MIS Class Diagram

Phase 2

Client Side CLIENT TCPConnector Class TCPConnector +TCPStream* connect(int port, const char* server) -int resolveHost(const char* host, struct in_addr* addr) **TCPAcceptor** Class TCPAcceptor +m_listening: bool +m_lsd: int +m port: int +m_address: string +TCPAcceptor(int port, const char* address = "") + ~TCPAcceptor(); +int start() +TCPStream* accept() -TCPAcceptor() **TCPStream** Class TCPStream +buffer: char* +m_sd: int +m peerPort: int +m_peerIP_string +TCPAcceptor(): friend class +TCPConnector(): friend class + ~TCPStream + ssize_t send(const char * buffer, size_t len) + ssize_t recieve(char* buffer, size_t len) + string getPeerIP() + int getPeerPort() **Thread** Class Thread LineQueue -pthread t m tid -int m_running Class LineQueue -int m_detatched +Thread() linequeue() + Virtual ~Thread()

+ ~linequeue()

void add(T item)

T Remove(): return item

int size(): return size

+int start()

+int join()

+int detach()

+pthread_t self() +virtual void* run() =0

SERVER MACHINE TCPStream Parser Class TCPStream Class Parser +buffer: char* -parsed: vector<string> +m sd: int - cmdMap: +m peerPort: int vector<vector<string>> +m_peerIP_string +Parser() +TCPAcceptor(): friend class +vector<string> parseFile(string file) +TCPConnector(): friend class + ~TCPStream +vector<vector<string>> parseInstructions() + ssize_t send(const char * buffer, size_t len) + ssize_t recieve(char* buffer, size_t len) + string getPeerIP() + int getPeerPort() **TCPAcceptor** Class TCPAcceptor +m_listening: bool +m_lsd: int +m port: int +m address: string +TCPAcceptor(int port, const char* address = "") + ~TCPAcceptor(); +int start() +TCPStream* accept() -TCPAcceptor() Linequeue Class LineQueue linequeue() + ~linequeue() void add(T item) T Remove(): return item int size(): return size

Server Side

Jump Add Class Jump Class Add Data Class Data + Add(); Jump(); - std::map<std::string,VAR *> varMap; +virtual ~Add(); virtual ~Jump(); std::map<std::string,int> labelMap; + virtual void virtual void execute(Data - int current; execute(Data *d, * d,vector<string> line); +vector<string> line); virtual Instruction * + Data() + virtual Instruction * clone(); int getCurrent(): return current; clone(); void setCurrent(int i) +void addVar(string name, VAR * v) +VAR * getVar(string name): return varMap[name]; void addLabel(string label, int i) Char +int getLabel(string label): return labelMap[label]; Class Char -value: char +Char(); **JumpGT** + Char(std::string n, char v); + ~Char(); Class JumpGT + void initialize(vector<string> line); + VAR * clone(vector<string> line); + char getValue() const; JumpGT(); + void setValue(char c); virtual ~JumpGT(); + friend std::ostream& operator<<(std::ostream& virtual void execute(Data * d,vector<string> line); os, + const Char& var); virtual Instruction * clone(); Real SET STR CHAR Class Real Class SET_STR_CHAR double value; SET STR CHAR(); Real(); virtual ~SET STR CHAR(): Real(std::string n, double v); virtual void execute(Data * d,vector<string> line); virtual ~Real(); virtual Instruction * clone(); virtual void initialize (vector<string> line); VAR * clone (vector<string> line); GET STR CHAR void setValue(double v); double getValue() const; Class GET_STR_CHAR Real operator*(const Real& other); Real operator/(const Real& other); Real operator-(const Real& other); Real operator+(const Real& other); GET_STR_CHAR(); Real& operator=(const Real& other); virtual ~GET_STR_CHAR(); Real& operator=(const int& n); virtual void execute(Data *d, vector<string> line); Real& operator+=(const Real& other): virtual Instruction * clone(); Real& operator+=(const int& i); Real& operator+=(const double& d); Real& operator+=(const Numeric& num); Div Real& operator*=(const Real& other); Class Div Real& operator*=(const int& i); Real& operator*=(const double& d); Real& operator*=(const Numeric& num); friend std::ostream& operator<<(std::ostream& os, const + Div(); Real& var): + virtual ~Div(); +virtual void execute(Data *d, vector<string> line);

+ virtual Instruction * clone();

JumpGT

INSTR

Class Jump@

JumpGTE() virtual ~JumpG virtual void execute(Data * d, virtual Instruction

Jumpl

Class Jur

JumpLT virtual ~Jur virtual void execute(Data * virtual Instruction

JumpL

Class Jun

JumpLT virtual ~Jum virtual void execute(Data * virtual Instruction

Jumpl

Class Jur

JumpN. virtual ~Jun virtual void execute(Data * virtual Instruction

Sle

Class :

Sleei virtual ~ virtual void execute(Data

virtual Instruc

ISTRUCTION **VAR** Class Instruction Class VAR String Name Instruction(); String Type virtual ~Instruction(); virtual void execute(Data * d,vector<string> line)=0; VAR(); virtual string getType() const; virtual ~VAR(); virtual Instruction* clone()=0; string getType() const; string getName() const; npGTE JumpZ virtual void initialize (vector<string> line)=0; virtual VAR* clone (vector<string> line)=0; Class JumpZ JumpGTE String Class String JumpZ(); npGTE(); virtual ~JumpZ(); ~JumpGTE(); int length; Data * d,vector<string> line); virtual void execute(Data * d,vector<string> line); std::string value; virtual Instruction * clone(); static const size_t MAX = 256; ruction * clone(); String(); String(std::string n, std::string v, int l); Label JumpLT virtual ~String(); Class Label ialize (vector<string> line); ass JumpLT VAR * clone (vector<string> line); void setValue(std::string v); char getChar(int i); Label(); int getLength(); JumpLT(); virtual ~Label(); string getValue(); ual ~JumpLT(); virtual void execute(Data *d,vector<string> line); e(Data * d,vector<string> line); void setAt(char c,int i); virtual Instruction * clone(); char &operator[](int i); nstruction * clone(); Numeric Mul umpLTE Numeric Class Mul ass JumpLTE int value Mul(); JumpLTE(); virtual ~Mul(); al ~JumpLTE(); Numeric(); virtual void execute(Data *d, vector<string> line); e(Data * d,vector<string> line); Numeric(string n, int v); virtual Instruction * clone(); nstruction * clone(); virtual ~Numeric(); virtual void initialize (vector<string> line); Out lumpNZ virtual VAR * clone(vector<string> line); Class Out void setValue(int v); ass JumpNZ int getValue() const; Numeric operator*(const Numeric& other); Numeric operator/(const Numeric& other); Numeric operator-(const Numeric& other); Out(); JumpNZ(); virtual ~Out(); Numeric operator+(const Numeric& other); ual ~JumpNZ(); virtual void execute(Data *d, vector<string> line); Numeric& operator=(const Numeric& other); e(Data * d,vector<string> line); Numeric& operator=(const int& n); virtual Instruction * clone(); nstruction * clone(); Numeric* operator+=(const Numeric* other); Numeric* operator+=(const int& i); Sub Sleep Numeric* operator+=(const double& d): Class Skeep Class Sub Numeric* operator*=(const Numeric* other); Numeric* operator*=(const int& i); Numeric* operator*=(const double& d);

Sub();

virtual ~Sub();

virtual void execute(Data *d, vector<std::string>

virtual Instruction * clone();

line);

Sleep();

line);

virtual ~Sleep();

Instruction * clone();

ute(Data *d, vector<std::string>

Assign

Class Assign

+Assign();

+virtual ~Assign();

+virtual void

execute(Data *d,

+vector<string>

line);

+ virtual Instruction

* clone();

friend std::ostream& operator<<(std::ostream& os, const

Numeric& var);