服务器核心模块概要设计

1	引言	計		2
	1.1	编写	目的	2
	1.2	背景		2
2	总体	本设计		2
	2.1	需求	规定	2
	2.2	运行	环境	2
	2.3	开发	环境	2
	2.4	基本	设计概念和处理流程	2
		2.4.1	调用模型	2
		2.4.2	基于事件驱动的数据处理模型	3
		2.4.3	事件数据设计	3
		2.4.4	框架体系设计	4
	2.5	功能	模块划分	4
3	模均	块详细设	计	5
	3.1	数据	队列服务 QueueService 模块设计	5
		3.1.1	基本设计概念	5
		3.1.2	类与接口设计	5
		3.3.1	运行流程	9
	3.2	网络	引擎 TCPSocketEngine 模块设计	9
		3.2.1	网络重叠完成端口 IOCP 简介	9
		3.2.2	基本设计概念	10
		3.2.3	网络数据命令包结构设计	
		3.2.4	网络数据加密设计	11
		3.2.5	类与接口设计	12
	3.3	数据	库引擎 DataBaseEngine 模块设计	12
		3.3.1	基本设计概念	12
		3.3.2	类与接口设计	12
		3.3.3	运行流程	24
	3.4	定时	器引擎 TimerEngine 模块设计	24
		3.4.1	基本设计概念	24
		3.4.2	类与接口设计	24
		3.4.3	运行流程	27
	3.5	调度	引擎 AttemperEngine 模块设计	28
		3.5.1	基本设计概念	28
		3.5.2	类与接口设计	28
		3.5.3	运行流程	32
	3.6	异步	引擎 AsynchronismEngine 模块设计	32
		3.6.1	基本设计概念	33
		3.6.2	类与接口设计	33
		3.6.3	运行流程	36
	3.7	服务	引擎 ServiceEngine 模块设计	37
		3.7.1	基本设计概念	37
		3.7.2	类与接口设计	37

1 引言

1.1 编写目的

方便二次开发人员理解服务器核心技术

1.2 背景

- 3.3.1 为一般游戏服务器设计的核心模块
- 3.3.2 本项目的任务开发者:Godzilar 系统使用用户:游戏服务器二次开发人员

2 总体设计

2.1 需求规定

3.3.1 系统功能

提供网络服务接口,数据库访问接口,定时器接口,并提供调度钩子接口,让二次开发人员调用, 更多精力处理游戏逻辑

3.3.2 系统性能

网络负载支持 2000 同时在线连接以上 数据库访问支持 100 并发访问 定时器时间精度达 25ms 之内

2.2 运行环境

硬件平台:x86 体系结构 软件平台:Window2003 操作系统

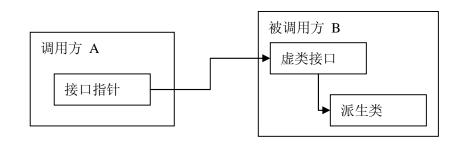
2.3 开发环境

开发工具:Visual C++ 2003 开发语言:C++

2.4 基本设计概念和处理流程

2.4.1 调用模型

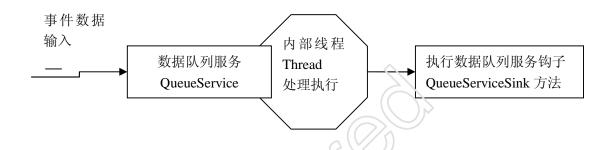
利用面向对象思想多态性,**调用方**保存着**被调用方**的基础接口指针(一般称呼为 钩子),调用方直接调用接口指针里面方法,方法具体实现逻辑由该接口的派生类实现,示意图:



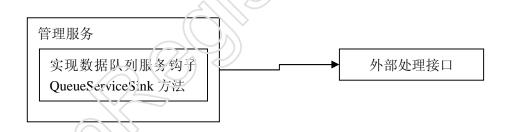
2.4.2 基于事件驱动的数据处理模型

为系统功能设计,需要处理网络事件数据 SocketEvent,数据库事件数据 DatabaseEvent,定时器事件数据 TimerEvent 等,为此建立数据队列服务 QueueService,为每一队列建立多个子线程 QueueServiceThread 处理。数据队列服务提供添加事件数据方法 AddToQueue,设置数据队列服务钩子 SetQueueServiceSink,让数据队列服务钩子 QueueServiceSink 做具体逻辑事件数据处理。

示意图:



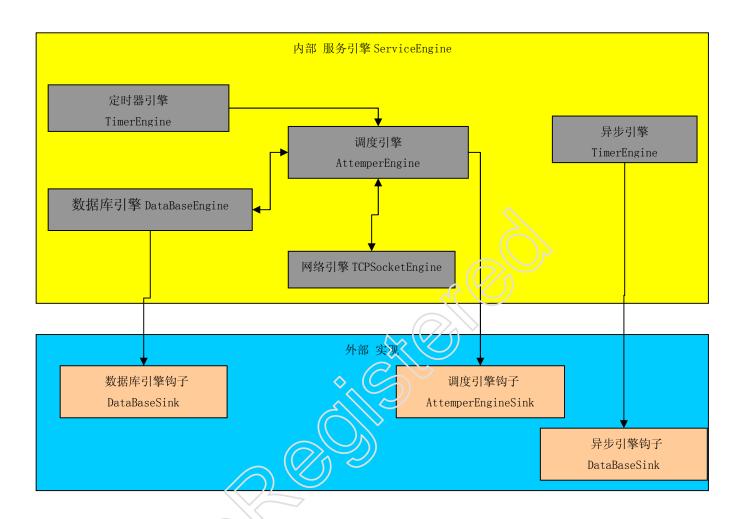
再者,根据网络,数据库等特定功能,构建网络,数据库等管理服务,实现数据队列服务钩子QueueServiceSink 方法,调度事件数据执行外部处理接口,外部处理接口具体由二次开发用户实现。示意图:



2.4.3 事件数据设计

名称	描述	类型
定时器事件	根据时间间隔,定期产生	自发,被动
TimerEvent		
数 据 库 事 件	请求访问数据库	用户自定义, 主动
DatabaseEvent		
网络应答事件	客户端连接网络服务,成功连接	自发,被动
SocketAcceptEvent		
网络读取事件	客户端发送数据,网络服务成功读取	自发,被动
SocketReadEvent		
网络关闭事件	客户端失去连接	自发,被动
SocketControlEvent		
控制事件	用户自定义控制服务	用户自定义, 主动
ControlEvent		

2.4.4 框架体系设计



2.5 功能模块划分

名称	描述	开发优先
数据队列服务	外部接受输入事件数据,内部实现事件数据链表保存与管	1
QueueService	理, 处理子线程从链表获取事件数据调用数据队列钩子方	
	法处理事件数据	
网络引擎	实现数据队列服务钩子方法,管理网络事件数据,内部启	2
TCPSocketEngine	动网络服务器监听服务,接收与发送网络数据,并调用网络	
	处理钩子进行逻辑处理	
数据库引擎	实现数据队列服务钩子方法,管理数据库事件数据,并调	2
DataBaseEngine	用数据库处理钩子进行逻辑处理,还提供数据库访问的帮	
	助类	
定时器引擎	实现设置定时,删除定时请求,发生定时事件	2
TimerEngine		
调度引擎	接收,调度与管理网络事件数据,数据库事件数据,定时	2

AttemperEngine	器事件数据,自定义调度事件数据	
异步引擎	设置多个数据处理钩子,异步接收请求并处理	2
AsynchronismEngine		
服务引擎	管理网络引擎,数据库引擎,定时器引擎,调度引擎,异	3
ServiceEngine	步引擎	

3 模块详细设计

3.1 数据队列服务 QueueService 模块设计

3.1.1 基本设计概念

建立一个内存链表,保存事件数据,对外部提供方法往链表添加事件数据并通知线程,同时启动多个处理线程,从数据链表里获取事件数据,执行外部钩子方法进行处理.线程事件通知采用完成端口技术.

3.1.2 类与接口设计

	● 接口	设计	
IQueueServi	CeEngine		
队列引擎接口]		
模块	QueueS	ervice	
继承 IQueueServiceEngine → IUnknownEx			
实现			
		数据成员	
	名称	说明	
		方法	
	ol <u>cdec</u>	1 StartService(BVIE cbThreadCount)	
说明	干始服务		
1	参数	BYTE cbThreadCount 处理事件数据的线程数量	
ij	5回	类型为 Bool, 启动成功返回 true, 否则 false	
举例			
virtual boo	ol_cdec	StopService()	
说明 停止服务			
3			
ij	5回	类型为 Bool, 启动成功返回 true, 否则 false	
举例			
virtual boo	olcdec	l SetQueueServiceSink(IUnknownEx * pIUnknownEx)	
说明	设置接口外	部事件处理钩子	
-	参数	IUnknownEx * pIUnknownEx 外部事件数据处理钩子	
ij	5回	类型为 Bool, 设置成功返回 true, 否则 false	
举例		• •	
virtual boo	ol cdec	1 GetBurthenInfo(tagBurthenInfo & BurthenInfo)	
		列负荷信息	
- 1	参数	tagBurthenInfo & BurthenInfo 负荷信息结构体	
	5回	类型为 Bool, 设置成功返回 true, 否则 false	

举例	

IQueueSe:	rviceSink				
	队列服务钩子,由外部继层实现				
模块	QueueS	ervice			
继承	IQueues	ServiceSink	→ IUnknow	mEx	
实现					
			数据	成员	
	名称			说明	
			方	去	
virtual	void <u>cdec</u>	1 OnQueueSer	cviceSink(B	YTE cbThreadIndex,	
WORD wide	entifier, <mark>v</mark>	oid * pBuffe	er, WORD wD	ataSize, DWORD dwInsertTime)	
说明	通知回调函数				
	参数	BYTE cbThr	eadIndex	处理事件数据的线程索引	
		WORD wIden	tifier	事件数据标记	
	void * pBu		ffer	数据指句	
WORD wData		Size	数据大小		
DWORD dwIn		sertTime	插入时间值		
	返回		0,0		
举例			4		

CQueueSei	CQueueServiceThread					
队列处全	队列处金钱全售					
模块	QueueS	ervice				
继承	CQueue	ServiceThread $ o$ CServiceThre	ad			
实现						
		数据成员				
名称 说明						
HANDLE	m_hComplet	ionPort	完成端口句柄			
BYTE	m_cbThread	dIndex	线程索引			
BYTE	m_cbBuffer	[MAX_QUEUE_PACKET]	事件接收缓冲			
	方法					
bool Ini	tThread(HAN	DLE hCompletionPort,BYTE cbTh	readIndex)			
说明	配置					
	参数	HANDLE hCompletionPort 完成	端口句柄			
		BYTE cbThreadIndex 线程	索引			
	返回	类型为 Bool, 设置成功返回 true	, 否则 false			

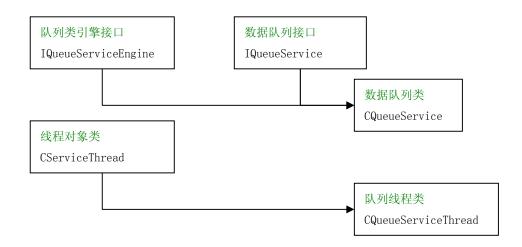
举例			
bool Un	InitThread	()	
说明	取消配置		
	参数		
	返回	类型为 Bool, 设置成功返回 t	rue, 否则 false
举例			
virtual	bool Repe	titionRun()	
说明	线程运行	函数	
	参数		
	返回	类型为 Bool	
举例			
			^
<u> </u>			
CQueueS	ervice		(())
队列服务	5 类		
模块	Queu	eService	A V
继承	CQuei	ueService → IQueueServiceEn	gine, IQueueService
实现 IQueueServiceEngine, IQueueSer		ueServiceEngine, IQueueServic	se /
		数据成员	
		名称	说明
bool m_bService			服务标志
IQueueS	erviceSink	* m_pIQueueServiceSink	回调钩子接口
CThread	Lock m_T	hreadLock	线程锁
CDataSt	orage m_D	ataStorage	数据存储

数据成员 名称 説明				
boolm_bService服务标志IQueueServiceSink * m_pIQueueServiceSink回调钩子接口CThreadLock烟_ThreadLock线程锁CDataStorage数据存储				
IQueueServiceSink * m_pIQueueServiceSink 回调钩子接口 CThreadLock m_ThreadLock 线程锁 CDataStorage m_DataStorage 数据存储				
CThreadLock m_ThreadLock 线程锁 CDataStorage m_DataStorage 数据存储				
CDataStorage m_DataStorage 数据存储				
CO C				
CQueueServiceThread*m_QueueServiceThread				
HANDLE* m_hCompletionPort 完成端口指针				
BYTE m_ceThreadCount 线程处理数目				
方法				
virtual boolcdecl AddToQueue(WORD wIdentifier, void * const pBuffer, wDataSize)	WOR			
说明 添加事件数据				
参数 WORD wIdentifier 事件数据标记				
void * const pBuffer 事件数据指针				
WORD wDataSize 事件数据大小				
返回 类型为 Bool, 添加成功返回 true, 否则 false				
举例				

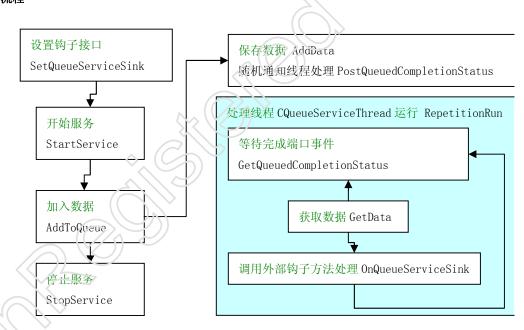
hool adoal StartSarviga (PVTF ahThroadCount)			
开始服务			
参数	BYTE cbThreadCount	线程数量	
返回	类型为 Bool, 开始成功返回 tru	ıe, 否则 false	
	开始服务参数	参数 BYTE cbThreadCount	

virtual	boolcdec	1 StopService()	
说明	停止服务		
	参数		
	返回	类型为 Bool,停止成功返回 true, 否则 false	
举例			
virtual	bool <u>cdec</u>	l SetQueueServiceSink(IUnknownEx * pIUnknownEx)	
说明	说明 设置接口外部事件处理钩子		
	参数	IUnknownEx * pIUnknownEx 外部事件数据处理钩子	
	返回	类型为 Bool, 设置成功返回 true, 否则 false	
举例			
virtual	bool <u>cdec</u>	l GetBurthenInfo(tagBurthenInfo & BurthenInfo)	
说明	获取数据队	列负荷信息	
	参数	tagBurthenInfo & BurthenInfo 负荷信息结构体	
	返回	类型为 Bool, 设置成功返回 true, 否则 false	
举例			
bool Get	Data(tagDat	aHead & DataHead, void * pBuffer, WOky wBufferSize)	
说明	获取事件数	2据	
	参数	tagDataHead & DataHead 事件数据头	
		void * pBuffer 数据指针	
		WORD wBufferSize 数据大小	
W 6-1	返回	类型为 Bool, 获取成功返回 true, 否则 false	
举例			
		Thream BYTE chinesdindex, const tagDataHead & DataHead, void *	
说明	WORD wDataSiz?) 事件消息, 主要由队列线程调用		
DE-91			
	参数	BYTE cbThreadIndex 线程索引	
		P tagDataHead & DataHead 事件数据头 void * pBuffer 数据指针	
		WORD wBufferSize 数据大小	
	返回	Man in the second secon	
举例			
	1		

● 类继承图



3.3.1 运行流程



3.2 网络引擎 TCPSocketEngine 模块设计

3.2.1 网络重叠完成端口 IOCP 简介

完成端口: 是一个 FIFO 队列,操作系统的 IO 子系统在 IO 操作完成后,会把相应的 IO packet 放入该队列。

-等待者线程队列:通过调用 GetQueuedCompletionStatus API,在完成端口上等待取下一个 IO packet。

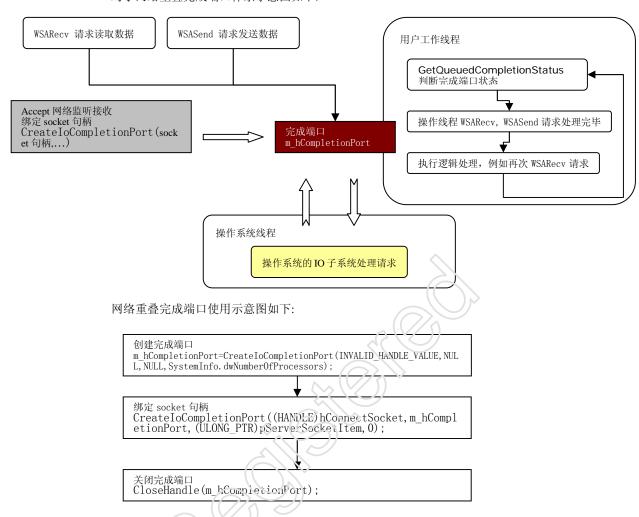
执行者线程组:已经从完成端口上获得 IO packet,在占用 CPU 进行处理。

除了以上三种类型的参与者。我们还应该注意两个关联关系,即:

IO Handle 与完成端口相关联:任何期望使用 IOCP 的方式来处理 IO 请求的,必须将相应的 IO Handle 与该完成端口相关联。需要指出的时,这里的 IO Handle,可以是File 的 Handle,或者是 Socket 的 Handle。

-线程与完成端口相关联:任何调用 GetQueuedCompletionStatus API 的线程,都将与该完成端口相关联。在任何给定的时候,该线程只能与一个完成端口相关联,与最后一次调用的 GetQueuedCompletionStatus 为准.

对于网络重叠完成端口体系示意图如下:



3.2.2 基本设计概念

建立数据队列服务 QueueService,对外部输入的对 socket 的发送,关闭请求等处理。同时 建立工作 线程 SocketAcceptThread 做应答 Socket,建立读写工作线程 ServerSocketRSThread 判断完成端口状态,等待系统线程处理读写操作完成,如果完成读操作,发送调度事件到调度引擎处理,并继续投递读操作,如果完成发送操作,判断还有发送请求就继续投递发送请求。最后建立检测工作线程,发送心跳包,检测断线 socket 连接。

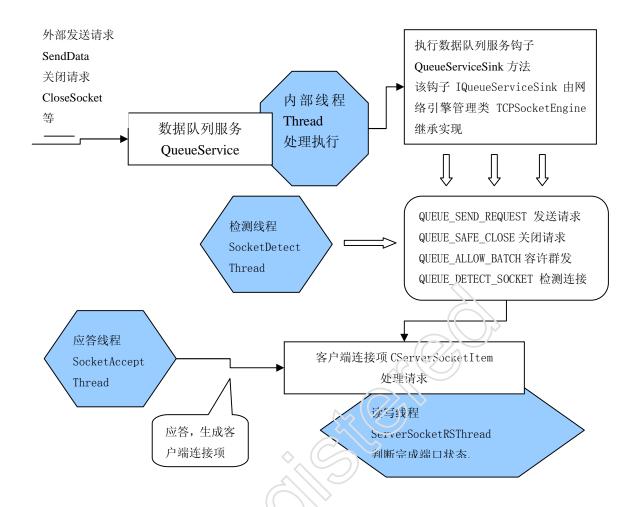
● 数据设计与管理

先设计客户端连接项 ServerSocketItem, 实现对客户端连接 Socket 的发送, 接收数据,加密/解密网络包,关闭等操作

再设计网络引擎 TCPSocketEngine,对客户端连接项的管理,同时接收外部的输入操作请求,还有对应答线程,读写线程,检测线程管理

设计应答线程,读写线程,检测线程的功能进行封装,还有重叠数据的帮助封装。

● 工作线程运行设计



3.2.3 网络数据命令包结构设计

网络数据包分两个区域.包头,用户数据

包头:每个网络包都带的数据区域,装载数据包结构信息与数据包命令信息 用户数据:根据命令信息不一样,用户自定义的负载数据 具体结构如下:



3.2.4 网络数据加密设计

数据加密部分为CMD_Command 数据包命令信息 与 CMD_Data 用户命令数据, CMD_Info 数据包结构

信息不加密.

加密算法为根据字节影射表,将字节替换。例如数据包第6个字节是0x90,通过字节影射表查找,对应的是0x1d,则该字节替换为0x1d发送.

3.2.5 类与接口设计

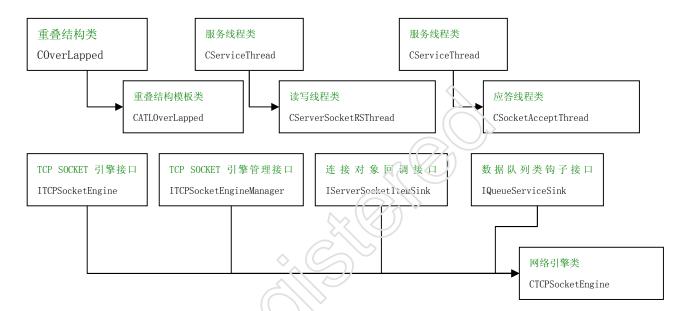
● 接口设计

请查阅 SocketEngine.h 头文件

类设计

请查阅 SocketEngine.h 头文件

● 类继承图



3.3 数据库引擎 DataBaseEngine 複块设计

3.3.1 基本设计概念

数据库引擎建立数据队列服务,接受外部数据库事件数据输入,并调用外部数据库处理 钩子方法处理且提供数据库帮助类对象,实现对数据库访问,数据访问技术采用 ADO.

3.3.2 类与滚口设计

INErro	I Common of the					
数据序错	数据。基础口					
模块	Databas	Database				
继承	IAD0Er	ror → IUnkr	nownEx			
实现						
	数据成员					
	名称		说明			
	方法					
virtual I	LPCTSTRc	decl GetErro	orDescribe()			
说明	获取错误描	获取错误描述				
	参数					
	返回	描述错误的]文本字符串			

nADOErrorTy	ype <u>cdecl</u> GetErrorType()	
获取错误类型		
参数		
返回	错误类型的枚举值	
	获取错误类 参数	

IDataBase	е					
数据库连	接并访问接					
模块	Databa	ase				
继承	IData	Base → IUnkı	nownEx	Α.		
实现						
数据成员						
名称				说明		
			<u> </u>	2(//)		
			方法			
virtual	oool <u>cde</u>	cl OpenConnec	tion()	\uparrow		
说明	打开连接					
	参数		\$ CO			
	返回	操作成功返	回 ture,否则 flase			
举例		(
virtual l	ooolcde	cl CloseRecor	dset			
说明	关闭记录					
	参数					
		返回 操作成功返回 ture,否则 flase				
举例 🔨	7					
virtual)	opo! cde	cl CloseConne	ection()			
说明	关闭连接					
	参数					
	返回	操作成功返	回 ture,否则 flase			
举例						
virtual N	oool <u></u> cde	cl TryConnect	Again(bool bFocusConn	ect, CComError * pComError)		
说明	重新连接					
	参数	bool bFocu	sConnect	强制性连接标记		
		CComError	* pComError	错误保存		
	返回	操作成功返	回 ture,否则 flase			
举例						
virtual N	ooolcde	cl SetConnect	ionInfo(LPCTSTR szIP,			
LPCTSTR :	PCTSTR szPipeName, WORD wPort, LPCTSTR szData,					
LPCTSTR :	CTSTR szName, LPCTSTR szPass)					

说明	设置信息		
	参数	LPCTSTR szIP	数据库服务器 IP 地址
		LPCTSTR szPipeName	数据库命名实列通道
		WORD wPort	端口
		LPCTSTR szData	数据库名称
		LPCTSTR szName	访问用户名
		LPCTSTR szPass	访问密码
	返回	操作成功返回 ture,否则 flase	•
举例			
virtual	bool <u>cdec</u>	l IsConnectError()	
说明	判断是否连	接错误	
	参数		
	返回	操作成功返回 ture,否则 flase	
举例			
virtual l	boolcdec	l IsRecordsetOpened ()	
说明	判断是否打	开记录集	(A)
	参数		
	返回	操作成功返回 ture,否则 flase	
举例			
virtual	boolcdec	1 MoveToNext ()	
说明	往下移动记	L 录游标	
	参数		
	返回	操作成功返回 ture,否则 flase	
举例		2 (0)	
virtual	bool _cdec	MoveTcFirst()	
说明	移到开头		
	参数		
	返回	操作成功返回 ture,否则 flase	·
举例			
virtual	boolcdec	l IsEndRecordset ()	
说明	是否结束		
	参数		
	返回	操作成功返回 ture,否则 flase	>
举例			
virtual	long <u>cdec</u>	l GetRecordCount ()	
说明	获取数目		
	参数		
	返回	操作成功返回记录数目	
举例		1	
virtual	longcdec	l GetActualSize(LPCTSTR ps	szParamName)

说明	获取字段大小			
		LPCTSTR pszParamName 参数名称		
	返回	操作成功返回字段大小		
举例		,		
virtual	longcdec	l BindToRecordset(CADORecordBinding * pBind)		
说明	绑定对象			
	参数	CADORecordBinding * pBind 数据库记录绑定对象指针		
	返回	操作成功返回 0		
举例				
virtual	longcdec	l NextRecordset ()		
说明	下一记录集	The Control of the Co		
	参数			
	返回	操作成功返回 0		
举例				
virtual	boolcdec	1 GetFieldValue ()		
说明	获取记录集	長的数据		
	参数	LPCTSTR lpFieldName 字段名称		
		X & bValue 字段数据值		
	返回	操作成功返回 true		
举例				
	void <u>cdec</u>	1 SetSPName(IPCISTR pszSpName)		
说明	获取存储过程名称			
	参数	LPCTSTR pszSpName 存储过程名称		
	返回)) 🤝		
举例				
		cl AddParamter(LPCTSTR pszName, ADOCG::ParameterDirectionEnum		
		ataTypeEnum Type, long lSize, _variant_t & vtValue)		
说明	添加参数			
	多数	LPCTSTR pszName 字段名称		
		ADOCG::ParameterDirectionEnum Direction 参数输入输出方向标记		
		ADOCG::DataTypeEnum Type 参数类型		
		ADOCG::DataTypeEnum Type 参数类型 long lSize 参数大小		
	返回	ADOCG::DataTypeEnum Type 参数类型		
举 例	返回	ADOCG::DataTypeEnum Type 参数类型 long lSize 参数大小		
举例 virtual		ADOCG::DataTypeEnum Type 参数类型 long lSize 参数大小		
		ADOCG::DataTypeEnum Type 参数类型 long lSize 参数大小 _variant_t & vtValue 参数值		
virtual	voidcdec	ADOCG::DataTypeEnum Type 参数类型 long lSize 参数大小 _variant_t & vtValue 参数值		
virtual	voidcdec 清除参数	ADOCG::DataTypeEnum Type 参数类型 long lSize 参数大小 _variant_t & vtValue 参数值		
virtual	voidcdec 清除参数 参数	ADOCG::DataTypeEnum Type 参数类型 long lSize 参数大小 _variant_t & vtValue 参数值		

virtual v	oid <u>cdecl</u>	GetParameterValue(LPCTSTR pszParamName, _variant_t & vtValue)				
说明	获取参数值					
	参数	LPCTSTR pszParamName 参数名称				
		_variant_t & vtValue 参数返回值				
	返回					
举例						
virtual 1	long <u>cdecl</u>	GetReturnValue()				
说明	获取数据库	返回值				
	参数					
	返回					
举例						
virtual b	oool <u>cdecl</u>	Execute(LPCTSTR pszCommand)				
说明	执行语句					
	参数	LPCTSTR pszCommand SQL 语句				
	返回	操作成功返回 ture,否则为 false				
举例						
virtual b	oool <u>cdecl</u>	ExecuteCommand(bool bRecordsst)				
说明	执行命令					
	参数 bool bRecordset 是否返回记录集					
	返回	操作成功返回 ture,否则为 false				
举例						
IDataBase	eSink //					
数据库钩	子接口					
模块	Databas	<u> </u>				
继承	IDataBa	seSink → IUnknownEx				
实现		W III N II				
	h ih	数据成员				
	名称					
virtual b	ooolcdecl	StartService(IUnknownEx * pIUnknownEx)				
说明	数据库模块启动					
	参数	IUnknownEx * pIUnknownEx 服务框架接口				
	返回	操作成功返回 ture,否则 false				
举例						
virtual b	oool <u>cde</u> cl	StopService(IUnknownEx * pIUnknownEx)				
说明	数据库模块					
	参数	IUnknownEx * pIUnknownEx 服务框架接口				

	返回	操作成功返回 ture,否则 false	
举例			
virtual b	oolcdecl	OnDataBaseRequest(BYTE cbThreadIndex, const NTY_DataBaseEvent &	
DataBaseI	Event, <mark>void</mark>	* pDataBuffer, WORD wDataSize)	
说明	数据事件操作处理		
	参数	BYTE cbThreadIndex 线程索引	
		const NTY_DataBaseEvent & DataBaseEvent 数据库事件	
		void * pDataBuffer 数据指针	
		WORD wDataSize 数据大小	
	返回	操作成功返回 ture,否则 false	
举例			

IDataBaseH	Engine					
数据库引擎	接口					
模块	Databas	e	a (V/)			
继承	IDataBaseEngine → IUnknownEx					
实现						
		数据成员				
	名称	C	说明			
		方法	10			
	ool <u>cdec</u> 启动服务	1 StartService(BYTE cbThrea	idCount)			
_						
<u> </u>	参数	/	数据库线程数量			
	返回	操作成功返回 ture,否则 false				
举例						
	bool _cdec1 StopService()					
说明	停止服务					
	参数					
	返回	操作成功返回 ture,否则 false				
举例						
virtual bo	ool <u>cdec</u>	l SetDataBaseSink(IUnknownE	Ex * pIUnknownEx)			
说明	设置数据事	4件操作处理钩子				
	参数	IUnknownEx * pIUnknownEx	钩子指针			
	返回	操作成功返回 ture,否则 false				
举例						
virtual vo	oid *cd	ecl GetQueueService(const I	IID & Guid, DWORD dwQueryVer)		
说明	获取接口					
	参数	onst IID & Guid	接口标记			
		DWORD dwQueryVer	查询版本			

	返回	操作成功返回内存指针,否则 Null
举例		

1	● 类设计					
CADOErro	r					
数据库错	误类					
模块	Databas	base				
继承	CAD0Er	ror → IADOError →	IUnknownE	Ex		
实现	IAD0Er	ror				
		数据	成员			
	Ŕ	名称			说明	
enADOErr	orType	m_enErrorType	错误代号	号		
CString	m_strErro	rDescribe;	错误信息	息		
		方	法		-(()b	
virtual	LPCTSTRc	decl GetErrorDescribe	()			
说明	获取错误抗	描述		46		
	参数					
	返回	描述错误的文本字符串	572 (7	()		
举例						
virtual	enADOErrorT	Typecdec1 GetFirorT	ype!)			
说明	获取错误约					
	参数					
		btt. 25 26 至年275 45 25 25				
<i>₩ I</i> nd	返回	错误类型的枚举值				
举例						
		ADOErrorType enError	Гуре, LPC	TSTR psz	Describe)	
说明	设置错误	设置错误				
	参数	enADOErrorType enErr	orType '	错误类型	的枚举值	
		LPCTSTR pszDescribe		描述错误	的文本字符串	
	返回					
举例						

CDataBase	CDataBase				
数据库连接并访问类					
模块	Database				
继承	CDataBase → IDataBase → IUnknownEx				
实现	IDataBase				
数据成员					
	名称				

CADOD	ADOD		烘告社会		
	r m_ADOErro		错误对象		
CString	m_strConne		连接字符串		
CString	-		错误信息		
DWORD	m_dwConnectCount;		重试次数		
DWORD	= '		错误时间		
const DWO	_	esumeConnectCount;	恢复次数		
const DWG	ORD m_dwR	esumeConnectTime;	恢复时间		
_Commandl	_	ommand;	命令对象		
_Records	etPtr m_DBR	ecordset;	记录集对象		
_Connect:	ionPtr m_DB	Connection;	数据库连接对象		
			方法		
virtual l	bool <u>cdec</u>	1 OpenConnection()			
说明	打开连接		^		
	参数				
	返回	操作成功返回 ture,否	则 flase		
举例		·			
virtual l	hool cdec	1 CloseRecordset()	25 (0)		
说明	关闭记录				
00 73		T			
	参数				
1	返回	操作成功返回 ture,否则 flase			
举例					
virtual boolcdecl CloseConnection()					
说明	兑明 关闭连接				
	参数		7		
	返回	操作成功返回 ture,否则 flase			
举例					
virtual l	bool cosc	TryConnectAgain(bo	ool bFocusConnect, CComError * pComError)		
说明	重新连接		, , , , , , , , , , , , , , , , , , , ,		
		1 115 0	10 Mildry 4 4 4 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7		
	参数	bool bFocusConnect	33,1112,033,11.13		
)F.F.	CComError * pComEr			
314 br.1	返回	操作成功返回 ture,否	则 flase		
举例					
		1 SetConnectionInfo			
		WORD wPort, LPCTSTR	szData,		
		TSTR szPass)			
说明	说明 设置信息				
	参数	LPCTSTR szIP	数据库服务器 IP 地址		
		LPCTSTR szPipeName	数据库命名实列通道		
		WORD wPort	端口		
		LPCTSTR szData	数据库名称		
		LPCTSTR szName	访问用户名		
		1	*******		

		LPCTSTR szPass 访问密码
	返回	操作成功返回 ture,否则 flase
<u></u>	70.	
virtual	bool <u>cdec</u>	l IsConnectError()
说明	判断是否连	接错误
	参数	
	返回	操作成功返回 ture,否则 flase
举例		
virtual	boolcdec	l IsRecordsetOpened ()
说明	判断是否打	开记录集
	参数	
	返回	操作成功返回 ture,否则 flase
举例		
virtual	boolcdec	l MoveToNext ()
说明	往下移动记	2录游标
	参数	47
	返回	操作成功返回 ture,否则 flase
举例		<u>~~~~~</u>
virtual	boolcdec	1 MoveToFirst()
说明	移到开头	
	参数	
	返回	操作成功返回 ture,否则 flase
举例		_ (()/5) = (
	\leftarrow	I IsEndRecordset ()
说明	是否结束	
	参数	
	返回	操作成功返回 ture,否则 flase
举例		
virtual		1 GetRecordCount ()
说明	获取数目	
	参数	
W /r.l	返回	操作成功返回记录数目
举例		I a l lol (I Domann D l
virtual		1 GetActualSize(LPCTSTR pszParamName)
说明	获取字段大	
	参数	LPCTSTR pszParamName 参数名称
举例	返回	操作成功返回字段大小
	long	DindToDooondgot (CADOD JD: - J:
virtual 说明	#定对象	1 BindToRecordset(CADORecordBinding * pBind)
かいり	沙儿们多	

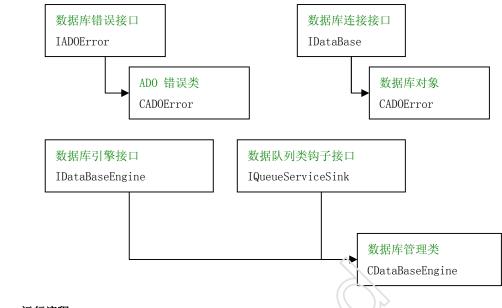
	参数	CADORecordBinding * pBind 数据库记录绑定对象指针
	返回	操作成功返回 0
举例		
virtual 3	long <u>cdec</u>	NextRecordset ()
说明	下一记录集	
	参数	
	返回	操作成功返回 0
举例		
virtual N	boolcdec	l GetFieldValue ()
说明	获取记录集	的数据
	参数	LPCTSTR lpFieldName 字段名称
		X & bValue 字段数据值
	返回	操作成功返回 true
举例		
virtual	void <u></u> cdec	l SetSPName(LPCTSTR pszSpName)
说明	获取存储过	程名称
	参数	LPCTSTR pszSpName 存储过程名称
	返回	$\langle \langle \langle \langle \rangle \rangle \rangle$
举例		
virtual	voidcded	AddParamter(LPCTSTR ps::Name, ADOCG::ParameterDirectionEnum
Direction	n, ADOCG::Da	ataTypeEnum Type, long 1Size, _variant_t & vtValue)
说明	添加参数	
	参数	LPCTS IR pszName 字段名称
		ADOCC::ParameterDirectionEnum Direction 参数输入输出方向标记
		ADOCG: DataTypeEnum Type 参数类型
		long lSize 参数大小
	4	_variant_t & vtValue 参数值
	返回	
举例		
virtual	voidcdec	ClearAllParameters()
说明	清除参数	
	参数	
	返回	
举例		
virtual v	void <u>cdec</u> l	GetParameterValue(LPCTSTR pszParamName, _variant_t & vtValue)
说明	获取参数值	<u> </u>
	参数	LPCTSTR pszParamName 参数名称
		_variant_t & vtValue 参数返回值
	返回	
举例		
virtual	long <u>cdec</u>	l GetReturnValue()

说明	获取数据	军返回值			
	参数				
	返回				
举例					
virtual	boolcde	cl Execute(LPCTSTR pszCommand)			
说明	执行语句				
	参数	LPCTSTR pszCommand SQL 语句			
	返回	操作成功返回 ture,否则为 false			
举例					
virtual	bool _cde	cl ExecuteCommand(bool bRecordset)			
说明	执行命令				
	参数	bool bRecordset 是否返回记录集			
	返回	操作成功返回 ture,否则为 false			
举例					
LPCTSTR	GetComErro	rDescribe(CComError & ComError)			
说明	获取错误	获取错误描述			
	参数	CComError & ComError 结误对象			
	返回	操作成功返回文本			
举例					
void Se	tErrorInfo	enADOErrorType ercrrciType, LPCTSTR pszDescribe)			
说明	设置错误				
	参数	enADOErrorType enErrorType 错误类型			
		LPCISTR pszDescribe 错误描述			
	返回				
举例					
		ightharpoons			

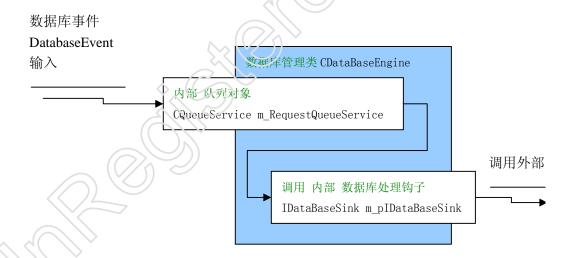
CDataBas	eEn yi ne				
数据库引	敬义.				
模块	Databas	Database			
继承	CDataBa	aseEngine → IDataBaseEn	gine → IUnknownEx		
实现	IDataBa	aseEngine			
		数据成	ე		
		名称	说明		
bool	m_bSe	rvice;	运行标志		
CQueueSe:	rvice m_Req	uestQueueService;	队列对象		
IDataBaseSink *m_pIDataBaseSink; 通知钩子					
		方法			
virtual l	bool <u></u> cdec	l StartService(BYTE cbTh	readCount)		
说明	启动服务				
	参数	BYTE cbThreadCount	数据库线程数量		

返回 操作成功返回 ture,否则 false 学例 学业版 学生 操作成功返回 ture,否则 false 学例 学生 操作成功返回 ture,否则 false 学例 操作成功返回 ture,否则 false 学例 操作成功返回 ture,否则 false 学例 操作成功返回 ture,否则 false 操作成功返回 操作成功返回内存指针,否则 false 操作成功返回 false 操作成功返回 false 操作数据标记				
Pub Pu		返回	操作成功返回 ture,否则 fa	lse
停止服务	举例			
参数	virtual	boolcde	ecl StopService()	
返回 操作成功返回 ture,否则 false 挙例	说明	停止服务		
Yellow		参数		
wirtual boolcdecl SetDataBaseSink(IUnknownEx * pIUnknownEx)		返回	操作成功返回 ture,否则 fa	lse
説明	举例			
多数 IUnknownEx * pIUnknownEx 钩子指针 返回 操作成功返回 ture,否则 false 挙例	virtual	boolcde	ecl SetDataBaseSink(IUnkno	wnEx * pIUnknownEx)
返回 操作成功返回 ture,否则 false 学例	说明	设置数据	事件操作处理钩子	
本例 virtual void *cdecl GetQueueService(const IID & Guid, DWORD dwQueryVer)		参数	IUnknownEx * pIUnknown	Ex 钩子指针
virtual void *cdecl GetQueueService(const IID & Guid, DWORD dwQueryVer) 成期 获取接口 参数 onst IID & Guid 接口标记 DWORD dwQueryVer 查询版本 返回 操作成功返回内存指针,否则 Null 举例 virtual voidcdecl OnQueueServiceSink(SiTE cbTbreadIndex, WORD wIdentifier, vo pBuffer, WORD wDataSize, DWORD dwInsertTime) 说明 实现队列钩子接口的函数 参数 BYTE cbThreadIndex 线程索引 WORD wIdentifier 事件数据标记 void * pRuffer 数据指针 WORD wDataSize 数据大小 DWORD dwInsertTime 加入时间 返回		返回	操作成功返回 ture,否则 fa	lse
	举例		·	^
参数 onst IID & Guid 接口标记 DWORD dwQueryVer 查询版本 返回 操作成功返回内存指针,否则 Nu!! 挙例 virtual voidcdecl OnQueueServiceSink(SiTE cbTbreadIndex, WORD wIdentifier, vopBuffer, WORD wDataSize, DWORD dwInsertTime) 求现队列钩子接口的函数 参数 BYTE cbThreadIndex 线程索引 WORD wIdentifier 事件数据标记 void * pBuffer 数据指针 WORD wDataSize 数据大小 DWORD dwInsertTime 加入时间 返回	virtual	void *o	cdecl GetQueueService(cons	t IID & Guid, DWORD dwQueryVer)
DWORD dwQueryVer 查询版本 返回 操作成功返回内存指针,否则 No.!! 举例 virtual voidcdecl OnQueueServiceSink(SiTE cbTbreadIndex, WORD wIdentifier, vo pBuffer, WORD wDataSize, DWORD dwInsertTime) 实现队列钩子接口的函数 参数 BYTE cbThreadIndex 线程索引 WORD wIdentifier 事件数据标记 void * pBuffer 数据指针 WORD wPataSize 数据大小 DWORD dwInsertTime 加入时间 返回	说明	获取接口		
返回 操作成功返回内存指针,否则 No.!! 挙例 virtual voidcdecl OnQueueServiceSink(SiTE cbTbreadIndex, WORD wIdentifier, vopBuffer, WORD wDataSize, DWORD dwInsertTime) 实现队列钩子接口的函数 参数 BYTE cbThreadIndex 线程索引 WORD wIdentifier 事件数据标记 void * pBuffer 数据指针 WORD wPataSize 数据大小 DWORD dwInsertTime 加入时间 返回		参数	onst IID & Guid	接口标记
学例 virtual voidcdecl OnQueueServiceSink(SiTE cbTbreadIndex, WORD wIdentifier, vo pBuffer, WORD wDataSize, DWORD dwInsertTime) 这明 字現队列钩子接口的函数 参数 BYTE cbThreadIndex 线程索引 WORD wIdentifier 事件数据标记 void * pBuffer 数据指针 WORD wPataSize 数据大小 DWORD dwInsertTime 加入时间 返回			DWORD dwQueryVer	查询版本
virtual voidcdec1 OnQueueServiceSink(SiTE cbTbreadIndex, WORD wIdentifier, vo pBuffer, WORD wDataSize, DWORD dwInsertTime) 实现队列钩子接口的函数 参数 BYTE cbThreadIndex 线程索引 WORD wIdentifier 事件数据标记 void * pBuffer 数据指针 WORD wPataSize 数据大小 DWORD dwInsertTime 加入时间		返回	操作成功返回内存指针,否	则 Noll
pBuffer, WORD wDataSize, DWORD dwInsertTime) 实现队列钩子接口的函数 参数 BYTE cbThreadIndex 线程索引 WORD wIdentifier 事件数据标记 void * pBuffer 数据指针 WORD wDataSize 数据大小 DWORD dwInsertTime 加入时间 返回	举例		<u> </u>	? (//)
说明 实现队列钩子接口的函数 参数 BYTE cbThreadIndex 线程索引 WORD wIdentifier 事件数据标记 void * pRuffer 数据指针 WORD wPataSize 数据大小 DWORD dwInsertTime 加入时间	virtual	voidcde	cl OnQueueServiceSink(BYTE	EcbTbreadIndex, WORD wIdentifier, void *
多数 BYTE cbThreadIndex 线程索引 WORD wIdentifier 事件数据标记 void * pBuffer 数据指针 WORD wPataSize 数据大小 DWORD dwInsertTime 加入时间	pBuffer,	WORD wDat	taSize, DWORD dwInsertTime	
WORD wIdentifier 事件数据标记 void * pBuffer 数据指针 WORD wPataSize 数据大小 DWORD dwInsertTime 加入时间	说明	实现队列	钩子接口的函数	7
void * pRuffer 数据指针 WORD wPataSize 数据大小 DWORD dwInsertTime 加入时间		参数	BYTE cbThreadIndex	线程索引
WORD wPataSize 数据大小 DWORD dwInsertTime 加入时间 返回			WORD wIdentifier	事件数据标记
DWORD dwInsertTime 加入时间			void * pBuffer	数据指针
返回			WORD wPataSize	数据大小
			DWORD dwInsertTime	加入时间
举例		返回		
	举例			

● 类继承图



3.3.3 运行流程



3.4 定时器引擎 TimerEngine 模块设计

3.4.1 基本设计概念

内部启动工作线程 TimerThread,工作线程周期检测定时请求子项 tagTimerItem,每个定时请求项记录当前时间计数 dwTimeLeave,处理一次,将时间计数减去时间间隔 dwTimerSpace,如果为 0 时,发生定时器事件 TimerEvent,通知到调度引擎,并将时间计数复位为定时时间再次倒计时.

3.4.2 类与接口设计

ITimerEngine	ITimerEngine				
定时器引擎接	[€] □				
模块	TimerEngine				
继承	TimerEngine → IUnknownEx				
实现					

			数据成	员		
	名称			说明		
		<u> </u>				
			方法			
virtual wParam)	boolcdecl	SetTimer(W	ORD wTimerID,	DWORD dwElapse,	DWORD dwRepeat,	WPARAM
说明	设置定时器	i.				
	参数	WORD wTime	rID	时间标记		
		DWORD dwE1	apse	时间间隔		
		DWORD dwRe	peat	重复次数		
		WPARAM wPa	ram	绑定参数	ζ	
	返回	操作成功返	回 ture,否则 fa	lse		
举例						
virtual	boolcdec	l KillTimer	(WORD wTimerI	D)		
说明	删除定时器	Ţ				
	参数	WORD wTime	rID	时间标记	()	
	返回	操作成功返	回 ture,否则 fa	lse		
举例			_	(0)		
virtual	boolcdec	l KillAllTin	mer()			
说明	删除全部定	时器	0 C			
	参数		5/1/2	ij.		
	返回	操作成功返	回 ture,否则 fa	lse		
举例						
		72	/			
ITimerE	ngineManager	$\overline{}$				
定时器引	擎领推接了					

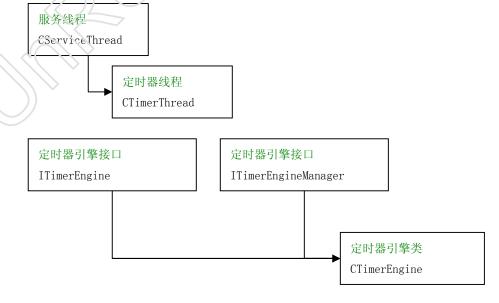
ITimerEn	gineManager			
定时器具	擎急性接入	\nearrow		
模块	TimerEr	ngine		
继承	ITimer	EngineManage	er → IUnknownEx	
实现				
			数据成员	
	名称			说明
			方法	
virtual	bool <u>cdec</u>	l StartServi	ice()	
说明	启动服务			
	参数			
返回 操作成功返回 ture,否则 false				
举例				
virtual	bool <u>cdec</u>	StopServic	ce()	
说明	停止服务			

	参数					
	返回	操作成功返回 ture,否则 false				
举例		·				
virtual b	oool <u>cdec</u>	l SetTimerEngineSink(IUnknownEx * pIUnknownEx)				
说明	设置时间引擎钩子接口					
	参数	IUnknownEx * pIUnknownEx 调度钩子接口指针				
	返回 操作成功返回 ture,否则 false					
举例						

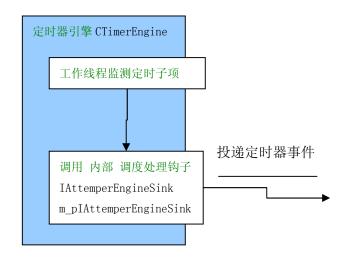
CTimerEng			
定时器引擎	擎		
模块	TimerEı	ngine	
继承	CTimer	Engine → ITimerEngine,	ITimerEngineManager → IUnknownEx
实现	ITimer	Engine ,ITimerEngineMana	ager (V)
		数据成	员
	2	名称	说明
DWORD	m_dwT	imerSpace;	的间间隔
bool	m_bSe	rvice;	运行标志
DWORD	m_dwT	imePass;	经过时间
DWORD	m_dwT	imeLeave;	倒计时间
CTimerIte	emPtr m_Tim	erItemFree;	空闲数组
CTimerIte	emPtr m_Tim	erItemActive:	活动数组
CThreadLo	ock m_Thr	eadLock;	线程锁
CTimerThr	read m_Tim	erThread;	定时器线程
CQueueSer	rviceFvent m	n_AttemperEvent;	通知组件
		方法	
virtual b	oo! _cdecl	SetTimer(WORD wTimerID,	
virtual b wParam)	oo! _cdec!	SetTimer(WORD wTimerID,	
	oo! _cdecl 设置定时器		
wParam)			
wParam)	设置定时器		DWORD dwElapse, DWORD dwRepeat, WPARAM
wParam)	设置定时器	WORD wTimerID	, DWORD dwElapse, DWORD dwRepeat, WPARAM 时间标记
wParam)	设置定时器	WORD wTimerID DWORD dwElapse	,DWORD dwElapse,DWORD dwRepeat,WPARAM 时间标记 时间间隔
wParam)	设置定时器	WORD wTimerID DWORD dwElapse DWORD dwRepeat	DWORD dwElapse, DWORD dwRepeat, WPARAM 时间标记 时间间隔 重复次数 绑定参数
wParam)	设置定时器 参数	WORD wTimerID DWORD dwElapse DWORD dwRepeat WPARAM wParam	DWORD dwElapse, DWORD dwRepeat, WPARAM 时间标记 时间间隔 重复次数 绑定参数
wParam) 说明	设置定时器 参数 返回	WORD wTimerID DWORD dwElapse DWORD dwRepeat WPARAM wParam	,DWORD dwElapse,DWORD dwRepeat,WPARAM 时间标记 时间间隔 重复次数 绑定参数
wParam) 说明 举例	设置定时器 参数 返回	WORD wTimerID DWORD dwElapse DWORD dwRepeat WPARAM wParam 操作成功返回 ture,否则 fa	,DWORD dwElapse,DWORD dwRepeat,WPARAM 时间标记 时间间隔 重复次数 绑定参数
wParam) 说明 举例 virtual b	设置定时器 参数 返回	WORD wTimerID DWORD dwElapse DWORD dwRepeat WPARAM wParam 操作成功返回 ture,否则 fa	,DWORD dwElapse,DWORD dwRepeat,WPARAM 时间标记 时间间隔 重复次数 绑定参数
wParam) 说明 举例 virtual b	设置定时器 参数 返回 poolcdec 删除定时器	WORD wTimerID DWORD dwElapse DWORD dwRepeat WPARAM wParam 操作成功返回 ture,否则 fa	DWORD dwElapse, DWORD dwRepeat, WPARAM 时间标记 时间间隔 重复次数 绑定参数 alse ID)



● 类继承图



3.4.3 运行流程



3.5 调度引擎 AttemperEngine 模块设计

3.5.1 基本设计概念

内部管理数据队列,对外提供接口,接收外部输入网络事件,数据库事件,定时器事件等数据,对这些事件做初步处理,并调度分发传递到外部调度引擎钩子,让外部实现逻辑处理

3.5.2 类与接口设计

IAttempe:	rEngine				
调度引擎	钩子接口				
模块	Attempe	attemperEngine			
继承	IAttemperEngine → IUnknownEx				
实现					
		数据成员			
	名称	说明			
	(())				
		方法			
virtual l	boocdec	1 StartService(BYTE cbThreadCount)			
说明	启动服务				
	参数	BYTE cbThreadCount 线程数量			
	返回	操作成功返回 ture,否则 false			
举例					
virtual l	bool <u>cdec</u>	1 StopService()			
说明	停止服务				
	参数				
	返回	操作成功返回 ture,否则 false			
举例					
virtual 1	boolcdec	l SetSocketEngine(IUnknownEx * pIUnknownEx)			
说明	设置网络引	掌接口			

	参数	IUnknownEx * pIUnknownEx 网络引擎接口指针
	返回	操作成功返回 ture,否则 false
举例		
virtual	bool <u></u> cdec	el SetAttemperEngineSink(IUnknownEx * pIUnknownEx)
说明	设置调度引	引擎钩子接口
	参数	IUnknownEx * pIUnknownEx 调度钩子接口指针
	返回	操作成功返回 ture,否则 false
举例		
virtual	void *cd	decl GetQueueService(const IID & Guid, DWORD dwQueryVer)
说明	获取数据图	人列接口
	参数	const IID & Guid 接口标记
		DWORD dwQueryVer 接口版本
	返回	操作成功返回数据队列接口
举例		
	ı	
	wEnginoCipl	
IAttempe	renginesink	

IAttemper	EngineSink			
调度钩子挂	₹ 钩子接口			
模块	Attempe	erEngine		
继承	IAttemp	perEngineSink -+ IUnknownEx	7	
实现				
		数据成员		
	名称		说明	
		方法		
virtual b		StartService(IUnknownEx * p	IUnknownEx)	
说明	调度模块启	动		
	参数	IUnknownEx * pIUnknownEx	服务框架接口	
	返回	操作成功返回 ture,否则 false		
举例				
virtual b	oolcdec	l StopService(IUnknownEx * pI	UnknownEx)	
说明	调度模块关	闭		
	参数	IUnknownEx * pIUnknownEx	服务框架接口	
	返回	操作成功返回 ture,否则 false		
举例				
virtual b	ooolcdec	OnAttemperEvent (BYTE cbThr	readIndex,WORD wIdentifier, void *	
pBuffer,	WORD wDatas	Size, DWORD dwInsertTime)		
说明	调度事件操作处理			
	参数	BYTE cbThreadIndex	线程索引	
		WORD wIdentifier	调度事件标记	
		void * pDataBuffer	数据指针	

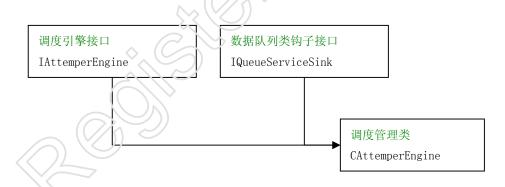
		WORD wDataSize	数据大小
		DWORD dwInsertTime	插入时间
	返回	操作成功返回 ture,否则 false	
举例			
virtual	boolcd	ecl OnEventTimer(BYTE cbThreadIndex, W	ORD wTimerID, WPARAM
wBindPar	am)		
说明	定时器事件		
	参数	BYTE cbThreadIndex	线程索引
		WORD wTimerID	定时器标记
		WPARAM wBindParam	绑定参数
	返回	操作成功返回 ture,否则 false	
举例			
virtual	boolcdec	1 OnEventDataBase(BYTE cbThreadIndex, vo	<pre>id * pDataBuffer, WORD</pre>
wDataSiz	e, NTY_Data	BaseEvent * pDataBaseEvent)	
说明	数据库事件		
	参数	BYTE cbThreadIndex	线程索引
		void * pDataBuffer	数据指针
		WORD wDataSize	数据大小
		NTY_DataBaseEvent * pDataBaseEvent	数据库事件
	返回	操作成功返回 ture,否则 false	
举例			
virtual b	oool <u>_</u> cdecl	OnEventSocketAccept (BYTE cbThreadIndex,	NTY_SocketAcceptEvent*
pSocketA	cceptEvent)		
说明	网络应答事	件	
	参数	BYTE cbThreadIndex	线程索引
		NTY_SocketAcceptEvent * pSocketAcceptE	vent 网络应答事件
	返回	操作成功返回 ture,否则 false	
举例		>	
virtual l	boo!cdecl	OnEventSocketRead(BYTE cbThreadIndex, CM	D_Command Command, void
* pDataB	uffer, WORD	wDataSize, NTY_SocketReadEvent * pSocket	etReadEvent)
说明	网络应答事	4件	
	参数	BYTE cbThreadIndex	线程索引
		CMD_Command Command	网络命令
		void * pDataBuffer	数据指针
		WORD wDataSize	数据大小
		NTY_SocketReadEvent * pSocketReadEvent	网络读取事件
	返回	操作成功返回 ture,否则 false	
举例			
	boolcdec loseEvent)	1 OnEventSocketClose(BYTE cbThreadIndex,	NTY_SocketCloseEvent *
说明	网络应答事	· 件	

	参数	BYTE cbThreadIndex 线程索引
		NTY_SocketCloseEvent * pSocketCloseEvent 网络读取事件
	返回	操作成功返回 ture,否则 false
举例		

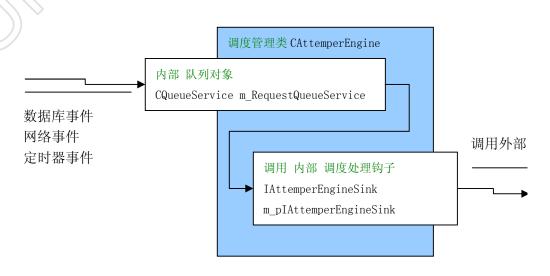
_	 		
CAttemper	rEngine		
调度引擎	类		
模块	AttemperEngine		
继承	术 CAttemperEngine → IAttemperEngine → IUnknownEx		
实现	IAttemp	perEngine	
		数据成员	
		名称	
bool	m_bSe	ervice; 运行标志	
CQueueSei	rvice m_Req	uestQueueService; 以列对象	
ITCPSocke	etEngine* m	n_pITCPSocketEngine 网络引擎	
IAttempe	rEngineSink	* m_pIAttemperEngineSink; 挂接接口	
		方法。	
virtual b	ooolcdec	1 StartService (BYTE chimeadCount)	
说明	启动服务	\$\sqrt{\sq}}}}}}}\sqrt{\sqrt{\sqrt{\sqrt{\sqrt{\sqrt{\sqrt{\sqrt{\sqrt{\sq}}}}}}}}\sqrt{\sqrt{\sqrt{\sqrt{\sqrt{\sqrt{\sqrt{\sq}}}}}}}}\signitiqnitite\sent\sign{\sqrt{\sqrt{\sq}}}}}}}\sqrt{\sqrt{\sqrt{\sqrt{\sqrt{\sqrt{\sqrt{\sq}}}}}}\sqrt{\sintiting{\sq}}}}}}}}\sintite\sent\sintitin}}\signitite\sent\sintitite{\sintitinitita}}}}	
	参数	BYTE cbThreadCount 线程数量	
	返回	操作成功返回 ture,否则 false	
举例			
virtual b	oool _cdec	i StopService()	
说明	停止服务		
	参数		
	返回	操作成功返回 ture,否则 false	
举例			
virtual h	posil cdec	1 SetSocketEngine(IUnknownEx * pIUnknownEx)	
说明	设置网络引		
9274			
	参数	IUnknownEx * pIUnknownEx 网络引擎接口指针	
举例	返回	操作成功返回 ture,否则 false	
	1 1		
virtual b		1 SetAttemperEngineSink(IUnknownEx * pIUnknownEx)	
说明	攻直调度 5	擎钩子接口	
	参数	IUnknownEx * pIUnknownEx 调度钩子接口指针	
	返回	操作成功返回 ture,否则 false	
举例			
virtual v	void *cd	ecl GetQueueService(const IID & Guid, DWORD dwQueryVer)	
说明	获取数据队	人列接口	
	l .		

	参数	const IID & Guid	接口标记
		DWORD dwQueryVer	接口版本
	返回	操作成功返回数据队列接口	I
举例			
virtual v	oid <u>cdecl</u>	OnQueueServiceSink(BYTE	cbThreadIndex,WORD wIdentifier, void *
pBuffer,	WORD wData	Size, DWORD dwInsertTime)	
说明	实现队列钩]子接口的函数	
	参数	BYTE cbThreadIndex	线程索引
		WORD wIdentifier	事件数据标记
		void * pBuffer	数据指针
		WORD wDataSize	数据大小
		DWORD dwInsertTime	加入时间
	返回		
举例			

● 类继承图



3.5.3 运行流程



3.6.1 基本设计概念

接收请求事件,内部启动线程,线程运行异步获取请求事件,再调用钩子处理事件.异步方式采用 window 窗口消息实现.

3.6.2 类与接口设计

		·		
	onismEngine			
异步引擎	接口			
模块	AsynchronismEngine			
继承	IAsynch	nronismEngin	e → IUnknownE	Ĉx .
实现				
			数据成员	
	名称			说明
			方法	
virtual	boolcdec	l StartServi	ce ()	(()>
说明	启动服务			
	参数			18 (O)
	返回	操作成功返	回 ture,否则 false	
举例			\sim	(6/3)
virtual	bool cdec	l StopServic	e ()	
 说明	停止服务		\$ (S)	
			<u> </u>	
	返回	揭作成功法		
举 例	心固	床下风切足	ET ture, ET WI Tarse	
	haal die	Ingura	WORD WPag	questID, void * const pBuffer, WORD
		ex * plinkno		questib, voiu * const phurrer, word
说明	数据事件操	\leftarrow	(WILLY)	
\ \ \			. TD	N# JN I= N=
	参数	WORD wRequ		请求标记
			st pBuffer	数据库指针
		WORD wData		数据大小
	海回		* pIUnknownEx	
举例	返回	深下风切区	回 ture,否则 false	
, , , ,	hool ada	1 Dogistar A	um ahnani amEr wi	nacial (Hiphnamer + - Hill F.)
virtual 说明	注册调度钩		Synchron1smeng1	neSink(IUnknownEx * pIUnknownEx)
近明	1土川 炯及4	1		
	参数		* pIUnknownEx	
	返回	操作成功返回 ture,否则 false		
举例				
virtual		l UnRegister	AsynchronismEn	gineSink(IUnknownEx * pIUnknownEx)
说明	取消注册			
	参数	IUnknownEx	* pIUnknownEx	引擎处理钩子

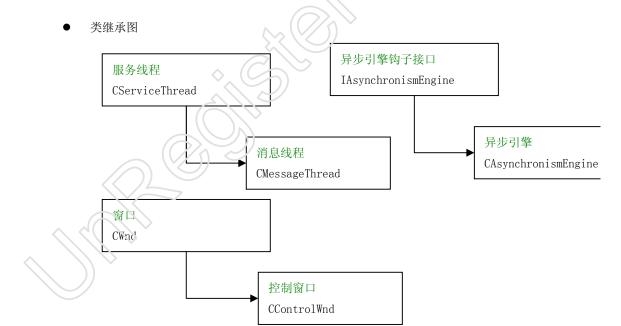
	返回	操作成功返回 ture,否则 false
举例		

	ronismEngin	e9111K	<u> </u>		
异步引擎	擎钩子接口				
模块	Asynch	AsynchronismEngine			
继承	IAsyno	chronismEngineSink → 1	[UnknownEx		
实现					
		数据	成员		
	名称		说明		
		方	法		
virtual	boolcde	cl OnAsynchronismEngine	Start()		
说明	启动异步	引擎事件调用该方法			
	参数		a (V/s)		
	返回	操作成功返回 ture,否则	false		
举例			. (0)		
virtual	boolcde	cl OnAsynchronismEngine	Stop()		
说明	停止异步	引擎事件调用该方法			
	参数)		
	返回	操作成功返回 ture,否则	false		
举例					
virtual	boolcde	cl OnAsynchronismReques	st(WORD wRequestID, void * pBuffer, WORD		
wDataSi	ze)				
说明	异步请求	异步请求			
	参数	WORD wRequestID	请求标记		
		void * const pBuffer			
		WORD wDataSize	数据大小		
	返回	操作成功返回 ture,否则			
举例		,,,,,,			

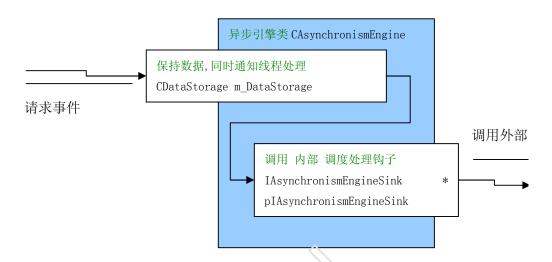
CAsynchronismEngine			
异步引擎类			
模块	模块 AsynchronismEngine		
继承	继承 CAsynchronismEngine → IAsynchronismEngine → IUnknownEx		
实现	y 现 IAsynchronismEngine		
数据成员			

		名称	说明
bool	m_bSe	rvice;	服务标志
ВҮТЕ	m_cbB	uffer[MAX_QUEUE_PACKET];	数据缓冲
CAsynchronis	异步钩子		
CControlWnd	控制窗口		
CThreadLock	m_Thr	eadLock;	线程同步
CDataStorage	m_Dat	aStorage;	数据存储
CMessageThre	ad m_Mes	sageThread;	线程组件
		方法	
virtual bool	cdecl Sta	artService()	
说明	启动服务		
	参数		
	返回	操作成功返回 ture,否则 false	\wedge
举例			
virtual bool	cdecl Sto	ppService()	
说明	停止服务		
	参数	(()	7
	返回	操作成功返回 ture,否则 teise	
举例		X (Q)	
virtual bool	cdecl Ir	nsertRequest(WCFD wRequestID, void *	const pBuffer, WORD
wDataSize, I	UnknownEx *	pIUnknownEx)	
说明	数据事件操	作处理	
	参数	WORD wRequestID 请求标记	
		void * const pBuffer 数据库指针	
		WORD wDataSize 数据大小	
		IUnknownEx * pIUnknownEx 引擎处理钩	子
	返回	操作成功返回 ture,否则 false	
举例			
virtual bool	cdecl Reg	gisterAsynchronismEngineSink(IUnknown	Ex * pIUnknownEx)
说明	注册调度钩	子	
	参数	IUnknownEx * pIUnknownEx 引擎处理钩	子
	返回	操作成功返回 ture,否则 false	
举例			
virtual bool	cdecl UnF	RegisterAsynchronismEngineSink(IUnkno	wnEx * pIUnknownEx)
说明	取消注册		
	参数	IUnknownEx * pIUnknownEx 引擎处理钩	
	返回	操作成功返回 ture,否则 false	
举例			
bool OnMessa	geThreadStaı	rt ()	
说明	线程启动		
	参数		

	返回	操作成功返回 ture,否则 false
举例		
bool OnMessa	geThreadStop) ()
说明	线程停止	
	参数	
	返回	操作成功返回 ture,否则 false
举例		
void OnAs	ynchronismRe	equest(WORD wRequestID, IAsynchronismEngineSink *
pIAsynchroni	smEngineSink	;)
说明	异步请求	
	参数	WORD wRequestID 请求标记
		IAsynchronismEngineSink*pIAsynchronismEngineSink 异步引
		擎钩子接口
	返回	操作成功返回 ture,否则 false
举例		



3.6.3 运行流程



3.7 服务引擎 ServiceEngine 模块设计

3.7.1 基本设计概念

管理定时器引擎,数据库引擎,调度引擎,网络引擎,异步引擎,并且对外提供查询引擎接

3.7.2 类与接口设计

•	按口及1	
IService	Engine	
服务引擎	接口	\$\langle \(\langle \)
模块	Service	Engine
继承	IServi	ceEngine → IUnkrownEx
实现		
		数据成员
	名称	说明
($\mathcal{C}(\mathcal{N})$	方法
virtual	bool _cdec	StartService(BYTE cbDBThreadCount = 12,
	BYTE cbAtt	temperThreadCount = 1)
说明	启动服务	
	参数	BYTE cbDBThreadCount 数据库线程数量
		BYTE cbAttemperThreadCount 调度引擎线程数量
	返回	操作成功返回 ture,否则 false
举例		
virtual	boolcdec	el StopService()
说明	停止服务	
	参数	
	返回	操作成功返回 ture,否则 false
举例		
virtual	bool <u>cdec</u>	el SetEventService(IUnknownEx * pIUnknownEx)

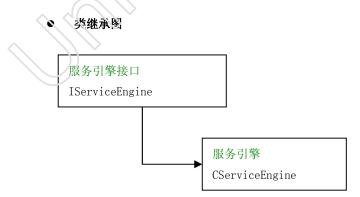
说明	设置事件服务接口			
	参数	IUnknownEx * pIUnknownEx 事件服务接口指针		
	返回	操作成功返回 ture,否则 false		
举例				
virtual	bool <u>cdec</u>	l SetDataBaseSink(IUnknownEx * pIUnknownEx)		
说明	设置数据库钩子接口			
	参数	IUnknownEx * pIUnknownEx 数据库钩子接口指针		
	返回	操作成功返回 ture,否则 false		
举例				
virtual	bool <u>cdecl</u> SetAttemperEngineSink(IUnknownEx * pIUnknownEx)			
说明	设置调度钩	3子接口		
	参数	IUnknownEx * pIUnknownEx 调度钩子接口指针		
	返回	操作成功返回 ture,否则 false		
举例				
virtual	boolcdec	l RegisterAsynchronismEngineSink(IUnknownfx * pIUnknownEx)		
说明	注册异步处理钩子接口			
	参数	IUnknownEx * pIUnknownEx 异步调用钩子接口指针		
	返回	操作成功返回 ture,否则 false		
举例				
virtual	boolcdec	l InitServiceEngine (WORD wListenPort, WORD wMaxSocketItem)		
说明	设置网络引擎接口			
	参数	WORD wListenPort 监听端口		
		WORD wMaxSockecitem 最大连接数目		
	返回	操作成功返回 ture,否则 false		
举例				
		ec) GetTimerEngine(const IID & Guid, DWORD dwQueryVer)		
说明	获取定时器	接口		
	参数	const IID & Guid 接口标记		
		DWORD dwQueryVer 接口版本		
	返回	操作成功返回接口指针		
举例				
virtual		ecl GetDataBaseEngine(const IID & Guid, DWORD dwQueryVer)		
说明	获取数据库	· 引擎接口		
	参数	const IID & Guid 接口标记		
		DWORD dwQueryVer 接口版本		
	返回	操作成功返回接口指针		
举例				
virtual	void *cd	ecl GetAttemperEngine(const IID & Guid, DWORD dwQueryVer)		
说明	获取调度引	擎接口		

		DWORD dwQueryVer 接口版本		
	返回	操作成功返回接口指针		
举例	21	PKII 784 22 - 134 71		
virtual	void * cd	ecl GetTCPSocketEngine(const IID & Guid, DWORD dwQueryVer)		
	参数	const IID & Guid 接口标记		
		DWORD dwQueryVer 接口版本		
	返回	操作成功返回接口指针		
举例				
virtual	void *cd	ecl GetAsynchronismEngine(const IID & Guid, DWORD dwQueryVer)		
说明	获取异步引	 擎接口		
	参数	const IID & Guid 接口标记		
		DWORD dwQueryVer 接口版本		
	返回	操作成功返回接口指针		
举例				
virtual	void *cde	ecl GetDataBaseQueueService(const 11D & Guia, DWORD dwQueryVer)		
说明	获取数据库	引擎数据队列接口		
	参数	const IID & Guid 接口标记		
		DWORD dwQueryVer 接口版本		
	返回	操作成功返回接口指针		
举例				
virtual	void *cde	ecl GetAttemperQueueService(const IID & Guid, DWORD dwQueryVer)		
说明	获取调度引	擎数据队列接口		
	参数	const IID & Guid 接口标记		
		DWORD dwQueryVer 接口版本		
	返回	操作成功返回接口指针		
举例 🔷	(4(//	>		
virtual	oskocdec	l IsService()		
说明	判断服务状态			
	参数			
	返回	运行中返回 ture,否则 false		
举例				
virtual	boolcdec	1 ControlService(void * pBuffer, WORD wDataSize)		
说明	外部控制			
	参数	void * pBuffer 数据指针		
		WORD wDataSize 数据大小		
	返回	运行中返回 ture,否则 false		
举例				

CServiceE	Engine				
服务引擎					
模块	ServiceI	ServiceEngine			
继承	CServic	CServiceEngine → IServiceEngine → IUnknownEx			
实现	IServiceEngine, IUnknownEx				
		数据成员			
		名称	说明		
bool	n	n_bService;	运行标志		
CTimerEngine m_TimerEngine; 定时器引擎					
CDataBaseEngine m_DataBaseEngine; 数据库引擎			数据库引擎		
CAttemper	CAttemperEngine m_AttemperEngine; 调度引擎				
CTCPSocke	etEngine m	n_TCPSocketEngine;	网络引擎		
CAsynchro	onismEnginer	m_AsynchronismEngine;	异步引擎		
		方法			
virtual b	ooolcdec	l StartService(BYTE cbDBThreadCoun	nt = 12,		
	BYTE cbAtt	emperThreadCount = 1)			
说明	启动服务				
-	参数	BYTE cbDBThreadCount 数据库	线程数量		
		BYTE cbAttemperThreadCount 调度	学线程数量		
-	返回	操作成功返回 ture,否则 talse	/		
举例		\$. (C)			
virtual b	oolcdec	1 StopService			
说明	停止服务				
-	参数				
_	返回	操作成功返回 ture,否则 false			
举例		Wife State and the state and t			
	vaal day	Catevant Canviac (Halmanner & all	Intro own Ex		
说明	设置事件服	SetEventService(IUnknownEx * pIU 文字口	JIIKHOWHEX)		
DT 1973	<u> </u>				
	参数	IUnknownEx * pIUnknownEx	事件服务接口指针		
	返回	操作成功返回 ture,否则 false			
举例					
virtual b	oolcdec	l SetDataBaseSink(IUnknownEx * pIU	JnknownEx)		
说明	设置数据库	钩子接口			
-	参数	IUnknownEx * pIUnknownEx	数据库钩子接口指针		
-	返回	操作成功返回 ture,否则 false			
举例					
virtual b	oolcdec	l SetAttemperEngineSink(IUnknownE	x * pIUnknownEx)		
说明	设置调度钩				
	参数	IUnknownEx * pIUnknownEx			
-	返回	操作成功返回 ture,否则 false	州汉刊(文刊)目刊		
举例	心口				
午四					

virtual l	bool cdec	1 RegisterAsynchronismEngineSink(IUnknownEx * pIUnknownEx)
	参数	IUnknownEx * pIUnknownEx 异步调用钩子接口指针
	返回	操作成功返回 ture,否则 false
举例		,
virtual l	bool cdec	1 InitServiceEngine(WORD wListenPort, WORD wMaxSocketItem)
	设置网络引	
		WORD wListenPort 监听端口
	多奴	WORD wMaxSocketItem 最大连接数目
	返回	操作成功返回 ture,否则 false
举例	21	
	void * cd	ecl GetTimerEngine(const IID & Guid, DWORD dwQueryVer)
说明	获取定时器	
00.73		
	参数 	const IID & Guid 接口标记
	返回	DWORD dwQueryVer 接口版本
<u>举例</u>	返 坦	操作成功返回接口指针
		T(T, 0 Set 1 DWODD 1 Over Ver)
		ecl GetDataBaseEngine(coxst 1150 & Guid, DWORD dwQueryVer)
说明	获取数据库	
	参数	const IID & Guid 接口标记
		DWORD dwQueryVer 接口版本
W 61	返回	操作成功返回接口指针
举例		
	//_	ec) GetAttemperEngine(const IID & Guid, DWORD dwQueryVer)
说明	获取调度引	
	参数	const IID & Guid 接口标记
		DWORD dwQueryVer 接口版本
	返回	操作成功返回接口指针
举例		
virtual	void *cd	ecl GetTCPSocketEngine(const IID & Guid, DWORD dwQueryVer)
说明	获取网络引	擎接口
	参数	const IID & Guid 接口标记
		DWORD dwQueryVer 接口版本
	返回	操作成功返回接口指针
举例		
virtual	void *cd	ecl GetAsynchronismEngine(const IID & Guid, DWORD dwQueryVer)
说明	获取异步引	擎接口
	参数	const IID & Guid 接口标记
		DWORD dwQueryVer 接口版本
	返回	操作成功返回接口指针

举例					
	virtual void *cdecl GetDataBaseQueueService(const IID & Guid, DWORD dwQueryVer)				
说明		获取数据库引擎数据队列接口			
	参数	const IID & Guid 接口标记			
		DWORD dwQueryVer 接口版本			
	返回	操作成功返回接口指针			
举例					
virtual	void *cd	ecl GetAttemperQueueService(const IID & Guid, DWORD dwQueryVer)			
说明	获取调度引	擎数据队列接口			
	参数	const IID & Guid 接口标记			
		DWORD dwQueryVer 接口版本			
	返回	操作成功返回接口指针			
举例					
virtual	boolcdec	l IsService()			
说明	判断服务状	态			
	参数	47			
	返回	运行中返回 ture,否则 false			
举例		5% (2)			
virtual	virtual boolcdecl ControlService(void * pRuffer, WORD wDataSize)				
说明	外部控制				
	参数	void * pBuffer 数据指针			
		WORD wDataSize 数据大小			
	返回	运行中返回 ture,否则 false			
举例					
					



3.7.3 运行流程

