ModuleName	FunctionNa	me	ClassName
UBCItime.cpp	~BCITIME())	BCITIME
Function			
BCITIME::~B0	CITIME()		
Purpose			
the destructor f	for the BCITIME of	pject	
Parameters			
N/A			
Returns			
N/A			
Group	Responsible	EMailAddress	
Albany	Gerv	schalk@wadsworth	h ora
Albally	Oct v	Schark@wausWorth	II.01g

ModuleName	FunctionNa	me	ClassName
UBCItime.cpp	BCITIME()		BCITIME
Function			
BCITIME::B0	CITIME()		
Purpose			
the constructo	or for the BCITIME (object	
Parameters			
N/A			
Returns			
N/A			
Croun	Dognonoible	EMailAddress	
Group	Responsible) org
Albany	Gerv	schalk@wadsworth	1.01g

ModuleName	FunctionNam	ne	ClassName
UBCItime.cpp	GetBCItime_n	ns()	BCITIME
Function			
unsigned short BO	CITIME::GetBCIti	ime_ms()	
Purpose			
gets the current ti	me from Windows	' high-performance tim	ers
Parameters			
Returns			
	en hit value for the	e current time in ms	
		6ms, or about 6.5 seco	nds)
Group F	Responsible	EMailAddress	
	Gerv	schalk@wadsworth.	org

ModuleName	FunctionName	ClassName	
UBCItime.cpp	TimeDiff	BCITIME	
unction			
unsigned short BC	CITIME::TimeDiff		
Purpose			
	Perence between two times (i.e.,	time2-time1)	
takes roll-over into	o account (in case time2 < time	e1)	
Parameters			
time1, time2 - two	16 bit integers		
Returns			
	me2-time1 >= 0 or time2-time	1+65536, if time2-time1 < 0	
,		,	

Group	Responsible	EMailAddress
Albany	Gerv	schalk@wadsworth.org

ModuleName	FunctionName		ClassName
UCoreMessage.cpp	~COREMESSA	GE()	COREMESSAGE
Function			
COREMESSAGE:	:~COREMESSAC	GE()	
Purpose			
the destructor for t	he COREMESSA	GE object	
Parameters			
N/A			
Returns			
N/A			
		I 	
	esponsible	EMailAddress	
Albany Ge	erv	schalk@wadsworth	n.org

ModuleName	FunctionNar	ne	ClassName	e
UCoreMessage.cpp	COREMESS	AGE()	COREMES	SSAGE
Function	·		·	
COREMESSAGE:	:COREMESSA	GE()		
Purpose				
the constructor for	the COREMES	SAGE object		
Parameters				
N/A				
Returns				
N/A				
Group Re	esponsible	EMailAddress		
	erv	schalk@wadswo	rth org	
r Hoally GC	/1 ¥	Schankw wads wo	111.015	

ModuleName	FunctionName		ClassName
UCoreMessage.cpp	GetBufPtr()		char *COREMESSAGE
Function			
char *COREMESSA	GE::GetBufPtr()		
Purpose			
Gets a pointer to the	GetLength() byte	es data in the coremes	ssage
Parameters			
N/A			
Returns			
pointer to the data			
Group Res	ponsible	EMailAddress	
Albany Gerv		schalk@wadsworth.c	org

ModuleName	FunctionNar	ne	ClassName
UCoreMessage.cp	op GetDescripto	r()	BYTE COREMESSAGE
Function			
BYTE COREME	SSAGE::GetDes	criptor()	
Purpose			
returns the descrip	ptor for this corer	message	
Parameters			
N/A			
Returns			
descriptor			
Group F	Responsible	EMailAddress	
	Gerv	schalk@wadsworth	h.org
<u> </u>			

ModuleName	FunctionName)	ClassName
UCoreMessage.cpp	GetLength()		COREMESSAGE
Function			
unsigned short COR	REMESSAGE::G	etLength()	
Purpose			
Returns the length of	of this coremessag	ge	
		th of the actual data,	
i.e., it EXcludes the	length of the hea	der when it is being	
transmitted. The len	gth of a coremess	sage cannot exceed 65	5kB
Parameters			
N/A			
Returns			
length of the core m	essage		
	C		
Group Re	sponsible	EMailAddress	
Albany Ge	rv	schalk@wadsworth.	org
		<u> </u>	

ModuleName	FunctionName		ClassName
UCoreMessage.cpp	GetSuppDescriptor()	BYTE COREMESSAGE
Function			
BYTE COREMESS	AGE::GetSuppDescr	iptor()	
Purpose			
returns the suppleme	ntal descriptor for the	is coremessage	
Parameters			_
N/A			
Returns			
supplemental descri	ptor		
Group Res	sponsible EM	ailAddress	
Albany Ger		alk@wadsworth.	n.org

	FunctionName	ClassName
UCoreMessage.cpp	ParseMessage()	COREMESSAGE
Function		·
int COREMESSA	GE::ParseMessage()	
int COREMESSAC	GE::ParseMessage()	
Purpose		
further processes the parameter, or statu	s, espective object (status,)	n the type of the coremessage, it processes the state,
Parameters N/A		
Returns		
the type of the core	message (as defined in U DNE, if the type is not re	<u> </u>

schalk@wadsworth.org

Albany

Gerv

ModuleName	FunctionName	ClassName
UCoreMessage.cpp	ReceiveBufBytes(TCustomWinSock	COREMESSAGE
Function		

int COREMESSAGE::ReceiveBufBytes(TCustomWinSocket *Socket, char *buf, int length)

Purpose

this procedure reads EXACTLY length bytes from the specified (non-blocking) socket connection, i.e., it does not return before

Parameters

Socket - the non-blocking socket

buf - the buffer that has been allocated to hold the data

length - the number of byes to read from the socket connection

Returns

length ... on success

0 ... on error

Group	Responsible	EMailAddress
Albany	Gerv	schalk@wadsworth.org

ModuleName	FunctionName	ClassName
UCoreMessage.cpp	ReceiveBufBytes(TWinSocketStrea	COREMESSAGE
Function		

int COREMESSAGE::ReceiveBufBytes(TWinSocketStream *stream, char *buf, int length)

Purpose

this procedure reads EXACTLY length bytes from the specified Socket stream that has been created for a (blocking) socket connection, i.e., it does not return before

Parameters

stream - the stream to the blocking socket buf - the buffer that has been allocated to hold the data length - the number of byes to read from the socket connection

Returns

length ... on success 0 ... on error

0 ... on error

Group	Responsible	EMailAddress
Albany	Gerv	schalk@wadsworth.org

ModuleNam	е	FunctionNa	ame	ClassName	
UCoreMessa	ge.cpp	ReceiveCor	eMessage(TCustomWin	COREMESSAGE	
Function					
int COREM	ESSAGI	E::ReceiveCo	oreMessage(TCustomW	finSocket *Socket)	
Purpose					
			ocking Socket connection		
			sage and sets the values		
	otor, the	supplementa	al descriptor and the len	gth	
accordingly	~~~ ~~ () 4	a familian mu	4		
use Parseivie	ssage() 1	o further pro	ocess the coremessage		
Parameters Socket - the	non-bloc	king socket			
	non-bloc	king socket			
	non-bloc	king socket			
Socket - the					
Socket - the					
Socket - the					
Socket - the					
Socket - the					
Socket - the					
Socket - the	CORE_1		EMailAddress		

ModuleName	FunctionName	ClassName	
UCoreMessage.cpp	ReceiveCoreMessage(TWinSocketS	COREMESSAGE	
Function			
int COREMESSAG	int COREMESSAGE::ReceiveCoreMessage(TWinSocketStream *stream)		

Purpose

given a handle to a Socket stream that has been created for a given Socket connection,

receives the content of a coremessage and sets the values of the descriptor, the supplemental descriptor and the length accordingly

use ParseMessage() to further process the coremessage

Parameters

stream - an active stream

Returns

ERRCORE NOERR success

ERRCORE_RECEIVEBUFBYTES on failure

Group	Responsible	EMailAddress
Albany	Gerv	schalk@wadsworth.org

ı	ModuleName	FunctionName	ClassName
	UCoreMessage.cpp	SendBufBytes(TCustomWinSocket	COREMESSAGE
ı	Function		

int COREMESSAGE::SendBufBytes(TCustomWinSocket *Socket, char *buf, int length)

Purpose

this procedure sends EXACTLY length bytes to the specified (non-blocking) socket connection, i.e., it does not return before

Parameters

Socket - the non-blocking socket

buf - the buffer that has been allocated to hold the data

length - the number of byes to read from the socket connection

Returns

<= 0 ... on error

>0 ... on success

Group	Responsible	EMailAddress
Albany	Gerv	schalk@wadsworth.org

ModuleName	FunctionName	ClassName
UCoreMessage.cpp	SendCoreMessage(TCustomWinSoc	COREMESSAGE
Function		

int COREMESSAGE::SendCoreMessage(TCustomWinSocket *Socket)

Purpose

given a handle to an open non-blocking Socket connection, sends a coremessage, including the appropriate header values for the descriptor, the supplemental descriptor and the length have to be specified accordingly

Parameters

Socket - the non-blocking socket

Returns

ERRCORE NOERR on success

ERRCORE_SENDBUFBYTES, if there was an error sending bytes out ERRCORE SOCKETNOTOPEN, if the socket connection was not open

Group Responsible EMailAddress
Albany Gerv schalk@wadsworth.org

ModuleName	FunctionName	ClassName
UCoreMessage.cpp	SendCoreMessage(TWinSocketStre	COREMESSAGE
Function		

int COREMESSAGE::SendCoreMessage(TWinSocketStream *stream)

Purpose

given a handle to an open Socket stream, sends a coremessage, including the appropriate header values for the descriptor, the supplemental descriptor and the length have to be specified accordingly

Parameters

stream - the stream that has been created for a blocking socket connection

Returns

ERRCORE_NOERR on success ERRCORE_WRITE on failure

Group	Responsible	EMailAddress
Albany	Gerv	schalk@wadsworth.org

ModuleName	FunctionName		ClassName		
UCoreMessage.cp	p SetDescriptor(B	YTE newdescriptor)	COREMESSAGE		
Function					
void COREMESS	AGE::SetDescripto	or(BYTE newdescript	otor)		
Purpose					
sets the descriptor	for this coremessag	ge			
Parameters					
newdescriptor - the	e new descriptor				
Returns					
N/A					
		I= · · · ·			
	esponsible	EMailAddress			
Albany	erv	schalk@wadsworth.	n.org		

	FunctionNar	me	ClassName	
UCoreMessage	e.cpp SetLength(un	signed short newlength (COREMESSAGE	
unction				
void COREMI	ESSAGE::SetLength	(unsigned short newleng	th)	
Purpose				
This length spe i.e., it EXclude		ngth of the actual data, eader when it is being		
Parameters newlength - le	ngth of the core mes	sage		
Returns				
N/A				
Group	Responsible	EMailAddress		
<u> </u>	•			

Purpose sets the supplemental descriptor for this coremessage Parameters newsuppdescriptor - the new supplemental descriptor	ModuleName	FunctionName		ClassName
Purpose sets the supplemental descriptor for this coremessage Parameters newsuppdescriptor - the new supplemental descriptor Returns N/A Group Responsible EMailAddress	UCoreMessage.cpj	p SetSuppDescript	tor(BYTE newsupp	COREMESSAGE
Purpose sets the supplemental descriptor for this coremessage Parameters newsuppdescriptor - the new supplemental descriptor Returns N/A Group Responsible EMailAddress	Function	·		
Parameters newsuppdescriptor - the new supplemental descriptor Returns N/A Group Responsible EMailAddress	void COREMESS.	AGE::SetSuppDesc	eriptor(BYTE newsu	ippdescriptor)
Parameters newsuppdescriptor - the new supplemental descriptor Returns N/A Group Responsible EMailAddress	Purpose			
newsuppdescriptor - the new supplemental descriptor Returns N/A Group Responsible EMailAddress	sets the supplemen	tal descriptor for th	is coremessage	
N/A Group Responsible EMailAddress	Parameters newsuppdescriptor	- the new supplement	ental descriptor	
N/A Group Responsible EMailAddress				
Group Responsible EMailAddress	Returns			
· · · · · · · · · · · · · · · · · · ·	N/A			
Albany Gerv schalk@wadsworth.org	Group R	esponsible	EMailAddress	
		erv	schalk@wadsworth.	.org

ModuleName	FunctionNa	me	ClassName
UParam.cpp	SetValue		PARAM
Function			
void PARAM::	SetValue(char *src	, int idx)	
Purpose			_
sets the idx'th v	value of the parame	ter	
Parameters			
src - char point			
	he value (0GetNu	imvaiues()-1)	
Returns			
N/A			
Group	Responsible	EMailAddress	
Albany	Gerv	schalk@wadsworth	n.org
<i>y</i>		2	·- <i>U</i>

ModuleName	FunctionNa	me	ClassName
UParameter.cpp	~PARAM()		PARAM
Function			
PARAM::~PARAM	1()		
Purpose			
The destructor for the			
it deletes the list of	values that this	parameter holds	
Parameters			
N/A			
Returns			
N/A			
Group Re	sponsible	EMailAddress	
Albany Ge		schalk@wadsworth	th org
Thoung Ge	1 V	Schark@wausworth	11.016

ModuleName	FunctionNa	me	ClassName
UParameter.cpj	~PARAMLIS	ST()	PARAMLIST
Function	·		
PARAMLIST::	~PARAMLIST()		
Durmaga			
Purpose	C 41 DADAMI IC	NT -1-14 It 1-1-4	1
		ST object. It deletes also the list object it	
ali FARAIVI OU	jects in the list and	also the list object it	tsen
Danamatana			
Parameters			
N/A			
Returns			
N/A			
IN/A			
Group	Responsible	EMailAddress	
Albany	Gerv	schalk@wadswo	orth.org
<u> </u>			

ModuleNam	e FunctionNar	me ClassNai	ne
UParameter.c	pp AddParamete	r2List(char *paramstri PARAMI	LIST
Function			
void PARAM	ILIST::AddParameter	2List(char *paramstring, int para	mlen)
Purpose			
adds a new	parameter to the list o	of parameters	
if a paramet	er with the same name	e already exists,	
it updates th	e currently stored par	rameter with the provided values	
D 4 -			
Parameters	ASCII string as defi	inad in project description	
		ined in project description,	
paramstring -		ined in project description,	
paramstring -		ined in project description,	
paramstring -		ined in project description,	
paramstring - defining this		ined in project description,	
paramstring - defining this particular the particul		ined in project description,	
paramstring -		ined in project description,	
paramstring - defining this particular the particul		ined in project description,	
paramstring - defining this particular the particul		ined in project description,	
paramstring - defining this particular the particul		ined in project description,	
paramstring - defining this particular the particul		ined in project description,	
paramstring - defining this particular the particul		ined in project description,	
paramstring - defining this particular the particul		ined in project description,	

ModuleName	FunctionNa	me	ClassName
UParameter.cpp	ClearParamL	List()	PARAMLIST
Function	·		
void PARAML	IST::ClearParamLi	ist()	
Purpose			
	all entries in the pa	aramatar list	
clears and frees	an entries in the pa	arameter list	
Parameters			
N/A			
Returns			
N/A			
Group	Responsible	EMailAddress	;
Albany	Gerv	schalk@wadsw	vorth.org

ModuleName	FunctionName	FunctionName ClassName		
UParameter.cpp	CloneParameter	2List(PARAM *par	PARAMLIST	
Function				
void PARAMLIST:	::CloneParameter2	2List(PARAM *para	m)	
Purpose				
adds a new paramet				
		st() is that it actually		
physically copies th		ne specified param		
object can then be f				
		es for a parameter in	the list,	
if one with the same	e name already exi	sts		
Parameters				
param - pointer to a	n existing PARA	M object		
Returns				
N/A				
Group Re	sponsible	EMailAddress		
Albany Ge	rv	schalk@wadsworth.	.org	

ModuleName	FunctionName	ClassName	
UParameter.cpp	ConstructParameterLine()	PARAM	
Function			
int PARAM::Constr	uctParameterLine()		
Purpose			
Construct a paramet in the PARAM object	er line, based upon the current ve	alues	
Parameters			
N/A			
Returns			
ERRPARAM_NOE	RR, or		
	ONSISTENTNUMVAL, if the n		
object is inconsistent	t with the actually present number	er of values	

EMailAddress

schalk@wadsworth.org

Group

Albany

Responsible

Gerv

ModuleName	FunctionNar		ClassName		
UParameter.cpp	DeleteParam(char *name)	PARAMLIST		
Function	unction				
void PARAMLI	ST::DeleteParam(c	char *name)			
Purpose					
	eter of a given name		neters		
	memory for this par				
it does not do an	ything, if the parar	neter does not exist			
D					
Parameters name - name of	.1 .				
	the parameter				
Returns					
N/A					
Group	Responsible	EMailAddress			
Albany	Gerv	schalk@wadsw	orth.org		
		2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2	- · · · · · ·		

ModuleName	FunctionName	ClassName
UParameter.cpp	get_argument(int ptr, char *buf, cha	int PARAM
Function		

int PARAM::get argument(int ptr, char *buf, char *line, int maxlen)

Purpose

parses the parameter line that is being sent in the core communication, or as stored in any BCI2000 .prm file it returns the next token that is being delimited by either a '' or '='

Parameters

ptr - index into the line of where to start buf - destination buffer for the token

line - the whole line

maxlen - maximum length of the line

Returns

the index into the line where the returned token ends

Group	Responsible	EMailAddress
Albany	Gerv	schalk@wadsworth.org

Modul	eName	FunctionName		Classi	Name
UParar	meter.cpp	GetListPtr()		TList	*PARAM
Functi	on				
TList	*PARAM::G	etListPtr()			
Purpos	se				
Returns	s a pointer to	this parameter's l	ist of values		
Param	eters				
N/A					
Return	ıs				
pointer	to the list				
Group	o Res	sponsible	EMailAddress		
Alban			schalk@wadsworth.	org	

ModuleName	FunctionNa	me	ClassName
UParameter.cpj	GetName()		char *PARAM
Function			
char *PARAM	::GetName()		
Purpose			
Returns a point	er to this parameter	r's name	
Note: paramete	r names are case in	sensitive	
Parameters			
N/A			
Returns			
char pointer to	the name		
Group	Responsible	EMailAddress	
Albany	Gerv	schalk@wadswor	th.org

ModuleName	FunctionNa	me	ClassNam	10
UParameter.cpp	GetNumPara	meters()	PARAML	IST
Function	·		·	
int PARAMLIS	T::GetNumParamo	eters()		
Purpose				
Returns the nur	mber of parameters	s in the list		
Parameters				
N/A				
Returns				
the number of p	arameters			
Group	Responsible	EMailAddres	s	
Albany	Gerv	schalk@wads		
<u> </u>	<u> </u>			J

ModuleName	FunctionNa	me	ClassName
UParameter.cp	p GetNumValu	ies()	PARAM
Function			
int PARAM::G	etNumValues()		
Purpose			
Returns the nur	nber of values in th	is parameter	
for most types	of parameters, this	will be one	
Parameters			
N/A			
Returns			
number of valu	es in this parameter		
Group	Responsible	EMailAddress	
Albany	Gerv	schalk@wadsworth.	org
J	<u> </u>		

ModuleName	FunctionNa	ne ClassNa	me		
UParameter.cpp	GetParamLin	e() char *PA	RAM		
Function					
char *PARAM::	GetParamLine()				
Purpose					
Returns a param	eter line in ASCII	format			
Tis parameter lii values in the PA		pased upon the current			
Parameters					
N/A					
Returns					
a pointer to the p	parameter line				
	Deeneneible	EMailAddress			
Group	Responsible	=man/taarooo			

ModuleName	FunctionNa	me	ClassNaı	me
UParameter.cp	op GetParamPtı	(char *name)	PARAM	*PARAMLIST
Function				
PARAM *PA	ARAMLIST::GetPar	ramPtr(char *name)		
Purpose				
given a parame	eter name, returns th	ne pointer to a PARAN	1 object	
J 1	,	1	3	
Parameters				
	f the parameter			
	i the parameter			
Returns				
ointer to a PA	ARAM object or			
		with this name exists is	n the list	
	·			
Group	Responsible	EMailAddress		
Albany	Gerv	schalk@wadswort	h.org	

ModuleName	FunctionName)	ClassName
UParameter.cpp	GetParamPtr(in	nt idx)	PARAM *PARAMLIST
Function			
PARAM *PARAM	1LIST::GetParaml	Ptr(int idx)	
Purpose			
given an index (00	GetListCount()-1),	, returns the pointer to	o a PARAM object
Parameters			
idx - index of the p	arameter		
Returns			
pointer to a PARA			
NULL, if the speci			
Group Re	esponsible	EMailAddress	
Albany Go	erv	schalk@Wadsworth	h.org

ModuleName	FunctionNa	me	ClassName
UParameter.cp	p GetSection()		char *PARAM
Function			
char *PARAM	::GetSection()		
Purpose			
Returns a poin	ter to this parameter	's section name	
Parameters			
N/A			
Returns			
char pointer to	the section name		
Group	Responsible	EMailAddress	
Albany	Gerv	schalk@wadsworth	org
ribally	Gerv	Scharka waasworth	015

ModuleName	FunctionNa	me	ClassName
UParameter.cpp	GetType()		char *PARAM
Function			
char *PARAM:	:GetType()		
Purpose			
	er to this parameter	r's tyne	
Returns a point	or to this parameter	is type	
_			
Parameters			
N/A			
Returns			
char pointer to t	he type		
Group	Responsible	EMailAddress	
Albany	Gerv	schalk@wadswor	rth.org
	1		

UParameter.cpp	0 177 1 0		
	GetValue()	char *	PARAM
Function			
char *PARAM:	:GetValue()		
Purpose			
	rs, except the *list p	lue of this parameter parameter types, only	
Parameters N/A			
Returns			
char pointer to t	the value		
Group	Responsible	EMailAddress	
Albany	Gerv	schalk@wadsworth.org	

ModuleName	FunctionName	ClassNam	е
UParameter.cpp	GetValue(int idx	char *PAR	AM
Function			
char *PARAM::0	GetValue(int idx)		
Purpose			
	to the i'th value of the		
	, except the *list par	ameter types, only	
have one value ar	nyways)		
Parameters			
idx index of th	e value		
Returns			
char pointer to th	e value		
		nter to the first value	
	1		
Group	Responsible	EMailAddress	

schalk@wadsworth.org

Albany

Gerv

ModuleName	FunctionName)	ClassName
UParameter.cpp	MoveParameter	2List(PARAM *new	PARAMLIST
Function			
void PARAMLIST:	:MoveParameter2	2List(PARAM *newp	param)
Purpose			
adds a new paramet	er to the list of pa	arameters	
this function assume			
_	,	n the destructor of the	
3 /		CloneParameter2List()) is
that it does not actual			
This function will no		er to the list, if one	
with the same name Parameters	already exists		
param - pointer to a	n ovietina DADA	M. alainat	
Returns			
N/A			
Group Re	sponsible	EMailAddress	
Albany Ge	rv	schalk@wadsworth.	.org
i noung	• •	oonance was worth.	

	e	FunctionNam	ne	ClassName
UParameter.	срр	PARAM		PARAM
Function				
				char *value, char *my_defaultvalue, char
*my_lowrang	ge, char	*my_highrange	e, char *my_comment)	
Purpose				
Constructs a	nd initia	lizes a paramet	er object	
Parameters				
self-explana	-			
Returns				
Returns N/A				
	Res	ponsible	EMailAddress	

ModuleName	FunctionNa	me	ClassName
UParameter.cpp	PARAM()		PARAM
Function	1		
PARAM::PARA	AM()		
Durance			
Purpose	C 4 DADAM	1: .	
The constructor	for the PARAM	object	
Parameters			
N/A			
Returns			
N/A			
Group	Responsible	EMailAddress	
Albany	Gerv	schalk@wadswor	th.org
	- -		· · · · · · ·

UParameter.cpp PARAM(char *paramstring) PARAM Function PARAM::PARAM(char *paramstring) Purpose Constructs and initializes a parameter object, based upon a parameter string, as defined in the project outline Parameters char *paramstring Returns N/A	ModuleName	FunctionName)	ClassName
Purpose Constructs and initializes a parameter object, based upon a parameter string, as defined in the project outline Parameters char *paramstring Returns N/A	UParameter.cpp	PARAM(char *	'paramstring)	PARAM
Purpose Constructs and initializes a parameter object, based upon a parameter string, as defined in the project outline Parameters char *paramstring Returns N/A	Function			
Constructs and initializes a parameter object, based upon a parameter string, as defined in the project outline Parameters char *paramstring Returns N/A	PARAM::PARAN	1(char *paramstrin	g)	
a parameter string, as defined in the project outline Parameters char *paramstring Returns N/A	Purpose			
char *paramstring Returns N/A				
Returns N/A				
N/A	cnar "paramstring	5		
	Returns			
	N/A			
Group Responsible EMailAddress	Group R	esponsible	EMailAddress	
Albany Gerv schalk@wadsworth.org			schalk@wadsworth.	org

ModuleName	FunctionNa	me	ClassName
UParameter.cpp	PARAMLIST	Γ()	PARAMLIST
Function			
PARAMLIST::P	ARAMLIST()		
Purpose			
the constructor fo	or the PARAMLIS	ST object	
Parameters			
N/A			
Returns			
N/A			
Group	Responsible	EMailAddress	
	Gerv	schalk@wadsworth.	1 org
ı mouny	GCI V	Scharkw wads worth.	1.015

ModuleName	FunctionName	ClassName
UParameter.cpp	ParseParameter(char *line, int lengt	PARAM
Function		

int PARAM::ParseParameter(char *line, int length)

Purpose

This routine is called by coremessage->ParseMessage()

it parses the provided ASCII parameter line and initializes

the PARAM object accordingly, i.e., it fills in values, name, section name, type, etc.

Parameters

line - pointer to the ASCII parameter line length - length of this parameter line

Returns

 ${\sf ERRPARAM_INVALIDPARAM}$ if the parameter line is invalid, or ${\sf ERRPARAM_NOERR}$

Group	Responsible	EMailAddress
Albany	Gerv	schalk@wadsworth.org

ModuleName		FunctionName ClassName		ssName	
UPara	ameter.cpp	SetListPtr(TList	t *temp_list)	void	PARAM
Funct	tion				
void	PARAM::Set	ListPtr(TList *te	mp_list)		
Purpo	ose				
Sets tl	his parameter's	list of values			
Paran	neters				
pointe	er to the list				
Retur	ns				
N/A					
Grou	ın Por	enonciblo.	EMailAddress		
Grou Albai		sponsible	schalk@wadsworth	ora	
Aivai	ily Gel	V	schaik@wausWolti	1.018	

ModuleName	FunctionNar	ne ClassNan	ne
UParameter.cj	pp SetName(cha	r *src) PARAM	
Function			
void PARAM	::SetName(char *src)		
Purpose			
	of the parameter		
Note: paramet	er names are case ins	sensitive	
Parameters char pointer to			
F			
Returns			
N/A			
Group	Responsible	EMailAddress	
	_		

ModuleName	e FunctionNa	me	ClassName	
UParameter.c	epp SetSection(cl	nar *src)	PARAM	
Function				
void PARAM	I::SetSection(char *sr	rc)		
Purpose				
sets the section	on name of the param	eter		
Parameters				
chai pointei t	o the section name			
Returns				
N/A				
Group	Responsible	EMailAddre	ss	
Albany	Gerv	schalk@wad		

ModuleName	FunctionNa	me	ClassNam	10
UParameter.cpp	SetType(char	*src)	PARAM	
Function				
void PARAM::S	SetType(char *src)			
Purpose				
sets the type of t	the parameter			
Parameters				
char pointer to t	he type name			
Returns				
N/A				
Group	Responsible	EMailAddress]
	Gerv			
Albany	Gerv	schalk@wadsv	vorui.org	

ModuleName	FunctionNa	me	ClassName
UState.cpp	~STATE()	S	TATE
Function			
STATE::~STA	ATE()		
D			
Purpose	2 4 07 47 1		
the destructor i	for the STATE obje	ct	
Parameters			
N/A			
Returns			
N/A			
Group	Responsible	EMailAddress	
Albany	Gerv	schalk@wadsworth.o	rg
<i>j</i>	1		<u> </u>

ModuleName	FunctionNar	me	ClassName
UState.cpp	~STATELIS	Γ()	STATELIST
Function			
STATELIST::~S	TATELIST()		
Purpose			
The destructor for object itself	r the STATELIST	Γ object. It deletes both	n all STATE objects in the list and also the list
Parameters			
N/A			
Returns			
N/A			
Group I	Responsible	EMailAddress	
	Gerv	schalk@wadsworth.	1.org

	e FunctionNa	me	ClassName
UState.cpp	AddState2Lis	st(char *statestring)	STATELIST
Function			
void STATE	LIST::AddState2List(char *statestring)	
Purpose			
if a state with	ate to the list of states the same name alrea currently stored state		ues
Parameters	1.00x		
		ined in project descrip	non,
defining this i	new state		
Returns	new state		
Č	new state		
Returns	Responsible	EMailAddress	

ModuleName	FunctionName		ClassName				
UState.cpp	State.cpp AddState2List(STATE *state) STATELIST						
Function							
void STATELIS	T::AddState2List(STA	ATE *state)					
Purpose							
Adds a new state	to the list of states						
	ies the STATE object	(the specified STAT	ГЕ				
	e freed elsewhere)						
	e same name already ex						
it updates the cur	rrently stored state with	h the provided value	es				
Parameters							
	an existing STATE of	niect					
state - pointer to	an existing STATE of) cci					
Determe							
Returns							
N/A							
Group	Responsible E	EMailAddress					
_	Gerv	schalk@wadsworth.	org				

ModuleName	FunctionNar	me Cl	lassName
UState.cpp	ClearStateLis	st()	ΓATELIST
Function			
void STATELIST	Γ::ClearStateList(
Purpose			_
clears and frees a	all entries in the st	tate list	
Parameters			
N/A			
Returns			
N/A			
	Responsible	EMailAddress	
Albany	Gerv	schalk@wadsworth.or	g

ModuleName	FunctionNar	me	ClassNan	ne
UState.cpp	ConstructStat	teLine()	STATE	
Function				
int STATE::C	onstructStateLine()			
Purpose				
Construct a st	ate line, based upon	the current values	in the STATE of	object
Parameters				
N/A				
Returns				
always ERRP	ARAM_NOERR			
Group	Posnonsible	EMailAddress		7
Group	Responsible			
Albany	Gerv	schalk@wadsw	orin.org	

ModuleNam	e FunctionNa	me	ClassName
UState.cpp	DeleteState(c	char *name)	STATELIST
Function			
void STATE	LIST::DeleteState(ch	ar *name)	
Purpose			
deletes a state	e of a given name in t	the list of states	
	ne memory for this pa		
it does not do	anything, if the state	does not exist	
Parameters			
name - name	of the state		
name - name	or the state		
Returns			
N/A			
Group	Responsible	EMailAddress	S

ModuleName	FunctionName	ClassName
UState.cpp	get_argument(int ptr, char *buf, cha	STATE
Function		

int STATE::get_argument(int ptr, char *buf, char *line, int maxlen)

Purpose

parses the state line that is being sent in the core communication

it returns the next token that is being delimited by either a ' ' or '='

Parameters

ptr - index into the line of where to start

buf - destination buffer for the token

line - the whole line maxlen - maximum length of the line

Returns

the index into the line where the returned token ends

Group	Responsible	EMailAddress
Albany	Gerv	schalk@wadsworth.org

ModuleName	FunctionNa	me	ClassName
UState.cpp	GetBitLocati	on()	STATE
Function			
int STATE::G	etBitLocation()		
Purpose			
Returns this s	tate's bit location in	the state vector	
Parameters			
N/A			
Returns			
this state's bit	location		
Group	Responsible	EMailAddress	
Albany	Gerv	schalk@wadsworth	1.org
2	1-2-		<u>U</u>

ModuleName	FunctionNa	me	ClassName	
UState.cpp	GetByteLoca	tion()	STATE	
Function				
int STATE::Ge	etByteLocation()			
Purpose				
Returns this sta	ate's byte location in	the state vector		
Parameters				
N/A				
Returns				
this state's byt	e location			
Group	Responsible	EMailAddress		
	Gerv		orth org	
Albany	UCI V	schalk@wadswo	orun.org	

Function	GetLength() th()		STATE
Function int STATE::GetLeng	th()		
int STATE::GetLeng	th()		
Purpose			
Returns this state's le	ngth		
Parameters			
N/A			
Returns			
this state's length			
Croun	noncible	EMailAddress	
	ponsible		200
Albany Ger	V	schalk@wadsworth	.org

ModuleName	FunctionNa	me	ClassName
UState.cpp	GetListPtr()		TList *STATELIST
Function			
TList *STATE	ELIST::GetListPtr()		
Purpose			
Returns a point	ter to this state's list	of values	
Parameters			
N/A			
Returns			
pointer to the li	st		
Group	Responsible	EMailAddress	
Albany	Gerv	schalk@wadsworth.	org
. Houriy	3017	Sommer was worth.	<u> </u>

ModuleName	FunctionNa	me C	ClassName
UState.cpp	GetName()	cl	har *STATE
Function		·	
char *STATE	::GetName()		
Purpose			
	me of this state		
Parameters			
N/A			
Returns			
a pointer to th	e state's name		
Group	Responsible	EMailAddress	
Albany	Gerv	schalk@wadsworth.or	arg
Tibally	GGIV	Scharka waas worth.or	15

ModuleName	FunctionNa	me	ClassName
UState.cpp	GetNumState	es()	STATELIST
Function			
int STATELIS	ST::GetNumStates()		
Purpose			
Returns the nu	umber of states in the	e list	
Parameters			
N/A			
Returns			
the number of	States		
Group	Responsible	EMailAddress	
Albany	Gerv	schalk@wadswor	th.org

ModuleName	FunctionNa	me Class	sName
UState.cpp	GetStateLine	() char	*STATE
Function			
char *STATE	::GetStateLine()		
Purpose			
Returns a state	e line in ASCII forma	at	
This state line	is constructed, based	d upon the current values in t	the STATE object
Parameters			
N/A			
Returns			
a pointer to th	e state line		
Group	Responsible	EMailAddress	
Cidap			

ModuleName	FunctionNa	me	ClassName
UState.cpp	GetStateListI	Ptr()	STATELIST *STATEVECTOR
Function			
STATELIST *	STATEVECTOR::	GetStateListPtr()	
Purpose			
returns a ptr to	the list of states des	scribing the state vect	tor
Parameters			
N/A			
Returns			
pointer to the li	ist of states		
Group	Responsible	EMailAddress	
Albany	Gerv	schalk@wadswor	rth org
Hoarry	July	schark@wauswoi	ui.org

ModuleName	FunctionNa	me	ClassName
UState.cpp	GetStatePtr(c	char *name)	STATE *STATELIST
Function			
STATE *STA	TELIST::GetStateP	tr(char *name)	
Purpose			
-	name returns the no	ointer to a STATE ob	niect
Parameters name - name o	f the state		
	T the state		
Returns			
pointer to a S'	TATE object or NU	LL, if no state with the	this name exists in the list
Group	Responsible	EMailAddress	
Albany	Gerv	schalk@wadswor	orth.org

ModuleName	FunctionName	ClassName
UState.cpp	GetStatePtr(int idx)	STATE *STATELIST
Function	·	
STATE *STATE	CLIST::GetStatePtr(int idx)	
Purpose		
given an index (0	GetListCount()-1), return	s the pointer to a STATE object
Parameters		
idx - index of the	state	
Returns		
	TE object or NULL if th	e specified index is out of range
pointer to a 517	The object of Trobb, if the	specified fidex is out of funge
Group	Responsible EMai	Address

UState.cpp	0 (0) (17.1 (
	GetState value(char *statename)	unsigned short STATEVECTOR
Function			
ansigned short ST	ΓATEVECTOR::G	etStateValue(char *s	statename)
Purpose			
eturns a state's v	alue from the state	vector	
Parameters			
statename - the na	ame or a state		
Returns			
the value of the st O on error (e.g., s			
Group I	Responsible	EMailAddress	
	Gerv	schalk@wadswort	

ModuleName	FunctionName	ClassName
UState.cpp	GetStateValue(int byteloc, int bitloc,	unsigned short STATEVECTOR
Function		

unsigned short STATEVECTOR::GetStateValue(int byteloc, int bitloc, int length)

Purpose

returns a state's value, based upon the state's location and size

Parameters

byteloc ... location of the byte of the state bitloc ... location of the bit of the state

length ... length of the state

Returns

ERRSTATEVEC NOSTATE on error (e.g., location outside range) the value of the state otherwise

Group	Responsible	EMailAddress
Albany	Gerv	schalk@wadsworth.org

ModuleName	FunctionNa	me	ClassName
UState.cpp	GetStateVect	torLength()	STATEVECTOR
Function			
int STATEVE	CTOR::GetStateVe	ctorLength()	
Purpose			
returns the len	gth of the state vector	or in bytes	
Parameters			
N/A			
Returns			
the length of the	ne state vector in byt	tes	
Group	Responsible	EMailAddress	<u> </u>
Albany	Gerv	schalk@wadsw	
1 Hourry	3017	Schulkte waas w	, 0, 0, 1, 0, 1

ModuleName	FunctionNa	me	ClassNam	e
UState.cpp	GetStateVect	torPtr()	BYTE *ST	TATEVECTOR
Function				
BYTE *STAT	EVECTOR::GetSta	ateVectorPtr()		
Purpose				
returns a BYTI	E ptr to the state vec	ctor data		
Parameters				
N/A				
Returns				
a pointer to the	state vector data			
Group	Responsible	EMailAddress	<u> </u>	
Albany	Gerv	schalk@wadsw		
. Hourry	3017	Schulkte waas	. 01.111.015	l

ModuleName	FunctionNa	me Cla	ssName
UState.cpp	GetValue()	uns	igned short STATE
Function			
unsigned short	STATE::GetValue()	
Purpose			
Returns this sta	ate's value		
Parameters			
N/A			
Returns			
this state's valu	10		
uns states varu	ie		
Group	Responsible	EMailAddress	
Albany	Gerv	schalk@wadsworth.org	

ModuleName	FunctionName	•	ClassName		
UState.cpp	Initialize_StateV	Vector()	STATEVECTOR		
Function					
void STATEVEC	ΓOR::Initialize_Sta	ateVector()			
Purpose					
	assignes locations in the state vector for each state; in addition, it initializes the state vector with the values of each state in the list				
Parameters					
N/A					
Returns					
N/A					
Group R	esponsible	EMailAddress			
Albany G	erv	schalk@wadsworth.	.org		

ModuleName	FunctionName	ClassName
UState.cpp	ParseState(char *line, int statelength	th STATE
Function		

int STATE::ParseState(char *line, int statelength)

Purpose

This routine is called by coremessage->ParseMessage()

it parses the provided ASCII state line and initializes

the STATE object accordingly, i.e., it fills in value, name, byte location, bit location, etc.

Parameters

line - pointer to the ASCII state line length - length of this state line

Returns

ERRSTATE INVALIDSTATE if the state line is invalid, or ERRSTATE NOERR

Group	Responsible	EMailAddress
Albany	Gerv	schalk@wadsworth.org

ModuleName	FunctionNa	me	ClassName
UState.cpp	SetBitLocation	on(int loc)	STATE
Function			
void STATE::	SetBitLocation(int lo	oc)	
D			
Purpose	12.1		
Sets this state	s bit location in the s	state vector	
Parameters			
this state's bit	location		
Returns			
N/A			
L			
Group	Responsible	EMailAddress	s
Albany	Gerv	schalk@wadsv	worth.org

ModuleName	FunctionNa	me	ClassName
UState.cpp	SetByteLoca	tion(int loc)	STATE
Function	·		
void STATE::S	etByteLocation(int	loc)	
Purpose			
	byte location in the	a stata waatar	
	oyee rocurren in une		
Parameters			
this state's byte	location		
Returns			
N/A			
0	D	-11 -114	
Group	Responsible	EMailAddres	
Albany	Gerv	schalk@wads	worth.org

ModuleName	FunctionName	ClassName
UState.cpp	SetStateValue(char *statename, unsi	STATEVECTOR
Function		

int STATEVECTOR::SetStateValue(char *statename, unsigned short stateval)

Purpose

sets a state's value in the state vector

Parameters

byteloc ... location of the byte of the state bitloc ... location of the bit of the state

length ... length of the state value ... value of the state

Returns

ERRSTATEVEC_NOSTATE on error (e.g., location out of range) ERRSTATEVEC_NOERR otherwise

Group	Responsible	EMailAddress
Albany	Gerv	schalk@wadsworth.org

	e FunctionNa	me	ClassName
UState.cpp	SetStateVect	or(BYTE *src, int le	engt STATEVECTOR
Function	·		
void STATE	VECTOR::SetStateV	Tector(BYTE *src, in	nt length, STATELIST *src_list)
Purpose			
sets the cont	ent of a whole state v	ector	
Parameters			
	to the source state vec	etor content	
length - lengt	th of the state vector i	n bytes	
	th of the state vector inter to the list of state		te vector
	th of the state vector in the to the list of state		te vector
			te vector
			te vector
src_list - poi			te vector
			te vector
src_list - poi			te vector
src_list - poi			te vector
src_list - poi			te vector
src_list - poi			te vector
src_list - poi			te vector
src_list - poi	nter to the list of state	s describing this sta	te vector
src_list - poi			

ModuleName	FunctionName		ClassName	
UState.cpp	SetValue		STATE	
Function				
void STATE::SetVa	alue(unsigned shor	rt new_value)		
Purpose				
Sets this state's valu	ie			
Parameters				
the new value for th	us state			
Returns				
N/A				
		··- ·		
Group Re	sponsible	EMailAddress		

ModuleName	FunctionNa	me C	ClassName
UState.cpp	STATE()	S	TATE
Function			
STATE::STATE	E()		
Purpose			
the constructor for	or the STATE obj	ect	
Parameters			
N/A			
Returns			
N/A			
Group	Responsible	EMailAddress	
	Gerv	schalk@wadsworth.or	rg
	<u> </u>		

ModuleName	FunctionNa	me	ClassName
UState.cpp	STATELIST		STATELIST
Function			
STATELIST::	STATELIST()		
Purpose			
	for the STATELIS	T object	
the constituctor	101 the STATELIS	1 object	
Parameters			
N/A			
Returns			
N/A			
Group	Responsible	EMailAddress	
Albany	Gerv	schalk@wadswortl	th.org
	 		

ModuleName	FunctionNa	me CI	assName
UStatus.cpp	GetCode()	Si	TATUS
Function			
int STATUS::C	GetCode()		
_			
Purpose			
Parameters			
Returns			
Returns			
Group	Responsible	EMailAddress	
Albany	Gerv	schalk@wadsworth.or	σ
J	1		

ModuleName	FunctionNa	me CI	assName
UStatus.cpp	GetStatus()	ch	ar *STATUS
Function		1	
char *STATUS	S::GetStatus()		
Purpose			
_			
Parameters			
Returns			
Returns			
<u> </u>			
Group	Responsible	EMailAddress	
Albany	Gerv	schalk@wadsworth.org	g
			<u>~</u>

ModuleName	FunctionNam	1 e	ClassName
UStatus.cpp	ParseStatus(cl	nar *line, int length)	STATUS
Function	, `		
int STATUS::P	arseStatus(char *line	e, int length)	
Purpose			
Тагрозс			
Parameters			
Returns			
Group	Responsible	EMailAddress	
Albany	Gerv	schalk@wadswortl	h.org
-	- I		