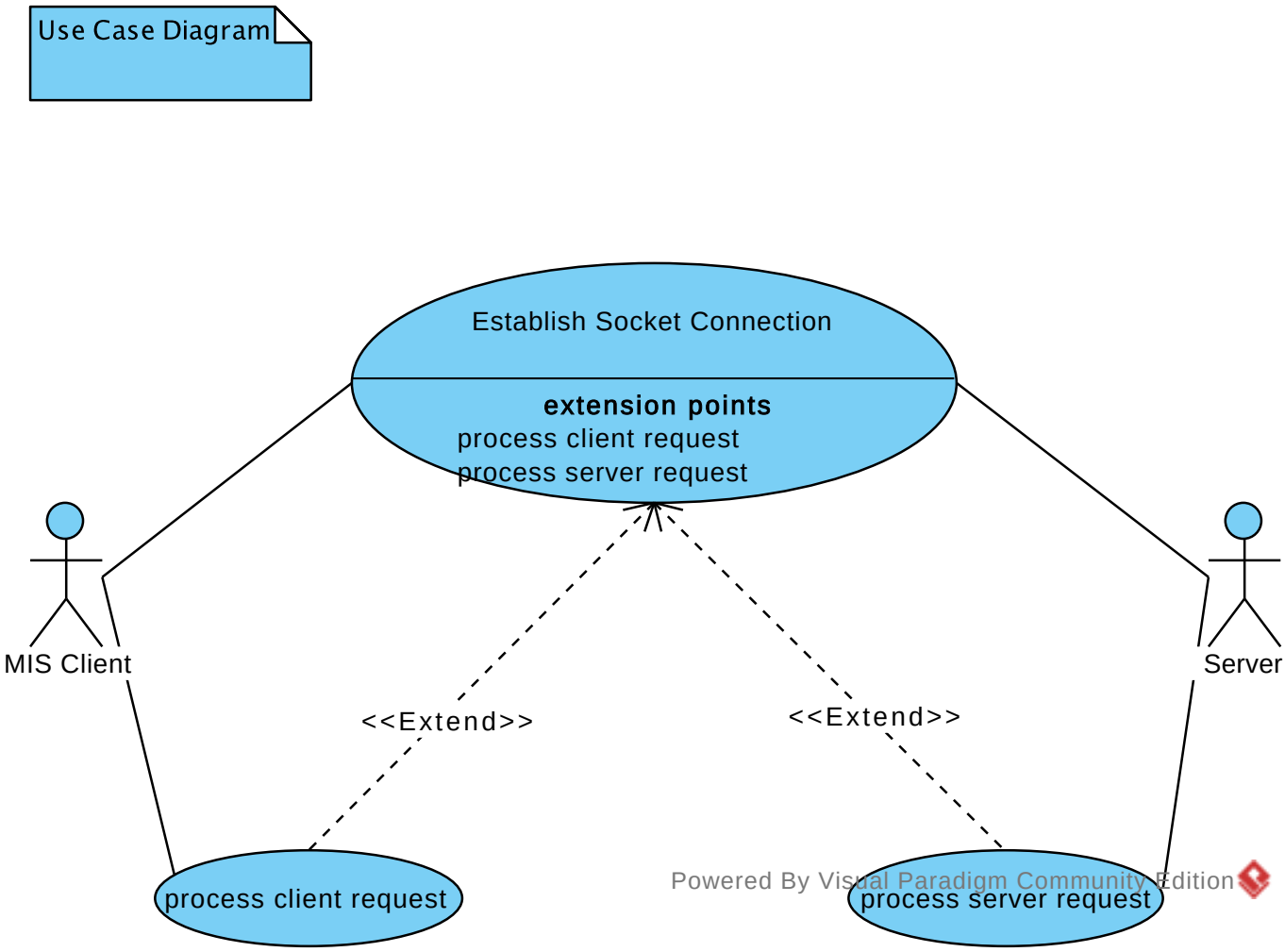
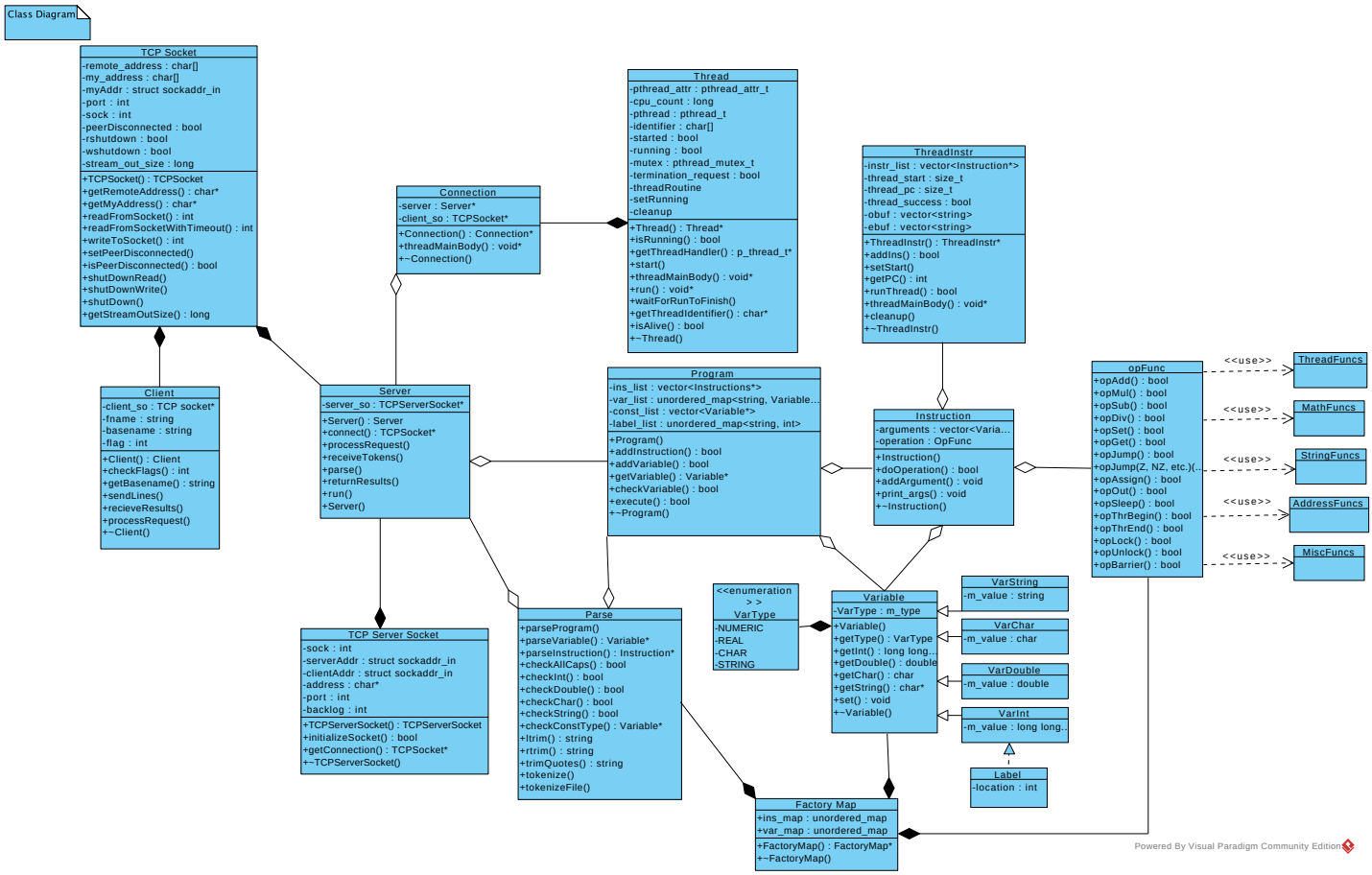
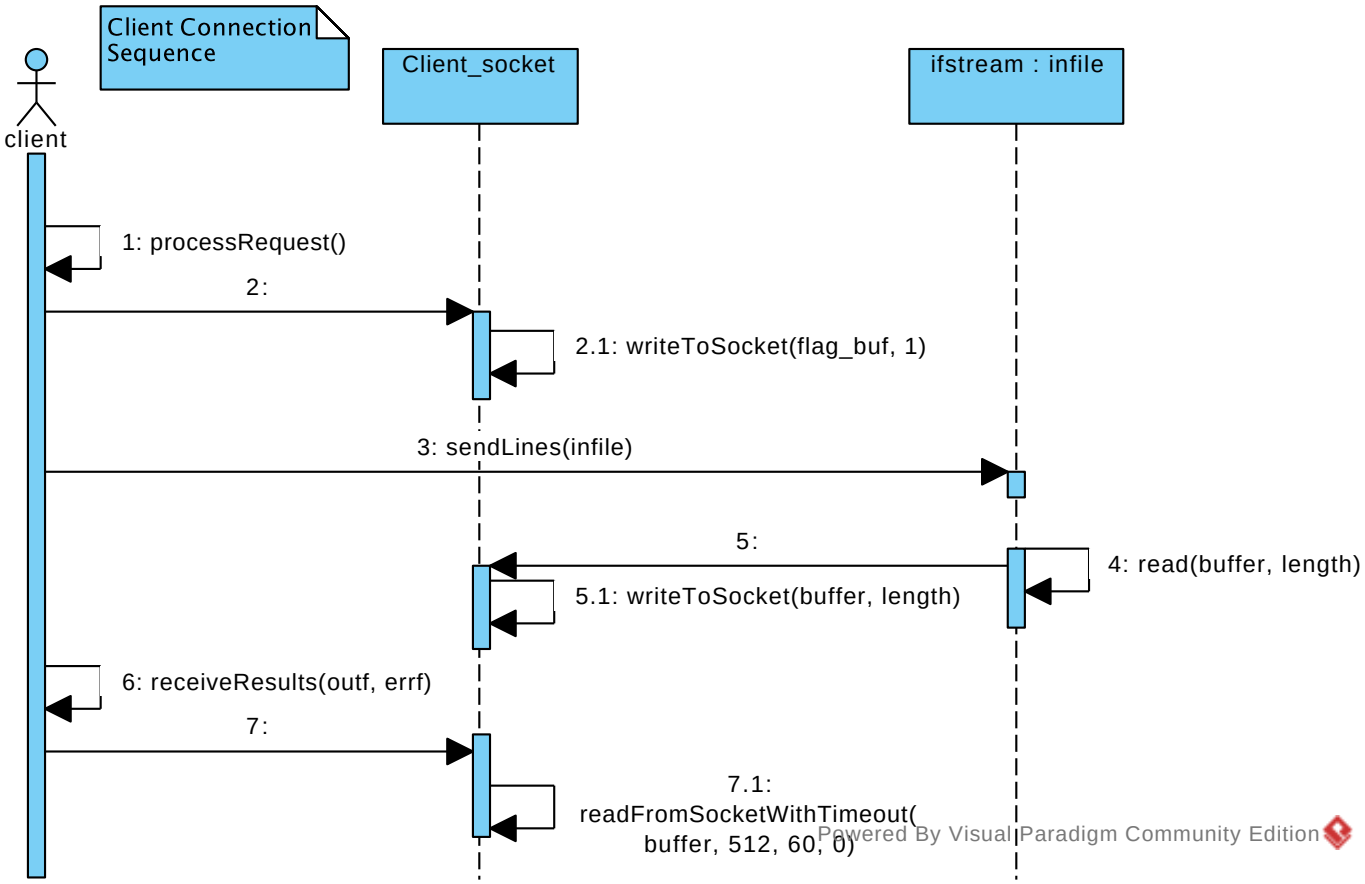
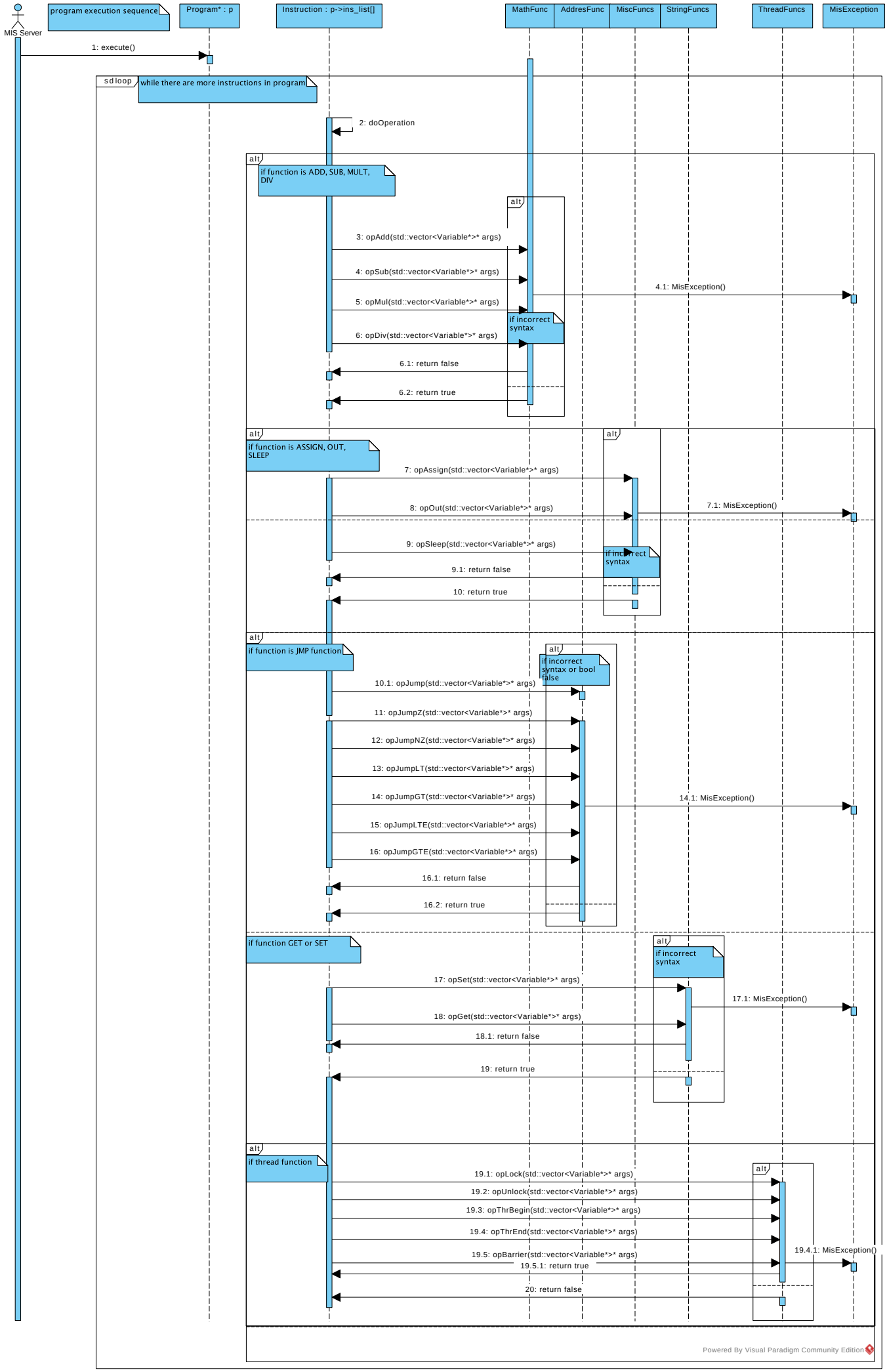
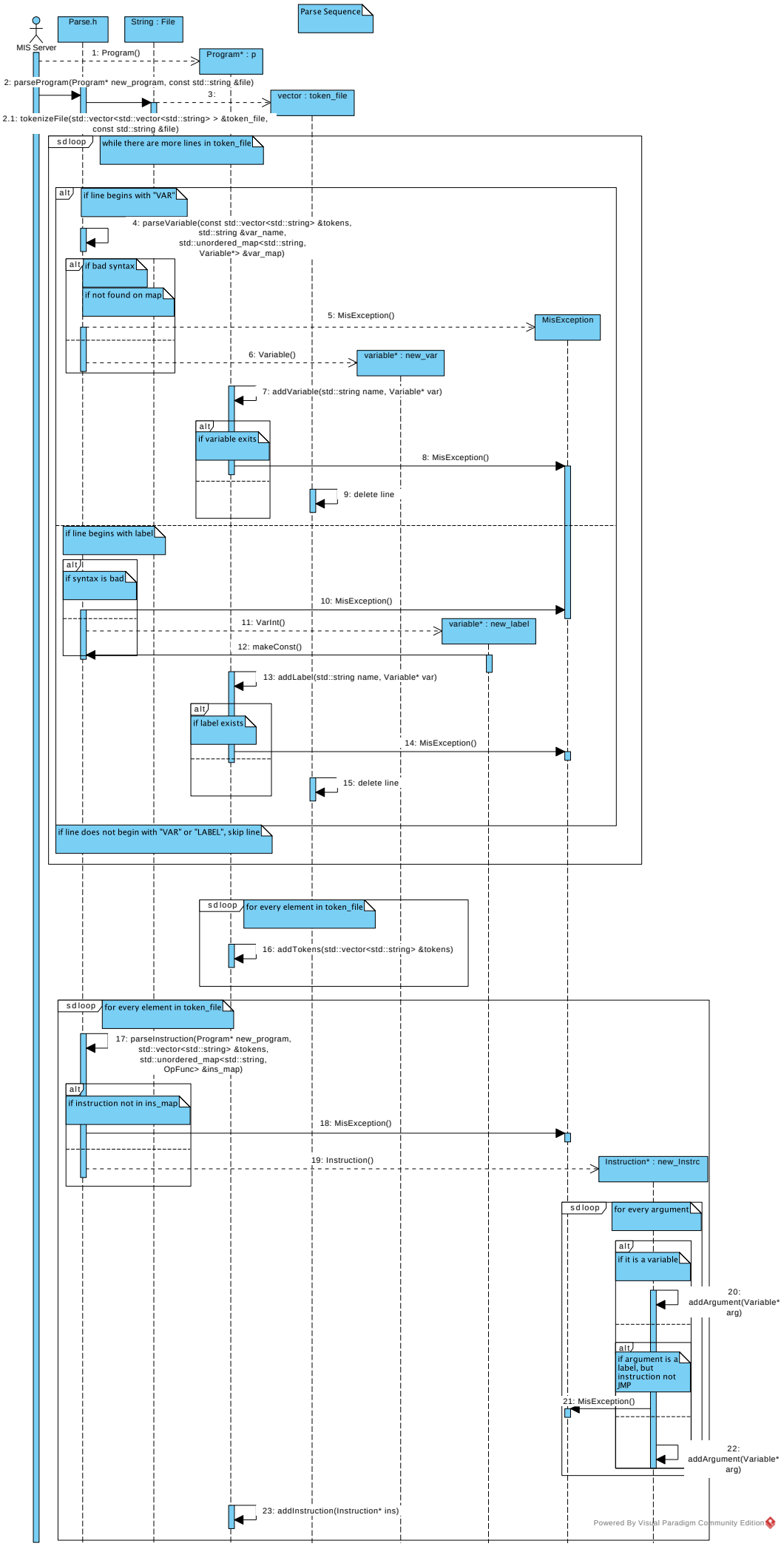
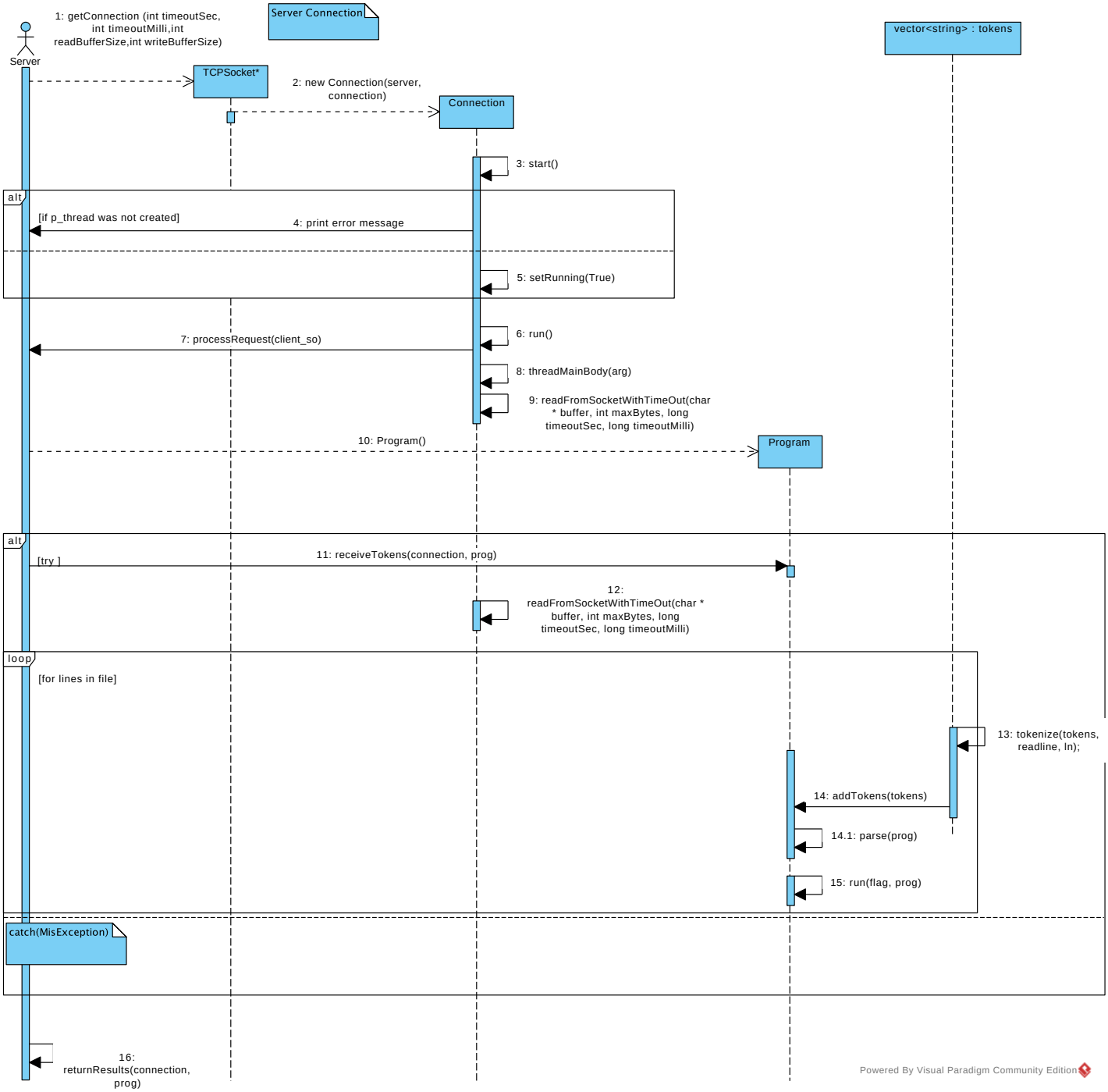

Use Case Diagram
Establish Socket Connection
extension points
process client request
process server request
MIS Client
Server
<<Extend>>
<<Extend>>
process client request
process server request
Powered By Visual Paradigm Community Edition


Class Diagram
TCP Socket
-remote_address : char[]
-my_address : char[]
-myAddr : struct sockaddr_in
-port : int
-sock : int
-peerDisconnected : bool
-rshutdown : bool
-wshutdown : bool
-stream_out_size : long
+TCPSocket() : TCPSocket
+getRemoteAddress() : char*
+getMyAddress() : char*
+readFromSocket() : int
+readFromSocketWithTimeout() : int
+writeToSocket() : int
+setPeerDisconnected()
+isPeerDisconnected() : bool
+shutDownRead()
+shutDownWrite()
+shutDown()
+getStreamOutSize() : long
Thread
-pthread_attr : pthread_attr_t
-cpu_count : long
-pthread : pthread_t
-identifier : char[]
-started : bool
-running : bool
-mutex : pthread_mutex_t
-termination_request : bool
-threadRoutine
-setRunning
-cleanup
+Thread() : Thread*
+isRunning() : bool
+getThreadHandler() : p_thread_t*
+start()
+threadMainBody() : void*
+run() : void*
+waitForRunToFinish()
+getThreadIdentifier() : char*
+isAlive() : bool
+~Thread()
Connection
-server : Server*
-client_so : TCPSocket*
+Connection() : Connection*
+threadMainBody() : void*
+~Connection()
ThreadInstr
-instr_list : vector<Instruction*>
-thread_start : size_t
-thread_pc : size_t
-thread_success : bool
-obuf : vector<string>
-ebuf : vector<string>
+ThreadInstr() : ThreadInstr*
+addIns() : bool
+setStart()
+getPC() : int
+runThread() : bool
+threadMainBody() : void*
+cleanup()
+~ThreadInstr()
Client
-client_so : TCP socket*
-fname : string
-basename : string
-flag : int
+Client() : Client
+checkFlags() : int
+getBasename() : string
+sendLines()
+recieveResults()
+processRequest()
+~Client()
Server
-server_so : TCPServerSocket*
+Server() : Server
+connect() : TCPSocket*
+processRequest()
+receiveTokens()
+parse()
+returnResults()
+run()
+Server()
Program
-ins_list : vector<Instructions*>
-var_list : unordered_map<string, Variable...
-const_list : vector<Variable*>
-label_list : unordered_map<string, int>
+Program()
+addInstruction() : bool
+addVariable() : bool
+getVariable() : Variable*
+checkVariable() : bool
+execute() : bool
+~Program()
Instruction
-arguments : vector<Varia...
-operation : OpFunc
+Instruction()
+doOperation() : bool
+addArgument() : void
+print_args() : void
+~Instruction()
opFunc
+opAdd() : bool
+opMul() : bool
+opSub() : bool
+opDiv() : bool
+opSet() : bool
+opGet() : bool
+opJump() : bool
+opJump(Z, NZ, etc.)(...
+opAssign() : bool
+opOut() : bool
+opSleep() : bool
+opThrBegin() : bool
+opThrEnd() : bool
+opLock() : bool
+opUnlock() : bool
+opBarrier() : bool
<<use>>
ThreadFuncs
<<use>>
MathFuncs
<<use>>
StringFuncs
<<use>>
AddressFuncs
<<use>>
MiscFuncs
TCP Server Socket
-sock : int
-serverAddr : struct sockaddr_in
-clientAddr : struct sockaddr_in
-address : char*
-port : int
-backlog : int
+TCPServerSocket() : TCPServerSocket
+initializeSocket() : bool
+getConnection() : TCPSocket*
+~TCPServerSocket()
Parse
+parseProgram()
+parseVariable() : Variable*
+parseInstruction() : Instruction*
+checkAllCaps() : bool
+checkInt() : bool
+checkDouble() : bool
+checkChar() : bool
+checkString() : bool
+checkConstType() : Variable*
+ltrim() : string
+rtrim() : string
+trimQuotes() : string
+tokenize()
+tokenizeFile()
<<enumeration>>
VarType
-NUMERIC
-REAL
-CHAR
-STRING
Variable
-VarType : m_type
+Variable()
+getType() : VarType
+getInt() : long long...
+getDouble() : double
+getChar() : char
+getString() : char*
+set() : void
+~Variable()
VarString
-m_value : string
VarChar
-m_value : char
VarDouble
-m_value : double
VarInt
-m_value : long long...
Label
-location : int
Factory Map
+ins_map : unordered_map
+var_map : unordered_map
+FactoryMap() : FactoryMap*
+~FactoryMap()
Powered By Visual Paradigm Community Edition


Client Connection Sequence
client
Client_socket
ifstream : infile
1: processRequest()
2:
2.1: writeToSocket(flag_buf, 1)
3: sendLines(infile)
5:
4: read(buffer, length)
5.1: writeToSocket(buffer, length)
6: receiveResults(outf, errf)
7:
7.1:
readFromSocketWithTimeout(
buffer, 512, 60, 0)
Powered By Visual Paradigm Community Edition


program execution sequence
MIS Server
Program* : p
Instruction : p->ins_list[]
MathFunc
AddresFunc
MiscFuncs
StringFuncs
ThreadFuncs
MisException
1: execute()
sd loop
while there are more instructions in program
2: doOperation
alt
if function is ADD, SUB, MULT, DIV
3: opAdd(std::vector<Variable*>* args)
4: opSub(std::vector<Variable*>* args)
5: opMul(std::vector<Variable*>* args)
6: opDiv(std::vector<Variable*>* args)
4.1: MisException()
if incorrect syntax
6.1: return false
6.2: return true
if function is ASSIGN, OUT, SLEEP
7: opAssign(std::vector<Variable*>* args)
8: opOut(std::vector<Variable*>* args)
7.1: MisException()
9: opSleep(std::vector<Variable*>* args)
9.1: return false
10: return true
if function is JMP function
if incorrect syntax or bool false
10.1: opJump(std::vector<Variable*>* args)
11: opJumpZ(std::vector<Variable*>* args)
12: opJumpNZ(std::vector<Variable*>* args)
13: opJumpLT(std::vector<Variable*>* args)
14: opJumpGT(std::vector<Variable*>* args)
14.1: MisException()
15: opJumpLTE(std::vector<Variable*>* args)
16: opJumpGTE(std::vector<Variable*>* args)
16.1: return false
16.2: return true
if function GET or SET
if incorrect syntax
17: opSet(std::vector<Variable*>* args)
17.1: MisException()
18: opGet(std::vector<Variable*>* args)
18.1: return false
19: return true
if thread function
19.1: opLock(std::vector<Variable*>* args)
19.2: opUnlock(std::vector<Variable*>* args)
19.3: opThrBegin(std::vector<Variable*>* args)
19.4: opThrEnd(std::vector<Variable*>* args)
19.5: opBarrier(std::vector<Variable*>* args)
19.4.1: MisException()
19.5.1: return true
20: return false
Powered By Visual Paradigm Community Edition


Parse Sequence
MIS Server
Parse.h
String : File
1: Program()
Program* : p
2: parseProgram(Program* new_program, const std::string &file)
3:
vector : token_file
2.1: tokenizeFile(std::vector<std::vector<std::string> > &token_file, const std::string &file)
sd loop
while there are more lines in token_file
alt
if line begins with "VAR"
4: parseVariable(const std::vector<std::string> &tokens, std::string &var_name, std::unordered_map<std::string, Variable*> &var_map)
alt
if bad syntax
if not found on map
5: MisException()
MisException
6: Variable()
variable* : new_var
7: addVariable(std::string name, Variable* var)
alt
if variable exits
8: MisException()
9: delete line
if line begins with label
alt
if syntax is bad
10: MisException()
11: VarInt()
variable* : new_label
12: makeConst()
13: addLabel(std::string name, Variable* var)
alt
if label exists
14: MisException()
15: delete line
if line does not begin with "VAR" or "LABEL", skip line
sd loop
for every element in token_file
16: addTokens(std::vector<std::string> &tokens)
sd loop
for every element in token_file
17: parseInstruction(Program* new_program, std::vector<std::string> &tokens, std::unordered_map<std::string, OpFunc> &ins_map)
alt
if instruction not in ins_map
18: MisException()
19: Instruction()
Instruction* : new_Instrc
sd loop
for every argument
alt
if it is a variable
20: addArgument(Variable* arg)
alt
if argument is a label, but instruction not JMP
21: MisException()
22: addArgument(Variable* arg)
23: addInstruction(Instruction* ins)
Powered By Visual Paradigm Community Edition


Server
1: getConnection (int timeoutSec, int timeoutMilli,int readBufferSize,int writeBufferSize)
Server Connection
TCPSocket*
2: new Connection(server, connection)
Connection
vector<string> : tokens
3: start()
alt
[if p_thread was not created]
4: print error message
5: setRunning(True)
7: processRequest(client_so)
6: run()
8: threadMainBody(arg)
9: readFromSocketWithTimeOut(char * buffer, int maxBytes, long timeoutSec, long timeoutMilli)
10: Program()
Program
alt
[try ]
11: receiveTokens(connection, prog)
12: readFromSocketWithTimeOut(char * buffer, int maxBytes, long timeoutSec, long timeoutMilli)
loop
[for lines in file]
13: tokenize(tokens, readline, In);
14: addTokens(tokens)
14.1: parse(prog)
15: run(flag, prog)
catch(MisException)
16: returnResults(connection, prog)
Powered By Visual Paradigm Community Edition