser		错误! 未定义书签。
81	Browser cookie control		setEnableBrowserCookie		74
82	Browser javascript control		setEnableBrowserJavascript		75
83	Set Enable Browser Pop Window		setEnableBrowserPopWindow		错误! 未定义书签。
84	Set Enable Browser Auto Fill		setEnableBrowserAutoFill		错误! 未定义书签。
85	Browser https Security Warning		setEnableBrowserSafetyWarning		错误! 未定义书签。
86	URL access	By the restrictions of black and white list access.	setBrowserUrlBlockingMode		错误! 未定义书签。
87	Add Browser Bookmark	Remote add defined browser bookmark is	addBrowserBookmark		错误! 未定义

		acceptable.			书签。
	App installation, operation and removing subclass API		Subclass MtmAppManager		错误! 未定义书签。
88	Silent mode of installation App	Force and silent install specific App.	forceInstallApp		错误! 未定义书签。
89	Force Uninstall App	Force and silent uninstall specific App.	forceUninstallApp		错误! 未定义书签。
90	Set Enable App Installation	Enable the installation of SD/TF card's apk, adb installation is not allowed.	setEnableAppInstallation		78
91	Set Enable App Uninstallation	Uninstallation apk on setting -->Apps is not allowed. Adb installation is forbidden.	setEnableAppUninstallation		错误! 未定义书签。
92	Set Enable Kill App	Set enable kill App specific process or service.	setEnableKillApp		错误! 未定义书签。
93	Set Enable ADB	Set enable ADB installation ways of APK.	setEnableADB		错误! 未定义书签。
94	Set Enable Execute App	Set Enable Execute App	setEnableExecuteApp		错误! 未定义书签。
95	Erase App Data	Remote erase App data	eraseAppData		错误! 未定义书签。
96	Set Enable Recent App	Set enable to empty recent App.	setEnableRecentApp		错误! 未定义书签。
	App's black and white list API		Subclass MtmAppFilterManager		错误! 未定义书签。
97	Set App Running Block Mode	Set App running block mode, including black list, white list and normal mode.	setAppRunningBlockMode		错误! 未定义书签。
98	Set App Running Block Config	Running block configuration's content	setAppRunningBlockConfig		错误! 未定义

		includes black list and white list information. It could activate black and white list through setting App running block mode.			书签。
99	Set App Install Block Mode	Set App install block mode, including black list, white list and normal mode.	setAppInstallBlockMode		错误! 未定义书签。
100	Set App Install Block Config	Install block configuration's content includes black list and white list information. It could activate black and white list through setting App running block mode.	setAppInstallBlockConfig		错误! 未定义书签。
101	Force App Keep Running	Setting force specific App running list information to maintain the operation.	forceAppKeepRunning		错误! 未定义书签。
102	Uninstall App Black List		uninstallAppInBlackList		错误! 未定义书签。
103	Set Forbid Uninstallation List	It must to set forbid uninstallation list and to ensure the App could not be uninstalled.	setForbitUninstallationList		错误! 未定义书签。
104	Set URL Blocking Mode	Set URL blocking mode, including black list, white list and normal mode.	setURLBlockingMode		错误! 未定义书签。
105	Set URL Blocking Config	Setting URL blocking configuration's content includes black list and white list information. It could activate black and white list through setting URL blocking mode.	setURLBlockingConfig	*	错误! 未定义书签。

The definition of High-level API functions could be seen appendix B

The essential and optional API

In this standard, there are some functions frequently-used by enterprise management, some are rarely used. For the rarely used functions could be regarded as “optional” functions. In the table 1 EMCG-MTM-API Basic Functions and table 2 EMCG-MTM-API High-level Functions, remarks with “*” is “optional” functions, without “*” is essential functions.

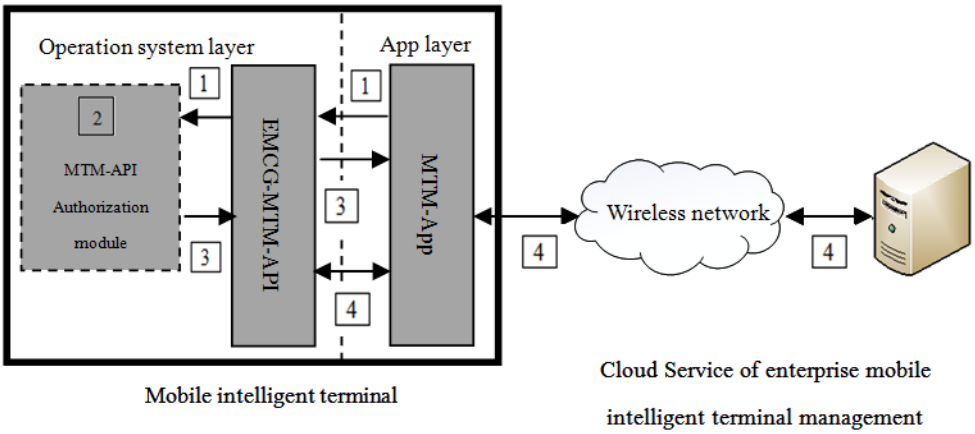
7 Authority Mechanisms

In order to ensure the security of mobile terminal and the validity of high-level API calls, the MTM-App can be used only if the authorization institude agrees.

7. 1 Authority Elements

1) High-level calls API’s MTM App is based on the standard of 《The regulation mechanisms of enterprise mobile terminal development, installation and operation (T/ZSA 3001.01-2016)》. MTM-App is signatured with EMCG attribute certificate.
2) Mobile intelligent terminal should have the function to verify EMCG-App validity, which inlude EMCG-App installation, operating the the signature verification mechanism.
3) Mobile intelligent terminal should preset the EMCG official certificate.
4) Mobile intelligent terminal should preset the MTM-App authorization module. (Authorization module can be an individule App or a system library function).

7. 2 Authority Mechanism



1	**Authorization Request** MTM-App sets the request to terminal system to call the Authenticate before called high-level API. (System transfers the MTM-App package name and the developer signature certificate to MTM-App authorization module.

Piture 2 MTM-App Authorization Process

Annex A

EMCG-MTM-API Basic Function Definition (Android OS)

Version	Writer	Time	Remarks
1.0.	Wang Ke, Yi Pingping, Liu Changshan, Songzhuo, Zhang Jianguo	2015-3-2015-4	Fisrt draft's design, discussion, writing and perfection.
1.1	Zhang Jianguo	2015-4-26	Fully integration
1.2	Wang Ke	2015-5-7	According to the content provided by Yi Pingping, Liu Changshan, Songzhuo, Zhang Jianguo
1.3	Liu Changshan, Hu Wenzhuan	2015-5-22	According to the content provided by Wang Ke

A. 1 Introduction

The appendix decrypts the basic functions program calls. Mobile manufacturers develop products and EMMS suppliers develop MTM-App according to the appendix to achieve unified management for the EMCG standard's hardware terminal device.

The basic API functions could get the related device information for enterprise intelligent management. Normally enterprise mobile management App can call the basic API functions without authorization.

There are total 34 calls functions.

A. 2 Android Calls Standard

Mobile manufacturers developing and offering calls to MTM-App based on the EMCG-MTM-API standard. EMCG-MTM-API is based on Android JAR package ways to deploy in Android's Framework layer. When EMMS developed MTM-App should pay attention to include EMCG-MTM-API's reference file, named: EMCG_MDM_SDK.jar (Case matters). Specific using method could be seen the reference of EMTM-App developed appendix.

EMCG-MTM-API definite the class management of MTM, named：EMCG_MTM_Manager. When initiated the function calls, the developers should make the class instantiation firstly. For example,

```
//Instantiated Object
EMCG_MTM_Manager    mtmManager = new EMCG_MTM_Manager();
…
// If there needs to call high level permission API, App should verify the permission. If it only calls basic API, there is no need to verify.
//retVal = mdmManager.authenticate();
…
// The specific methods of calling EMCG-MTM-API functions, such as forcing silent installation specific apk software.
int retVal = mtmManager.forceInstallApp("/SDCard/apks/qq.apk");
…
```

A. 3 Network configuration and management API

A.1.1 Subclass of Bluetooth API

```
Class MtmBluetoothManager
```

Class Description:

Bluetooth management subclass is responsible for the management various Bluetooth function of intelligent terminal device. MtmBluetoothManager is the subclass of EMCG_MTM_Manager.

Class MtmBluetoothDevice

Class Description:

Bluetooth device is used for description of the external Bluetooth device, which include Bluetooth device property, features and state, such as name,mac address and binding state, etc.

A. 3. 1. 1 Bluetooth State

int getState()

Function Description:

Getting Bluetooth state is the method of Mtm Bluetooth Manager.

Parameter Description:

- [Input]

  Nothing.

- [Return value]
  - BLUETOOTH_STATE_OFF
  - BLUETOOTH_STATE_TURNING_ON
  - BLUETOOTH_STATE_ON
  - BLUETOOTH_STATE_TURNING_OFF

A. 3. 1. 2 Bluetooth connection history records

Set<MtmBluetoothDevice> getBondedDevices()

Function Description:

Getting Bluetooth MAC history connected record information, used in device security control setting.

Parameter Description:

- [Input]

  Nothing.

- [Return value] Set<MtmBluetoothDevice>

  If the Bluetooth device information has already connected and occurred wrong information, it is null when returns.

Notice:

If Bluetooth is not BLUETOOTH_STATE_ON, API will return empty set.

A. 3. 1. 3 Get bluetooth MAC address information

String getAddress()

Function Description:

Getting bluetooth MAC address information, used in device security control setting

Parameter Description:

- [Input]

  Nothing.

- [Return value] String

  Bluetooth hardware MAC address, such as “00:11:22:AA:BB:CC”.

Notice:

Nothing.

A. 3. 1. 4 WiFi Subclass Managememt API

Class MtmWifiManager

Class Description:

WiFi subclass management is responsible for the management of various intelligent terminal device WiFi function. Mtm Wifi Manager is the subclass of EMCG_MTM_Manager.

A. 3. 2. 1 Get SSID Information

String getSSID()

Function Description:

Getting SSID (Wireless network name) information , provide identity authentication and device identification.

Parameter Description:

- [Input]

  Nothing.

- [Return value] String

  Current connected WiFi SSID information.

Notice:

It could receive the information only the WiFi connection is established or is establishing, or else it is null when returns.

A. 3. 2. 2 Wifi State

int getWifiState()

Function Description:

Get WiFi State

Parameter Description:

- [Input]

Nothing.

- [Return value]
  - WIFI_STATE_DISABLED
  - WIFI_STATE_DISABLING
  - WIFI_STATE_ENABLED
  - WIFI_STATE_ENABLING
  - WIFI_STATE_UNKNOWN

  WiFi unknown state occurs on the wrong process of open or close.

Notice:

Nothing.

A. 4 System Function Control API

It refers to the control of system functions which is provided by the terminal.

A. 5 Subclass of Telephone Information Management API

Class MtmTelephonyManager

Class Desciption:

Subclass of telephone information management API is responsible for the management of various telephone terminal device functions. Mtm Telephony Manager is the subclass of EMCG_MTM_Manager.

Telephone No information of SIM card, information state, ISO information could receive the information by type of network information.

A. 4. 1. 1 Get SIM card Telephone No Information

String getLine1Number()

Function Description:

Getting SIM card Tel No information, provide identity authentication and device identification. (Please noted that only parts of telephone No could receive the info, if the tel No data isn't loaded into SIM card, it could not received.)

Parameter Description:

- [Input]

  Nothing.

- [Return value] String

  SIM card's telephone number.

Notice:

Nothing.

A. 4. 1. 2 SIM card could get info by network type

int getNetworkType()

Function Description:

SIM card could get info by network type to calculate and audit.

Parameter Description:

- [Input]

  Nothing.

- [Return value]
  - NETWORK_TYPE_UNKNOWN
  - NETWORK_TYPE_GPRS
  - NETWORK_TYPE_EDGE
  - NETWORK_TYPE_UMTS
  - NETWORK_TYPE_HSDPA
  - NETWORK_TYPE_HSUPA
  - NETWORK_TYPE_HSPA
  - NETWORK_TYPE_CDMA
  - NETWORK_TYPE_EVDO_0
  - NETWORK_TYPE_EVDO_A
  - NETWORK_TYPE_EVDO_B
  - NETWORK_TYPE_1xRTT
  - NETWORK_TYPE_IDEN
  - NETWORK_TYPE_LTE
  - NETWORK_TYPE_EHRPD
  - NETWORK_TYPE_HSPAP

Notice:

Nothing.

A. 4. 1. 3 Get SIM Card's ISO Info

String getSimCountryIso()

Function Description:

Getting SIM card's info to calculate and audit.

Parameter Description:

- [Input]

  Nothing.

- [Return value] String

  SIM card's ISO（SIM card's country code of the operator）information.

Notice:

Nothing.

A. 4. 1. 4 Get SIM Card State

```
int getSimState()
```

Function Description:

It could not get SIM card's state info in the case of without card or unknown state, it will be needed pin, puk and network pin to unlock and recover well state.

Parameter Description:

- [Input]

  Nothing.

- [Return value]
  - SIM_STATE_UNKNOWN
  - SIM_STATE_ABSENT
  - SIM_STATE_PIN_REQUIRED
  - SIM_STATE_PUK_REQUIRED
  - SIM_STATE_NETWORK_LOCKED
  - SIM_STATE_READY
  - SIM_STATE_NOT_READY
  - SIM_STATE_PERM_DISABLED
  - SIM_STATE_CARD_IO_ERROR

Notice:

Nothing.

A. 4. 1. 5 Get SIM Operator Name

```
String getSimOperatorName()
```

Function Description:

Get SIM Operator Name

Parameter Description:

- [Input]

  Nothing.

- [Return value] String

  SIM card's operator name.

Notice:

SIM card must be on SIM_STATE_READY.

A. 4. 1. 6 Get Network Operator Name

```
String getNetworkOperatorName()
```

Function Description:

Get Network Administration Operator Name.

Parameter Description:

- [Input]

  Nothing.

- [Return value] String

  Current administration network operator name.

Notice:

It is valid for the users who have administered the network.

A.4.1.7 Get Network Operator Sequence Identity

```
String getNetworkOperator()
```

Function Description:

Get Current Administration Network Operator Sequence Identity（MCC+MNC）.

Parameter Description:

- [Input]

  Nothing.

- [Return value] String

Current Administration Network Operator Sequence Identity（MCC+MNC）

Notice:

It is valid for the users who have administered the network.

A.4.1.8 Get Network Roaming State Info

```
boolean isNetworkRoaming()
```

Function Description:

Getting network roaming state info to judge if it is roaming state. If yes, it could use the method of setting open/disable and data synchronization.

Parameter Description:

- [Input]

  Nothing.

- [Return value] boolean

  true：on Roaming State;

  false：on Non-roaming State.

Notice:

It is valid for the users who have administered the network.

A.4.1.9 Get Equipment Device API

A. 4. 2. 1 Get IMEI Info (Mobile Communication Equipment Identity)

String getIMEI()

Function Description:

Get IMEI Info (Mobile Communication Equipment Identity).

Notice: If it is without mobile communication PAD, there is without the identification.

Parameter Description:

- [Input]

  Nothing.

- [Return value] String

  Device Id information

Notice:

A. 4. 2. 2 Get ICCID Info (IC card's unique identification No)

String getICCID()

Function Description:

Getting ICCID Info (IC card's unique identification No), provide identity authentication and device identification.

Parameter Description:

- [Input]

  Nothing.

- [Return value] String

  ICCID Information.

Notice:

Nothing.

A. 4. 2. 3 Get IMSI (SIM card's unique identification)

String getIMSI()

Function Description:

Get IMSI device identification information.

Parameter Description:

- [Input]

  Nothing.

- [Return value] String

Including IMSI Info.

Notice:

Nothing.

A. 4. 2. 4 Get Mac Address Info

```
String getMacAddress()
```

Function Description:

Getting Mac address info, provide identity authentication and device identification.

Parameter Description:

- [Input]

  Nothing.

- [Return value] String

  MAC Address Info.

Notice:

Nothing.

A. 4. 2. 5 Get IP Address

```
String getIpAddress()
```

Function Description:

Getting Mac address info, provide identity authentication and device identification.

Parameter Description:

- [Input]

  Nothing.

- [Return value] String

  IP Address Info.

Notice:

Nothing.

A. 4. 2. 6 Get Device Serial Number

```
String getDeviceSerialNumber()
```

Function Description:

Getting Device Serial Number

Parameter Description:

- [Input]

  Nothing.

- [Return value] String

  Device Serial Number Info.

Notice:

Nothing.

A. 4. 2. 7 Get ROM Version

```
String getRomVersion()
```

Function Description:

Getting ROM version, used in device management setting.

Parameter Description:

- [Input]

  Nothing.

- [Return value] String

  ROM version info.

Notice:

Nothing.

A. 4. 2. 8 Get CPU usage rate, number, frequency and item No info

```
String getCpuInfo()
```

Function Description:

Getting CPU usage rate, number, frequency and item No info, used in device security control setting.

Parameter Description:

- [Input]

  Nothing.

- [Return value] String

  CPU usage rate, number, frequency and item No info.

Notice:

Nothing.

A. 4. 2. 9 Get Battery Level

```
String getBatteryLevel()
```

Function Description:

Getting battery level, used in device security control setting.

Parameter Description:

- [Input]

  Nothing.

- [Return value] String

  Battery level info.

Notice:

Nothing.

A. 4. 2. 10 Get Memory Info

String getMemoryInfo()

Function Description:

Get total memory capacity info and available info, used in device security control setting.

Parameter Description:

- [Input]

  Nothing.

- [Return value] String

Total memory capacity info and available info.

Notice:

Nothing.

A. 4. 2. 11 Get Internal Storage Info

String getInternalStorageInfo()

Function Description:

Going back inside memory capacity and available space.

Parameter Description:

- [Input]

  Nothing.

- [Return value] String

Going back inside memory capacity and available space.

Notice:

Used space can be calculated.

A. 4. 2. 12 Get SD Card Info

String getSDCardInfo()

Function Description:

Going back outside memory capacity and available space.

Parameter Description:

- [Input]

  Nothing.

- [Return value] String

Going back outside memory capacity and available space.

Notice:

Used space can be calculated.

A. 4. 2. 13 Get OS version

```
String getOSVersion()
```

Function Description:

Getting OS version, used in device management setting.

Parameter Description:

- [Input]

  Nothing.

- [Return value] String

  System Device Info.

Notice:

Nothing.

A. 4. 2. 14 Device Model

```
String getDeviceModel()
```

Function Description:

Getting device model, used in device management setting.

Parameter Description:

- [Input]

  Nothing.

- [Return value] String

  Device Model Property Info.

Notice:

Nothing.

A. 4. 1 Get GPS Info API

A. 4. 3. 1 Get GPS current location

```
String getCurrentLocation()
```

Function Description:

Getting current device location info

Parameter Description:

- [Input] String provider

  Input's provider。

- [Return value] Location

  Return back to user's location info.

Notice:

Nothing.

A. 4. 3. 2 Get GPS Last Known Location

```
String getLastKnownLocation()
```

Function Description:

Get current user's location info, such as latitude and longitude and time.

Parameter Description:

- [Input] String provider

  Input's provider。

- [Return value] Location

Return back user's location info, including latitude and longitude

Notice:

Nothing.

A. 6 Get Peripheral Info API

A. 5. 1 Get Number Of Cameras

```
int getNumberOfCameras()
```

Function Description:

Getting No of cameras, used in device management setting.

Parameter Description:

- [Input]

  Nothing.

- [Return value] int

  Return back to cameras number.

Notice:

Nothing.

A. 7 Get APP Info API

A. 6. 1 Get App Data Traffic

```
String getAppDataTrafficUsage(String appID);
```

Function Description:

Get the specific package App data traffic usage.

Parameter Description:

- [Input]String appID

  Input App package name.

- [Return value] String

  Return back App data traffic usage info from initial date to now. The specific format can be seen the attached.

Notice:

Nothing.

A. 6. 2 Get Installed App list

```
int getInstalledAppList(String []returnNames);
```

Function Description:

Get current device installed App package list, such as com.tencent.qq, com.baidu.map and so on.

Parameter Description:

- [Output]String [] returnNames

Return to an array of strings, which includes multi-App package name character string.

- [Return value]
  - ERRORCODE_SUCCESS

  Executive correct;
  - Other values

  If it is error, please see the chapter of “error code definition”.

Notice:

Nothing.

A. 6. 3 Get specific package App Info

```
String getAppInfo(String appID);
```

Function Description:

Getting specific installed App info, including program name, version,size, time length and start-up times, used in device management setting.

Parameter Description:

- [Input]String appID

  Input App program package name.
- [Return value] String

  Return to installed App specific package name info and AML structure, please see appendix format.

Notice:

Nothing.

A. 8 High level API calls authorization

```
int authenticate()
```

Function Description:

Terminal authorization MTM-App calls high-level API.

Parameter Description:

- [Input]

  Nothing.

- [Return value]
  - ERRORCODE_SUCCESS
  - ERRORCODE_INVALIDSIGNATURE

Notice:

Nothing.

A.9 Error Code Definition

Functions calls return back error code definition table

Error code definition	Value number	Meaning
ERRCODE_SUCCESS	0	Function operation is correct.
ERRCODE_UNKNOWN	-1	Unknown Exception Error
ERRCODE_EXECUTEERR	-2	Executive Error
ERRCODE_PERMISSIONERR	-3	Permission Error
ERRCODE_OBJECTNOTFOUNDERR	-4	Object is not found. (Such as SIM card, WIFI device and Bluetooth, etc.)
ERRORCODE_INVALIDSIGNATURE	-11	APP signature certificate is invalid.

Annex B

EMCG-MTM-API High-level Functions Definition (Android OS)

Version	Writer	Time	Remarks
1.0.	Wang Ke, Yi Pingping, Liu Changshan, Songzhuo, Zhang Jianguo	2015-3-2015-4	Fisrt draft's design, discussion, writing and perfection.
1.1	Zhang Jianguo	2015-4-26	Fully integration
1.2	Wang Ke	2015-5-7	According to the content provided by Yi Pingping, Liu Changshan, Songzhuo, Zhang Jianguo
1.3	Liu Changshan, Hu Wenzhuan, Zhang Jianguo	2015-6-14	According to the content provided by Wang Ke
1.4	Wang Ke	2015-6-21	Add Jin Yanwei's suggestion functions: 4.3.7 lock screen Password modification.
1.5	Wang Ke	2015-6-23	According to the content provided by Liu Changshan and Zhang Jianguo.

B. 1 Introduction

The appendix decrypts the High-level functions program calls. Mobile manufacturers develop products and EMMS suppliers develop MTM-App according to the appendix to achieve unified management, integrated and deep management for the EMCG standard's hardware terminal device.

High-level calls API must be signed by EMCG-App and needed official institute authorization before calls.

There are total 105 API calls functions.

B. 2 Android Calls Standard

EMCG-MTM-API is based on Android JAR package ways to deploy in Android's Framework layer. When EMMS developed MTM-App should pay attention to include EMCG-MTM-API's reference file, named: EMCG_MDM_SDK.jar (Case matters). Specific using method could be seen the reference of EMTM-App developed appendix.

EMCG-MTM-API definite the class management of MTM, named：EMCG_MTM_Manager. When initiated the function calls, the developers should make the class instantiation firstly. For example,

```
// Instantiated Object
EMCG_MTM_Manager    mtmManager = new EMCG_MTM_Manager();
…
// If there needs to call high level permission API, App should verify the permission.
retVal = mdmManager.authenticate();
…
// The specific methods of calling EMCG-MTM-API functions, such as forcing silent installation specific apk software.
int retVal = mtmManager.forceInstallApp("/SDCard/apks/qq.apk");
…
```

B. 3 Network configuration and management API

B. 3. 1 APN Subclass Management API

```
Class MtmAPNManager
```

Class Description:

APN management subclass is responsible for the management various APN function of intelligent terminal device. Mtm APN Manager is the subclass of EMCG_MTM_Manager.

B. 3. 1. 1 Query Specific APN

```
List<String> queryApn()
```

Function Description:

Query specific APN

Parameter Description:

- [Input] Map<String, String>apnparam

  Query APN's required matching key parameter, it could be one or more key parameters, which includes name, apn, mcc, mnc, numeric, user, password, server, proxy, port, mmsport, msproxy, mmsc, authtype, current and type.

- [Return value]
  - Return back the query APN's ID list.

  Executive Correct;

  - null

  Could not be found.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 3. 1. 2 Add APN

```
String addApn()
```

Function Description:

Add APN.

Parameter Description:

- [Input] Map<String, String>apnparam

  APN configuration parameter mainly includes name, apn, mcc, mnc, numeric, user, password, server, proxy, port, mmsport, mmsproxy, mmsc, authtype, current and type.

- [Return value]
  - id

  Return back the Add APN's ID identity if the execution is correct.

  - Other value

  If it is error, please see the chapter of "error code definition".

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 3. 1. 3 Delete Specific VPN

```
int deleteApn()
```

Function Description:

Delete Specific VPN

Parameter Description:

- [Input] String id

  Delete the required apn identification, which could be gotten from query apn.

- [Return value]
  - ERRORCODE_SUCCESS

    Excutive correct.

  - Other value

    If it is error, please see the chapter of "error code definition".

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 3. 1. 4 Update Specific VPN Configuration

```
int updateApn()
```

Function Description:

Update APN。

Parameter Description:

- [Input] String id

  Update the required apn identification, which could be gotten from query apn.

- [Input] Map<String, String> apnparam

  Update APN's configuration parameter, which includes name, apn, mcc, mnc, numeric, user, password, server, proxy, port, mmsport, msproxy, mmsc, authtype, current and type.

- [Return value]
  - ERRORCODE_SUCCESS

    Excutive Correct

  - Other value

    If it is error, please see the chapter of "error code definition".

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 3. 1. 5 Set Prefer APN

```
int setPreferApn()
```

Function Description:

Set Prefer APN

Parameter Description:

- [Input] String id

  Set the required apn identification, which could be gotten from query apn.

- [Return value]
  - ERRORCODE_SUCCESS

    Executive Correct.

➢ Other value

If it is error, please see the chapter of "error code definition".

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 3. 2 VPN Management API

Add, delete and update VPN, connect and disconnect VPN; could set parameter automatic, which used in device security control setting.

B. 3. 2. 1 Query VPN

```
List<String> queryVpn()
```

Function Description:

Query VPN。

Parameter Description:

- [Input] Map<String, String>vpnparam
  - Query APN's required matching key parameter, it could be one or more key parameters, which could be seen the Android source Vpn Profile; if vpnparam is null, it will return vpn list.
  - [Return value]

➢ Return to the query VPN's id list.

Executive Correct.

➢ null

Could not be found.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 3. 2. 2 Add VPN

```
String addVpn()
```

Function Description:

VPN parameter could be seen the Android source Vpn Profile.

Parameter Description:

- [Input] Map<String, String>vpnparam

  VPN parameter could be seen the Android source Vpn Profile.
- [Return value]

➢ id

If executive correct, it will return all vpn list;

➢ Other value

If it is error, please see the chapter of "error code definition".

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 3. 2. 3 Delete Specific VPN

```
int deleteVpn()
```

Function Description:

Delete Specific VPN

Parameter Description:

- [Input] String id

  Delete the required vpn identification, which could be gotten from query vpn.

- [Return value]
  - ERRORCODE_SUCCESS

    Executive Correct;

  - Other value

    If it is error, please see the chapter of "error code definition".

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 3. 2. 4 Update Specific VPN Configuration

```
int updateVpn()
```

Function Description:

Update Specific VPN Configuration.

Parameter Description:

- [Input] String id

  Update the required vpn identification, which could be gotten from query vpn.

- [Input] Map<String, String> vpnparam

  VPN parameter could be seen the Android source Vpn Profile.

- [Return value]
  - ERRORCODE_SUCCESS

    Executive Correct

  - Other value

    If it is error, please see the chapter of "error code definition".

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 3. 2. 5 Connect Specific VPN

```
int connectVpn()
```

Function Description:

Connect Specific VPN

Parameter Description:

- [Input] String id

  Connected the required vpn identification, which could be gotten from query vpn.

- [Return value]
  - ERRORCODE_SUCCESS

    Executive Correct;

  - Other value

    If it is error, please see the chapter of "error code definition".

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 3. 2. 6 Disconnect VPN

```
int disConnectVpn()
```

Function Description:

Disconnect Specific VPN

Parameter Description:

- [Input]

  Nothing.

- [Return value]
  - ERRORCODE_SUCCESS

    Executive Correct;

  - Other value

    If it is error, please see the chapter of "error code definition".

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 3. 3 NFC Management Subclass API

- Class MtmNFSManager

Class Description:

NFS management subclass is responsible for the management various NFS function of intelligent terminal device. Mtm NFS Manager is the subclass of EMCG_MTM_Manager.

B. 3. 3. 1 Set Enable NFC

```
int setEnableNFC()
```

Function Description:

Set enable to transmit NFC data at short range.

Parameter Description:

- [Input]boolean enable

  true：allowed

  false：not allowed

- [Return value]
  - ERRORCODE_SUCCESS

    Executive Correct;

  - Other value

    If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 3. 4 Bluetooth Management Subclass API

Class MtmBluetoothManager

Class Description:

Bluetooth management subclass is responsible for the management various Bluetooth function of intelligent terminal device. Mtm Bluetooth Manager is the subclass of EMCG_MTM_Manager.

B. 3. 4. 1 Only Enable Specific type of Bluetooth

int setWhiteBluetoothType()

Function Description:

Only enable specific type of Bluetooth, such as Bluetooth equipment, audio and video equipment connected with PC.

Parameter Description:

- [Input] List<int>bluetoothclass

  Only enable specific type of communication Bluetooth device, its type could be seen Bluetooth Class Device constants.

- [Return value]
  - ERRORCODE_SUCCESS

    Executive Correct;

  - Other value

    If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 3. 4. 2 Set Enable Bluetooth

int setEnableBluetooth()

Function Description:

Set Enable Bluetooth
Parameter Description:

- [Input]boolean enable
  true：allowed;
  false：not allowed.
- [Return value]
  - ERRORCODE_SUCCESS
    executive correct;

➢ Other value

If it is error, please see the chapter of "error code definition".

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 3. 4. 3 Set Enable Bluetooth Share Function

int setEnableBluetoothShare()

Function Description:

Set Enable Bluetooth Share Function
Parameter Description:

- [Input]boolean enable
  true：allowed;
  false：not allowed.
- [Return value]
  - ERRORCODE_SUCCESS
    executive correct;

➢ Other value

If it is error, please see the chapter of "error code definition".

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 3. 5 WiFi Subclass Management API

Class MtmWifiManager

Class Description:

WiFi management subclass is responsible for the management various WiFi function of intelligent terminal device. Mtm WiFi Manager is the subclass of EMCG_MTM_Manager.

Class MtmWifiConfiguration

Class Description:

WiFi network configuration is used to describe WiFi network info, which includes SSID, Authentication Protocol, State Info and WiFi network operation system unique NetID, etc.

B. 3. 5. 1 WiFi Connected History Record

```
List<MtmWifiConfiguration> getConfiguredNetworks()
```

Function Description:

Get WiFi Connected History Record.

Parameter Description:

- [Input]

  Nothing.

- [Return value]List<MtmWifiConfiguration>

  WiFi connected history; once WiFi is on off or failure state, it is null when returns.

Notice:

Nothing.

B. 3. 5. 2 Set Enable WiFi

```
int setEnableWifi()
```

Function Description:

Set Enable WiFi

Parameter Description:

- [Input]boolean enable

  true：allowed;

  false：not allowed.

- [return value]
  - ERRORCODE_SUCCESS

    Executive correct

  - Other value

    If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 3. 5. 3 Set Enable WiFi Share

```
int setEnableWifiShare()
```

Function Description:

Set Enable WiFi Share function.

Parameter Description:

- [Input]boolean enable

  true：allowed;

  false：not allowed.

- [return value]
  - ERRORCODE_SUCCESS

Executive correct

- Other value

If it is error, please see the chapter of "error code definition".

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 3. 5. 4 Add WiFi Network

```
int addNetwork()
```

Function Description:

Add WiFi Network

Parameter Description:

- [Input]MtmWifiConfiguration config
- [Return value]
  - netId

  If it is correct, it will return back WiFi network configuration id;
  - Other value

  If it is error, please see the chapter of "error code definition".

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 3. 5. 5 Remove WiFi Network

```
int removeNetwork()
```

Function Description:

Remove Specific WiFi Network

Parameter Description:

- [Input]int netId

  Remove the required WiFi network configuration id. NetID is the unique identity on Wifi network operation system and it belongs one of fields of MtmWifiConfiguration. When the Wifi network is added and updated, it will return back netid. Also it can get the configuration Wifi network list through getting ConfiguredNetworks.

- [Return value]
  ERRORCODE_SUCCESS

  Executive correct
  - Other value

  If it is error, please see the chapter of "error code definition".

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 3. 5. 6 Update WiFi Network

```
int updateNetwork()
```

Function Description:

Update Specific WiFi Network

Parameter Description:

- [Input]MtmWifiConfiguration config

Update required WiFi network configuration.

- [Return value]
  - ➢ netId

    Executive correct, WiFi network configuration id；

  - ➢ Other value

    If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 3. 5. 7 Set WiFi Network Priority

```
int enableNetwork()
```

Function Description:

Set WiFi network priority

Parameter Description:

- [Input]int netId

  Set specific connected Wifi network identity.
- [Input]boolean disableOthers

  Forbid another Wifi network or not.

  True means to forbid another Wifi networks, only setting specific Wifi connected priority;

  False means connected with another Wifi is available.
- [Return value]
  - ➢ ERRORCODE_SUCCESS

    Executive correct;

  - ➢ Other value

    If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 3. 5. 8 Disable WiFi Network

```
int disableNetwork()
```

Function Description:

Disable specific WiFi Network.

Parameter Description:

- [Input]int netId

Disable specific WiFi Network identity.

- [Return value]
  - ERRORCODE_SUCCESS

    Executive correct;

  - Other value

    If it is error, please see the chapter of "error code definition".

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 3. 5. 9 Disconnect WiFi Network

```
int disconnect()
```

Function Description:

Disconnect WiFi Network

Parameter Description:

- [Input]

  Nothing.

- [Return value]
  - ERRORCODE_SUCCESS

    Executive correct；

  - Other value

    If it is error, please see the chapter of "error code definition".

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 3. 5. 10 Reconnect WiFi Network

```
int reconnect()
```

Function Description:

Reconnect current active AP WiFi Network.

Parameter Description:

- [Input]

  Nothing.

- [Return value]

➢ ERRORCODE_SUCCESS

Executive correct；

➢ Other value

If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 4 System Function Control API

B. 4. 1 System Enable API

B. 4. 2 Set Enable Home Key

```
int setEnableHomeKey()
```

Function Description:

The scene of mobile marketing, mobile classroom are not allowed to swift the Home Key.（Android 5.0 version supports App block.）

Parameter Description:

- [Input]boolean enable

  true：allowed；

  false：not allowed.

- [Return value]

➢ ERRORCODE_SUCCESS

Executive correct；

➢ Other value

If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 4. 1. 1 Set Enable Back Key

```
int setEnableBackKey()
```

Function Description:

The scene of mobile marketing, mobile classroom are not allowed to swift the Back Key.（Android 5.0 version supports App block.）

Parameter Description:

- [[Input]boolean enable

  true：allowed；

  false：not allowed.

- [Return value]

➢ ERRORCODE_SUCCESS

Executive correct；

➢ Other value

If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 4. 1. 2 It appoints which package name are single App models

```
int setKioskApp()
```

Function Description:

It only allows specific App and exit the current interface is forbidden. It could achieve single App mode by home key, back key or App black.

Parameter Description:

- [Input] String pkgName

  Single App mode list.

- [Return value]

➢ ERRORCODE_SUCCESS

Executive correct；

➢ Other value

If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 4. 1. 3 Set Date Time

```
int setDataTime()
```

Function Description:

Set Date Time（CDMA terminal default network synchronization is not included.）

Parameter Description:

- [Input] String year

  Year

- [Input] String month

  Month

- [Input] String date

  Date

- [Input] String hour

  Hour

- [Input] String min

Minutes

- [Input] String sec

  Second

- [Return value]

  ➢ ERRORCODE_SUCCESS

  Executive correct；

  ➢ Other value

  If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 4. 1. 4 Set Default Launcher

```
int setDefaultLauncher()
```

Function Description:

Set Default Launcher。

Parameter Description:

- [Input] String pkg

  Launcher App’s package name；

- [Input] String className

  Launcher App’s Launcher interface activity corresponding class name.

- [Return value]

  ➢ ERRORCODE_SUCCESS

  Executive correct；

  ➢ Other value

  If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 4. 1. 5 Hide Navigation Bar

```
int hideNavigationBar()
```

Function Description:

Hide and display navigation bar.

Parameter Description:

- [Input]boolean enable

  true：hide the navigation bar;

  false：display the navigation bar.

- [Return value]
  - ERRORCODE_SUCCESS

    Executive correct；

  - Other value

    If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 4. 3 Communication Management AP

B. 4. 2. 1 Set Enable SMS

```
int setEnableSMS()
```

Function Description:

Set Enable function to receive and send SMS.

Parameter Description:

- [Input]boolean enable

  true：allowed;

  false：not allowed.

- [Return value]
  - ERRORCODE_SUCCESS

    Executive correct；

  - Other value

    If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 4. 2. 2 Set Enable Radio

```
int setEnableRadio()
```

Function Description:

When the radio is on off state, it could not call or send message.

Parameter Description:

- [Input]boolean enable

  true：On；

  false：Off.

- [Return value]
  - ERRORCODE_SUCCESS

    Executive correct；

➢ Other value

If it is error, please see the chapter of "error code definition".

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 4. 2. 3 Delete SMS

```
int deleteSMS()
```

Function Description:

Delete all tel No or related key words message from the in-box and outbox.

Parameter Description:

- [Input] String address

  Contact Tel No

- [Input] String bodyKey

  Message content's key words.

  If the address and bodyKey are not null, they could be matched; if address is null, bodyKey is not null, it will match bodyKey; if bodyKey is null, it matches address; if they are both null, it will delete all messages.

- [Return value]

➢ ERRORCODE_SUCCESS

Executive correct；

➢ Other value

If it is error, please see the chapter of "error code definition".

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 4. 2. 4 Delete Contact

```
int deleteContact()
```

Function Description:

Delete all tel No through matching name and contact No.

Parameter Description:

- [Input] String name

Contact's name.

- [Input] String phone

Contact's telephone number.

If the name and phone are not null, they could be matched; if name is null, phone is not null, it will match phone; if phone is null, it matches name; if they are both null, it will delete all contacts.

- [Return value]
  - ERRORCODE_SUCCESS

    Executive correct;

  - Other value

    If it is error, please see the chapter of "error code definition".

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 4. 2. 5 Delete Call History

```
int deleteCallHistory()
```

Function Description:

Delete all calls history.

Parameter Description:

- [Input]

  Nothing.

- [Return value]
  - ERRORCODE_SUCCESS

    Executive correct;

  - Other value

    If it is error, please see the chapter of "error code definition".

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 4. 2. 6 End Call

```
int endCall()
```

Function Description:

End call.

Parameter Description:

- [Input]

  Nothing.

- [Return value]
  - ERRORCODE_SUCCESS

    Executive correct;

  - Other value

    If it is error, please see the chapter of "error code definition".

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 4. 2. 7 Set enable only emergency call

```
int setEnableOnlyEmergencyCall()
```

Function Description:

Set enable only emergency call

Parameter Description:

- [Input]boolean enable

  true：Limit calls, only emergency calls are allowed；

  false：Without limit calls.

- [Return value]

  ➢ ERRORCODE_SUCCESS

  Executive correct；

  ➢ Other value

  If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 4. 2. 8 Set enable mobile data

```
int setEnableMobileData()
```

Function Description:

Set enable mobile data

Parameter Description:

- [Input]boolean enable

  true：allowed；

  false：not allowed.

- [Return value]

  ➢ ERRORCODE_SUCCESS

  Executive correct；

  ➢ Other value

  If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 4. 4 Equipment control API

B. 4. 3. 1 Remote reboot terminal

```
int reboot()
```

Function Description:

Remote reboot is acceptable.

Parameter Description:

- [Input]

  Nothing.

- [Return value]
  - ERRORCODE_SUCCESS

  Executive correct；

  - Other value

  If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 4. 3. 2 Set Airplane On

```
int setAirplaneOn()
```

Function Description:

Remote set Airplane on is acceptable.

Parameter Description:

- [Input]

  Nothing.

- [Return value]
  - ERRORCODE_SUCCESS

  Executive correct；

  - Other value

  If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 4. 3. 3 Set Airplane Off

```
int setAirplaneOff()
```

Function Description:

Remote set Airplane off is acceptable.

Parameter Description:

- [Input]

  Nothing.

- [Return value]
  - ERRORCODE_SUCCESS

  Executive correct；

➢ Other value

If it is error, please see the chapter of "error code definition".

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 4. 3. 4 Shut down

```
int shutDown()
```

Function Description:

Remote shut down is acceptable.

Parameter Description:

- [Input]

  Nothing.

- [Return value]

  ➢ ERRORCODE_SUCCESS

  Executive correct；

  ➢ Other value

  If it is error, please see the chapter of "error code definition".

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 4. 3. 5 Lock screen

```
int lockScreen()
```

Function Description:

Lock screen, which used in device security control setting.

Parameter Description:

- [Input]

  Nothing.

- [Return value]

  ➢ ERRORCODE_SUCCESS

  Executive correct；

  ➢ Other value

  If it is error, please see the chapter of "error code definition".

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 4. 3. 6 Unlock screen

```
int unlockScreen()
```

Function Description:

Unlock screen, which used in device security control setting.

Parameter Description:

- [Input]

  Nothing.

- [Return value]
  - ERRORCODE_SUCCESS

  Executive correct；

  - Other value

  If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 4. 3. 7 Set password

```
int setPassword()
```

Function Description:

Remote setting password is acceptable.

Parameter Description:

- [Input] String newPassword

  New password string.

- [Return value]
  - ERRORCODE_SUCCESS

  Executive correct；

  - Other value

  If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 4. 3. 8 Factory reset

```
int factoryReset()
```

Function Description:

Remote factory resetting is acceptable, which used in device security control scene.

Parameter Description:

- [Input] int flags

  0：wipe user’s data；

  1：do not wipe user’s data.

- [Return value]

➢ ERRORCODE_SUCCESS

Executive correct；

➢ Other value

If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 4. 5 Data wiping and backups API

B. 4. 4. 1 Delete email

```
int uninstallEmail()
```

Function Description:

Delete uninstall email.

Parameter Description:

- [Input]

  Nothing.

- [Return value]

➢ ERRORCODE_SUCCESS

Executive correct；

➢ Other value

If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 4. 4. 2 Wipe App temporary data

```
int wipeApp()
```

Function Description:

Remote wiping App temporary data is acceptable.

Parameter Description:

- [Input] String pName

  path+file-name.

- [Return value]

➢ ERRORCODE_SUCCESS

Executive correct；

➢ Other value

If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 4. 4. 3 Wipe device data

```
void wipeData()
```

Function Description:

Wipe device data

Parameter Description:

- [Input] int flags

  0： wipe inside data；

  1： wipe outside data-SD card.

- [Return value]

  ➢ ERRORCODE_SUCCESS

  Executive correct；

  ➢ Other value

  If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 4. 4. 4 Remove video

```
int removeVideo()
```

Function Description:

Remove video to provide in the use of enterprise data security control scene.

Parameter Description:

- [Input] String pName

  Path+file-name. (Video_Remove_All, refers to remove all video.)

- [Return value]

  ➢ ERRORCODE_SUCCESS

  Executive correct；

  ➢ Other value

  If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 4. 4. 5 Remove picture

```
int removePicture()
```

Function Description:

Remove picture to provide in the use of enterprise data security control scene.

Parameter Description:

- [Input] String pName

Path+file-name.(Picture_Remove_All, refers to remove all pictures.)

- [Return value]
  - ERRORCODE_SUCCESS

  Executive correct；
  - Other value

  If it is error, please see the chapter of "error code definition".

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 4. 4. 6 Remove schedule

```
int removeSchedule()
```

Function Description:

Remove schedule to provide in the use of enterprise data security control scene.

Parameter Description:

- [Input] String pName

  Path+file-name.(Schedule_Remove_All, refers to remove all schedules.)
- [Return value]
  - ERRORCODE_SUCCESS

  Executive correct；
  - Other value

  If it is error, please see the chapter of "error code definition".

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 4. 6 Copy and Paste API

B. 4. 5. 1 Set enable clip board

```
int setEnableClipboard()
```

Function Description:

Set enable clip board.

Parameter Description:

- [Input]boolean enable

  true：allowed；

  false：not allowed。
- [Return value]
  - ERRORCODE_SUCCESS

  Executive correct；

➢ Other value

If it is error, please see the chapter of "error code definition".

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 4. 5. 2 Set enable capture screen

int setEnableCaptureScreen()

Function Description:

Set enable capture screen function.

Parameter Description:

- [Input]boolean enable

  true：allowed；

  false：not allowed。

- [Return value]

➢ ERRORCODE_SUCCESS

Executive correct；

➢ Other value

If it is error, please see the chapter of "error code definition".

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 4. 7 Subclass GPS Control and Management API

Class MtmGPSManager

Function Description:

GPS control and management subclass is responsible for the management various GPS functions of intelligent terminal device. MtmGPSManager is the subclass of EMCG_MTM_Manager.

B. 4. 6. 1 GPS Control

int setEnableGps()

Function Description:

Set enable GPS.

Parameter Description:

- [Input]boolean enable

  true：allowed；

  false：not allowed。

- [Return value]

➢ ERRORCODE_SUCCESS

Executive correct；

➢ Other value

If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 4. 8 Open interface control API

B. 4. 7. 1 Set iptables configuration management

int setEnableIptables ()

Function Description:

Set iptables configuration management.

Parameter Description:

- [Input]boolean enable

  true：allowed；

  false：not allowed。

- [Return value]

➢ ERRORCODE_SUCCESS

Executive correct；

➢ Other value

If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 5 Peripheral Control API

B. 5. 1 Subclass USB Management API

Class MtmUSBManager

Class Description:

USB management subclass is responsible for the management various USB functions of intelligent terminal device. MtmUSBManager is the subclass of EMCG_MTM_Manager.

B. 5. 1. 1 USB debug mode

int setEnableUsbAdb()

Function Description:

Set enable USB Adb

Parameter Description:

- [Input]boolean enable

  true：On;

  false：Off.

- [Return value]
  - ERRORCODE_SUCCESS

  Executive correct；

  - Other value

  If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 5. 1. 2 USB Control

```
int setEnableUsbFunction()
```

Function Description:

If the USB communication is forbidden, only keeps battery charges.

Parameter Description:

- [Input]boolean enable

  true：allowed；

  false：not allowed。

- [Return value]
  - ERRORCODE_SUCCESS

  Executive correct；

  - Other value

  If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 5. 1. 3 USB-MTP Control

```
int setEnableUsbMTP()
```

Function Description:

Set enable USB media stream transmission.

Parameter Description:

- [Input]boolean enable

  true：allowed；

  false：not allowed。

- [Return value]
  - ERRORCODE_SUCCESS

  Executive correct；

➢ Other value

If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 5. 1. 4 USB-PTP Control

```
int setEnableUsbPTP()
```

Function Description:

Set enable USB picture transmission.

Parameter Description:

- [Input]boolean enable

  true：allowed；

  false：not allowed。

- [Return value]

➢ ERRORCODE_SUCCESS

Executive correct；

➢ Other value

If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 5. 1. 5 Set USB mass storage

```
int setEnableUsbMassStorage()
```

Function Description:

Set enable USB mass storage.

Parameter Description:

- [Input]boolean enable

  true：allowed；

  false：not allowed。

- [Return value]

➢ ERRORCODE_SUCCESS

Executive correct；

➢ Other value

If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 5. 1. 6 Set USB to swift network card control

```
int setEnableUsbRNDIS()
```

Function Description:

Set USB to swift network card control

Parameter Description:

- [Input]boolean enable

  true：allowed；

  false：not allowed。

- [Return value]
  - ERRORCODE_SUCCESS

    Executive correct；

  - Other value

    If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 5. 1. 7 USB network share

```
int setEnableUsbTethering()
```

Function Description:

Open or off the USB network share is acceptable.

Parameter Description:

- [Input]boolean enable

  true：On；

  false：Off.

- [Return value]
  - ERRORCODE_SUCCESS

    Executive correct；

  - Other value

    If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 5. 2 Storage Management API

B. 5. 2. 1 Storage access control (SD or TF card)

```
int setEnableSDCard ();
```

Function Description:

Set enable SD card storage access.

Parameter Description:

- [Input]boolean enable

  true：allowed；

  false：not allowed。

- [Return value]
  - ERRORCODE_SUCCESS

  Executive correct；

  - Other value

  If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 5. 2. 2 Mount SD Card

```
int mountSDCard ();
```

Function Description:

Mount SD card (TF card), which used in device security control scene or system support and App access SD card.

Parameter Description:

- [Input]

  Nothing.

- [Return value]
  - ERRORCODE_SUCCESS

Executive correct；

- Other value

If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 5. 2. 3 Mount SD Card

```
int umountSDCard ();
```

Function Description:

Mounting SD card (TF card) is used in device security control scene or system support & App access SD card.

Parameter Description:

- [Input]

  Nothing.

- [Return value]

➢ ERRORCODE_SUCCESS

Executive correct；

➢ Other value

If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 5. 2. 4 Unmount SD Card

```
int eraseSDCard();
```

Function Description:

Unmount all SD card’s data, which could unmount specific inside or outside storage.

Parameter Description:

- [Input] int flag

  0：wipe inside storage；

  1：wipe outside storage.

- [Return value]

➢ ERRORCODE_SUCCESS

Executive correct；

➢ Other value

If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 5. 3 Camera Management Subclass API

```
Class MtmCamaraManager
```

Class Description:

Camera management subclass is responsible for the management various camera functions of intelligent terminal device. MtmCameraManager is the subclass of EMCG_MTM_Manager.

B. 5. 3. 1 Set enable camera

```
int setEnabledCamera ();
```

Function Description:

Open or off the camera is acceptable, which provides mobile device security scene.

Parameter Description:

- [Input] boolean enable

  true：allowed；

  false：not allowed.

- [Return value]
  - ERRORCODE_SUCCESS

Executive correct；

- Other value

If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 5. 4 Recorder Management Subclass API

Class MtmRecorderManager

Class Description:

Recorder management subclass is responsible for the management various recorder functions of intelligent terminal device. MtmRecorderManager is the subclass of EMCG_MTM_Manager.

B. 5. 4. 1 Set Enable Microphone Mute

```
void setMicrophoneMute();
```

Function Description:

Set Enable Microphone Mute

Parameter Description:

- [Input] boolean enable

  true：allowed；

  false：not allowed.

- [Return value]
  - ERRORCODE_SUCCESS

Executive correct；

- Other value

If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 5. 4. 2 Set Enable Audio Recorder

```
int setEnableAudioRecorder（）；
```

Function Description:

Set Enable Audio Recorder

- [Input] boolean enable

  tr true：allowed；

  false：not allowed.

- [Return value]
  - ERRORCODE_SUCCESS

Executive correct；

- Other value

If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 6 App Management and Control API

B. 6. 1 App Configuration API

This chapter refers to the email and App configuration.

B. 6. 1. 1 Add Mail Config

```
int addMailConfig ();
```

Function Description:

It supports Exchange, POP3 and IMAP accounts.

Function Description:

- [Input] String configstring

  Mail configuration info and XML structure, please see appendix format.

- [Return value]
  - ERRORCODE_SUCCESS

Executive correct；

- Other value

If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 6. 1. 2 Delete Mail Config

```
int deleteMailConfig ();
```

Function Description:

Delete Mail config

Parameter Description:

- [Input]

  Nothing.

- [Return value]

➢ ERRORCODE_SUCCESS

Executive correct；

➢ Other value

If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 6. 1. 3 Update Mail Config

```
int updateMailConfig();
```

Function Description:

It supports Exchange, POP3 and IMAP accounts.

Parameter Description:

- [Input] String configstring

  Mail configuration info and XML structure, please see appendix format.

- [Return value]

➢ ERRORCODE_SUCCESS

Executive correct；

➢ Other value

If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 6. 1. 4 Get Running App List

```
int getRunningAppList();
```

Function Description:

Get Running App List, such as com.tencent.qq and com.baidu.map, etc.

Parameter Description:

- [Input]String returnNames

  Return to one parameter, which includes many App package name parameters.

- [Return value]

➢ ERRORCODE_SUCCESS

Executive correct；

➢ Other value

If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 6. 1. 5 Get App Install History

```
String getAppInstallHistory();
```

Function Description:

Get App Install History

Parameter Description:

- [Input]

  Nothing.

- [Return value] String

  App install history records are presented with XML format, which include every App's installation date, time, package name and whether the installation success. Please see attached specific format.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 6. 1. 6 Start App

```
int startApp();
```

Function Description:

Start specific App package program.

Parameter Description:

- [Input]String appID

  Input App package program name, such as: "com.baidu.browser".

- [Return value]
  - ERRORCODE_SUCCESS

Executive correct;

- Other value

If it is error, please see the chapter of "error code definition".

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 6. 1. 7 End App

```
int endApp();
```

Function Description:

End specific App program.

Parameter Description:

- [Input]String appID

  Input App program package name.

- [Return value]
  - ERRORCODE_SUCCESS

Executive correct;

➢ Other value

If it is error, please see the chapter of "error code definition".

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 6. 1. 8 Set Enable App Auto Run

```
int setEnableAppAutoRun();
```

Function Description:

Set Enable App Auto Run

Parameter Description:

- [Input]String appID

  Input App program package name,

- [Input]boolean enable

  true：add；

  false：delete.

- [Return value]

  ➢ ERRORCODE_SUCCESS

Executive correct；

➢ Other value

If it is error, please see the chapter of "error code definition".

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 6. 2 Native Browser Subclass API

```
Class MtmBrowserManager
```

Class Description:

Native Browser Subclass API is responsible for the subclass of EMCG_MTM_Manager.

B. 6. 2. 1 Browser access

```
int setEnableNativeBrowser();
```

Function Description:

Set Enable Native Browser

Parameter Description:

- [Input] boolean enable

  true：allowed；

  false：not allowed.

- [Return value]

  ➢ ERRORCODE_SUCCESS

Executive correct；

➢ Other value

If it is error, please see the chapter of "error code definition".

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 6. 2. 2 Browser cookie control

```
int setEnableBrowserCookie();
```

Function Description:

Set enable Browser cookie access.

Parameter Description:

- [Input] boolean enable

  true：allowed;

  false：not allowed.

- [Return value]

➢ ERRORCODE_SUCCESS

Executive correct；

➢ Other value

If it is error, please see the chapter of "error code definition".

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 6. 2. 3 Browser javascript control

```
int setEnableBrowserJavascript();
```

Function Description:

Set enable Browser javascript access.

Parameter Description:

- [Input] boolean enable

  true：allowed;

  false：not allowed.

- [Return value]

➢ ERRORCODE_SUCCESS

Executive correct；

➢ Other value

If it is error, please see the chapter of "error code definition".

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 6. 2. 4 Set Enable Browser Pop Window

```
int setEnableBrowserPopWindow();
```

Function Description:

Set Enable Browser Pop Window

Parameter Description:

- [Input] boolean enable

  true：allowed;

  false：not allowed.

- [Return value]
  - ERRORCODE_SUCCESS

Executive correct；

- Other value

If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 6. 2. 5 Set Enable Browser Auto Fill

```
int setEnableBrowserAutoFill ();
```

Function Description:

Set Enable Browser Auto Fill

Parameter Description:

- [Input]boolean enable

  true：allowed;

  false：not allowed.

- [Return value]
  - ERRORCODE_SUCCESS

Executive correct；

- Other value

If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 6. 2. 6 Browser https Security Warning

```
int setEnableBrowserSafetyWarning ();
```

Function Description:

Set Browser https Security Warning.

Parameter Description:

- [[Input]boolean enable

  true：allowed;

  false：not allowed.

- [Return value]
  - ERRORCODE_SUCCESS

Executive correct；

- Other value

If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 6. 2. 7 URL access

```
int setBrowserUrlBlockingMode();
```

Function Description:

Set URL access.

Parameter Description:

- [Input]int blockingMode

  URL access mode：

  - 0：not allowed to access；
  - 1：set black list mode；
  - 2：set whit list mode.

- [Return value]
  - ERRORCODE_SUCCESS

Executive correct；

- Other value

If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 6. 2. 8 Add Browser Bookmark

```
int addBrowserBookmark();
```

Function Description:

Add Browser Bookmark.

Parameter Description:

- [Input]String bookmarkName

  Bookmark Name；

- [Input]String url

  Bookmark Url address.

- [Return value]

➢ ERRORCODE_SUCCESS

Executive correct；

➢ Other value

If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 6. 3 App installation, operation and removing subclass API

Class MtmAppManager

Class Description:

App installation, operation and removing management subclass is the subclass of EMCG_MTM_Manager.

B. 6. 3. 1 Silent mode of installation App

int forceInstallApp ();

Function Description:

Force and silent install specific App.

Parameter Description:

- [Input]String appFilePath

  App FilePath.

- [Return value]

➢ ERRORCODE_SUCCESS

Executive correct；

➢ Other value

If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 6. 3. 2 Force Uninstall App

int forceUninstallApp ();

Function Description:

Force and silent uninstall specific package App.

Parameter Description:

- [Input]String appID

  Uninstallation App package name.

- [Return value]

➢ ERRORCODE_SUCCESS

Executive correct；

➢ Other value

If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 6. 3. 3 Set Enable App Installation

```
int setEnableAppInstallation();
```

Function Description:

Enable the installation of SD/TF card’s software package or the installation by third party.

Parameter Description:

- [Input]boolean enable

  true：allowed;

  false：not allowed.

- [Return value]

➢ ERRORCODE_SUCCESS

Executive correct；

➢ Other value

If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 6. 3. 4 Set Enable App Uninstallation

```
int setEnableAppUninstallation();
```

Function Description:

Set Enable App Uninstallation, which includes the user could not uninstall App through adb and third party’s App.

Parameter Description:

- [Input]boolean enable

  true：allowed;

  false：not allowed.

- [Return value]

➢ ERRORCODE_SUCCESS

Executive correct；

➢ Other value

If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 6. 3. 5 Set Enable Kill App

```
int setEnableKillApp();
```

Function Description:

Set enable kill App specific process or service, which includes the user could kill App through adb and third party's App.

Parameter Description:

- [Input]String appID

  Specific package App name or service

- [Input]boolean enable

  true：allowed;

  false：not allowed.

- [Return value]
  - ERRORCODE_SUCCESS

Executive correct；

- Other value

If it is error, please see the chapter of "error code definition".

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 6. 3. 6 Set Enable ADB

```
int setEnableADB();
```

Function Description:

Set enable ADB install or uninstall App, which includes relying on ADB negotiated third party functions.

Parameter Description:

- [Input]boolean enable

  true：allowed;

  false：not allowed.

- [Return value]
  - ERRORCODE_SUCCESS

Executive correct；

- Other value

If it is error, please see the chapter of "error code definition".

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 6. 3. 7 Set Enable Execute App

```
int setEnableExecuteApp();
```

Function Description:

Set enable executive specific package App name.

Parameter Description:

- [Input]String appID

  Specific package App name or service

- [Input]boolean enable

  true：allowed;

  false：not allowed.

- [Return value]
  - ERRORCODE_SUCCESS

Executive correct；

- Other value

If it is error, please see the chapter of "error code definition".

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 6. 3. 8 Erase App Data

```
int eraseAppData();
```

Function Description:

Erase specific App Data.

Parameter Description:

- [Input] String appID

  Specific App package name.

- [Return value]
  - ERRORCODE_SUCCESS

Executive correct；

- Other value

If it is error, please see the chapter of "error code definition".

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 6. 3. 9 Set Enable Recent App

```
int setEnableRecentApp();
```

Function Description:

Set Enable Recent App.

Parameter Description:

- [Input]boolean enable

  true：allowed;

  false：not allowed.

- [Return value]
  - ERRORCODE_SUCCESS

Executive correct；

- Other value

If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 6. 4 App’s black and white list API

```
Class MtmAppFilterManager
```

Class Description:

App’s black and white list filtering capability is the subclass of EMCG_MTM_Manager.

B. 6. 4. 1 Set App Running Block Mode

```
int setAppRunningBlockMode();
```

Function Description:

Set App Running Block/White Mode

Parameter Description:

- [Input]int blockingMode

  App blocking mode：

  - 0：black/white list mode are not allowed；
  - 1：set black list mode；
  - 2：set white list mode.

- [Return value]
  - ERRORCODE_SUCCESS

Executive correct；

- Other value

If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 6. 4. 2 Set App Running Block Config

```
int setAppRunningBlockConfig();
```

Function Description:

Running block configuration's content includes black list and white list information. It could activate black and white list through setting App running block mode.

Parameter Description:

- [Input] String blockingConfigInfo

  Set App Running Block Config.

  App Running Block Config Format：

```
<?xml version="1.0" encoding="UTF-8"?>
<APP_RUNNING_BLOCK_CONFIG>
    <BLACKLIST>
      <APPITEM>
        <NAME>aixiaochu</NAME>
        <APPID>com.test.aixiaochu</APPID>
      </APPITEM>
      <APPITEM>
        <NAME>kuaipao</NAME>
        <APPID>com.unknown.kuaipao</APPID>
      </APPITEM>
    </BLACKLIST >
    <WHITELIST>
      <APPITEM>
        <NAME>weixin</NAME>
        <APPID>com.tencent.weixin</APPID>
      </APPITEM>
      <APPITEM>
        <NAME> com.baidu.ime </NAME>
        <APPID>com.baidu.ime</APPID>
      </APPITEM>
    </WHITELIST>
</APP_RUNNING_BLOCK_CONFIG>
```

- [Return value]
  - ERRORCODE_SUCCESS

Executive correct；

- Other value

If it is error, please see the chapter of "error code definition".

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 6. 4. 3 Set App Install Block Mode

```
int setAppInstallBlockMode();
```

Function Description:

Set App black/white list install mode

Parameter Description:

- [Input]int blockingMode

  App blocking mode：

  - 0：black/white list mode are not allowed；
  - 1：set black list mode；
  - 2：set white list mode.

- [Return value]
  - ERRORCODE_SUCCESS

Executive correct；

- Other value

If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 6. 4. 4 Set App Install Block Config

```
int setAppInstallBlockConfig();
```

Function Description:

Install block configuration’s content includes black list and white list information. It could activate black and white list through setting App running block mode.

Parameter Description:

- [Input] String blockingConfigInfo

  App Install Block Config

  App Install Block Config File’s Format：

```
<?xml version="1.0" encoding="UTF-8"?>
<APP_INSTALL_BLOCK_CONFIG>
    <BLACKLIST>
      <APPITEM>
        <NAME>aixiaochu</NAME>
        <APPID>com.test.aixiaochu</APPID>
      </APPITEM>
      <APPITEM>
        <NAME>kuaipao</NAME>
        <APPID>com.unknown.kuaipao</APPID>
```

```
        </APPITEM>
    </BLACKLIST >
    <WHITELIST>
        <APPITEM>
            <NAME>weixin</NAME>
            <APPID>com.tencent.weixin</APPID>
        </APPITEM>
        <APPITEM>
            <NAME> com.baidu.ime </NAME>
            <APPID>com.baidu.ime</APPID>
        </APPITEM>
    </WHITELIST>
</APP_INSTALL_BLOCK_CONFIG>
```

- [Return value]
  - ERRORCODE_SUCCESS

Executive correct；

- Other value

If it is error, please see the chapter of “error code definition”.

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 6. 4. 5 Force App Keep Running

```
int forceAppKeepRunning();
```

Function Description:

Setting force specific App running’s list information to maintain the operation.

Parameter Description:

- [Input] String keepRunningAppInfo

  App force running configuration

  App force running configuration format：

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

<APPITEM>
    <NAME>aixiaochu</NAME>
    <APPID>com.test.aixiaochu</APPID>
</APPITEM>
<APPITEM>
    <NAME>kuaipao</NAME>
    <APPID>com.unknown.kuaipao</APPID>
</APPITEM>
```

```
</KEEP_RUNNING_APP_CONFIG>
```

- [Return value]
  - ERRORCODE_SUCCESS

Executive correct；

- Other value

If it is error, please see the chapter of "error code definition".

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 6. 4. 6 Uninstall App Black List

```
int uninstallAppInBlackList ();
```

Function Description:

Uninstall App all Black List.

Parameter Description:

- [Input]

  Nothing.

- [Return value]
  - ERRORCODE_SUCCESS

Executive correct；

- Other value

If it is error, please see the chapter of "error code definition".

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

Please see the reference of setting AppInstallBlockConfig function.

B. 6. 4. 7 Set Forbid Uninstallation List

```
int setForbitUninstallationList();
```

Function Description:

Set forbid uninstallation list and to ensure these certain Apps could not be uninstalled by users.

Parameter Description:

- [Input] String forbitUninstallationAppInfo

  App Running Config

  App Running Config File's Format：

```
<?xml version="1.0" encoding="UTF-8"?>
<FORBIT_UNINSTALL_APP_CONFIG>
<APPITEM>
    <NAME>aixiaochu</NAME>
    <APPID>com.test.aixiaochu</APPID>
```

```
</APPITEM>
<APPITEM>
    <NAME>kuaipao</NAME>
    <APPID>com.unknown.kuaipao</APPID>
</APPITEM>
</FORBIT_UNINSTALL_APP_CONFIG>
```

- [Return value]
  - ERRORCODE_SUCCESS

Executive correct；

- Other value

If it is error, please see the chapter of "error code definition".

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 6. 4. 8 Set URL Blocking Mode

```
int setURLBlockingMode();
```

Function Description:

Set URL blocking mode, including black list, white list and normal mode.

Parameter Description:

- [Input]int blockingMode

  App blocking mode：

  - 0：black/white list mode are not allowed；
  - 1：set black list mode；
  - 2：set white list mode.

- [Return value]
  - ERRORCODE_SUCCESS

Executive correct；

- Other value

If it is error, please see the chapter of "error code definition".

Notice:

- It should be finished the MTM-App identity authenticate function calls in advance.

B. 6. 4. 9 Set URL Blocking Config

```
int setURLBlockingConfig();
```

Function Description:

Setting URL block configuration's content includes black list and white list information. It could activate black and white list through setting URL Blocking mode.

Parameter Description:

- [Input] String blockingConfigInfo

  URL Blocking Configuration's format:

```
<?xml version="1.0" encoding="UTF-8"?>
<URL_BLOCK_CONFIG>
    <BLACKLIST>
      <URLITEM>
        <NAME>Bad Url#1</NAME>
        <URL>fakebank.com</ URL >
      </URLITEM>
      <URLITEM>
        <NAME> Bad Url#1</NAME>
        <URL>www.getsomemoney.com</URL>
      </URLITEM>
    </BLACKLIST>
    <WHITELIST>
      <URLITEM>
        <NAME>qiyeneiwang</NAME>
        <URL>192.168.1.112</URL>
      </URLITEM>
    </WHITELIST>
</URL_BLOCK_CONFIG>
```

- [Return value]
  - ERRORCODE_SUCCESS

Executive correct；

- Other value

If it is error, please see the chapter of "error code definition".

Notice:

It should be finished the MTM-App identity authenticate function calls in advance.

B. 7 Error Code Definition

Function Normal Error Code Definition Table

Error Code Definition	Value number	Meaning
ERRCODE_SUCCESS	0	Function operation is correct.
ERRCODE_UNKNOWN	-1	Unknown Exception Error
ERRCODE_EXECUTEERR	-2	Executive Error
ERRCODE_PERMISSIONERR	-3	Permission Error

ERRCODE_OBJECTNOTFOUNDERR	-4	Object is not found. (Such as SIM card, WIFI device and Bluetooth, etc.)
ERRORCODE_INVALIDSIGNATURE	-11	APP signature certificate is in valid.

Bibliography

《Enterprise intelligent mobile terminal App developed, installed and operating mechanism（ZISIA/EMCG 2014-3）》

《Trusted cloud service authentication evaluation's twelfth parts: Cloud Service of enterprise mobile management(EMM)》

《Information Security Technology The basic requirements of Information Security Technology Classified Protection's fourth part: Mobile Interconnection requirement（GB/T 22239-4-2015）》（Draft）