ddt - RCI

Generated by Doxygen 1.8.9.1

Wed Mar 18 2015 15:01:02

Contents

Chapter 1

Data Structure Index

1.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

Chapter 3

Data Structure Documentation

3.1 args_parse_options Struct Reference

Estrutura com os argumentos para a função parse_options.

`#include <options.h>`

Data Fields

- int * argc

  Ponteiro para o local em memória do contador de argumentos.
- char *** argv

  Ponteiro para o local em memória dos argumentos.
- struct startup_data * startup_data

  Ponteiro para uma das estruturas transversais ao projecto.

3.1.1 Field Documentation

3.1.1.1 int* args_parse_options::argc

3.1.1.2 char*** args_parse_options::argv

3.1.1.3 struct startup_data* args_parse_options::startup_data

The documentation for this struct was generated from the following file:

- include/options.h

3.2 peer_data Struct Reference

Estrutura com a informação referente a um par.

`#include <common.h>`

Data Fields

- int id

  Número do nó pelo qual o par é responsável.

- char node [256]

  Endereço do par.
- char service [16]

  Porto do par.
- int socket

  Socket TCP com o par.

3.2.1 Field Documentation

3.2.1.1 int peer_data::id

3.2.1.2 char peer_data::node[256]

3.2.1.3 char peer_data::service[16]

3.2.1.4 int peer_data::socket

The documentation for this struct was generated from the following file:

- include/common.h

3.3 startup_data Struct Reference

Estrutura com os dados a ser recolhidos pela linha de comandos.

`#include <common.h>`

Data Fields

- char ringport [16]

  Porto do servidor TCP.
- int family

  Família da socket.
- struct sockaddr * destination

  Estrutura de destino para as mensagens UDP.
- socklen_t dest_size

  Tamanho da estrutura de destino para as mensagens UDP.

3.3.1 Field Documentation

3.3.1.1 socklen_t startup_data::dest_size

3.3.1.2 struct sockaddr* startup_data::destination

3.3.1.3 int startup_data::family

3.3.1.4 char startup_data::ringport[16]

The documentation for this struct was generated from the following file:

- include/common.h

3.4 transversal_data Struct Reference

Estrutura com as referências necessárias aos vários módulos do projecto.

`#include <common.h>`

Data Fields

- int u
  Socket UDP.
- int t
  Socket TCP [accept].
- struct peer_data peer_pred
  Estrutura do predecessor.
- struct peer_data peer_succ
  Estrutura do sucessor.
- int ring
  Anel a ser utilizado.
- int id
  Identificador actual do nó.
- char reg
  Registered Node.
- struct startup_data startup_data
  Parâmetros da linha de comandos.
- char ext_addr [40]
  IP externo do servidor.
- int serv_arranq
- int socket_with_new_node

3.4.1 Field Documentation

3.4.1.1 char transversal_data::ext_addr[40]

3.4.1.2 int transversal_data::id

3.4.1.3 struct peer_data transversal_data::peer_pred

3.4.1.4 struct peer_data transversal_data::peer_succ

3.4.1.5 char transversal_data::reg

3.4.1.6 int transversal_data::ring

3.4.1.7 int transversal_data::serv_arranq

3.4.1.8 int transversal_data::socket_with_new_node

3.4.1.9 struct startup_data transversal_data::startup_data

3.4.1.10 int transversal_data::t

3.4.1.11 int transversal_data::u

The documentation for this struct was generated from the following file:

- include/common.h

Chapter 4

File Documentation

4.1 include/common.h File Reference

Cabeçalho com declarações das estruturas de dados transversais aos vários módulos do projecto.

```
#include <sys/socket.h>
#include <netinet/in.h>
#include <netinet/ip.h>
```

Data Structures

- struct startup_data

  Estrutura com os dados a ser recolhidos pela linha de comandos.
- struct peer_data

  Estrutura com a informação referente a um par.
- struct transversal_data

  Estrutura com as referências necessárias aos vários módulos do projecto.

4.2 include/interface.h File Reference

Cabeçalhos e protótipos associados às funções relacionadas com a interface com o utilizador.

Functions

- void print_ui ()

  Função de impressão dos comandos disponíveis.
- int interface (struct transversal_data *transversal_data)

  Função de análise dos comandos inseridos.

4.2.1 Function Documentation

4.2.1.1 int interface (struct transversal_data * *transversal_data*)

Após detecção de dados para ser lidos via stdin, esta função processa esses dados de maneira a executar o comando inserido pelo utilizador.

4.2.1.2 void print_ui ( )

Esta função mostra os comandos disponíveis para o utilizador. Os comandos sem parâmetros são hardcoded, enquanto que os comandos com parâmetros estão descritos em funções individuais de maneira a facilitar as mensagens de erro de formato.

4.3 include/net_common.h File Reference

Cabeçalhos e protótipos associados às funções relacionadas com a rede mas não associadas a nenhum protocolo em específico.

Functions

- void getsockaddr (char *node, char *service, int *family, socklen_t *size, struct sockaddr **sockaddress, int protocol)

  Função de leitura de endereços e portos e criação de estruturas de destino.

4.3.1 Function Documentation

4.3.1.1 void getsockaddr (char * *node*, char * *service*, int * *family*, socklen_t * *size*, struct sockaddr ** *sockaddress*, int *protocol*)

Esta função recebe um endereço e um porto, bem como a família (que é tanto uma entrada como uma saída), o tamanho da estrutura de saída e um ponteiro para a estrutura de saída. Caso esse ponteiro seja NULL, a estrutura é alocada.

A função getaddrinfo() é utilizada em vez da função gethostbyname(), por uma questão de preferência pessoal e também pelo facto da função getaddrinfo() estar especificada como thread-safe (a função gethostbyname() só é thread-safe em algumas implementações). Os detalhes da função getaddrinfo() podem ser consultados em RFC 2553.

4.4 include/net_tcp.h File Reference

Cabeçalhos e protótipos associados às funções relacionadas com o protocolo TCP.

Functions

- void createserver_tcp (struct transversal_data *transversal_data)

  Função de abertura do servidor TCP.

- int connect_tcp (char *node, char *service)

  Função de abertura de uma sessão TCP com outro nó.

4.4.1 Function Documentation

4.4.1.1 int connect_tcp (char * *node*, char * *service*)

Esta função cria uma socket e liga-a directamente ao nó, retornando então essa mesma socket.

4.4.1.2 void createserver_tcp (struct transversal_data * transversal_data)

Esta função cria a socket para recepção de mensagens via TCP. Esta função consiste de socket(), bind() e listen() exclusivamente, o accept deve ser feito a posteriori, de maneira a não bloquear a execução do programa.

4.5 include/net_udp.h File Reference

Cabeçalhos e protótipos associados às funções relacionadas com o protocolo UDP.

Functions

- void getudpdest (char *node, char *service, struct startup_data *startup_data)

  Função de determinação do destino do protocolo UDP.
- void createsocket_udp (struct transversal_data *transversal_data)

  Função de abertura da socket UDP com o servidor de arranque.

4.5.1 Function Documentation

4.5.1.1 void createsocket_udp (struct transversal_data * transversal_data)

Esta função recebe como parâmetros a estrutura de dados transversal ao programa e abre uma socket UDP tendo como destino o servidor de arranque.

4.5.1.2 void getudpdest (char * node, char * service, struct startup_data * startup_data)

Esta função recebe o endereço e porto do servidor de arranque e coloca na estrutura comum a estrutura correspondente a esse endereço.

Na estrutura comum é alocada memória para a struct sockaddr de destino.

4.6 include/options.h File Reference

Cabeçalhos e protótipos associados ao módulo de processamento das opções de linha de comandos.

```
#include "common.h"
```

Data Structures

- struct args_parse_options

  Estrutura com os argumentos para a função parse_options.

Functions

- void * parse_options (struct args_parse_options *params)

  Função responsável pela interpretação dos argumentos passados via linha de comandos.

4.6.1 Function Documentation

4.6.1.1 void* parse_options (struct args_parse_options * params)

Esta função recorre à função getopt() de maneira a obter os diversos parâmetros passados via linha de comandos. Os argumentos são passados via ponteiro para estrutura de maneira a que a função seja compatível com threads.

4.7 include/ring.h File Reference

```
#include "common.h"
```

Functions

- int join_ring (char *ring, char *num, struct transversal_data *transversal_data)

4.7.1 Function Documentation

4.7.1.1 int join_ring (char * ring, char * num, struct transversal_data * transversal_data)

4.8 include/trata_message.h File Reference

Functions

- int check_message (char **message, int num_words)
- void write_message_tcp (char *string, int socket)
- void trata_messagem (char *buffer)
- void preenche_predi_info (struct transversal_data *transversal_data, char *id, char *ip, char *porto)
- int dist (int ele, int eu)
- int verifica_se_responsavel (char *c, int eu_id, int predi_id)

4.8.1 Function Documentation

4.8.1.1 int check_message (char ** message, int num_words)

4.8.1.2 int dist (int ele, int eu)

4.8.1.3 void preenche_predi_info (struct transversal_data * transversal_data, char * id, char * ip, char * porto)

4.8.1.4 void trata_messagem (char * buffer)

4.8.1.5 int verifica_se_responsavel (char * c, int eu_id, int predi_id)

4.8.1.6 void write_message_tcp (char * string, int socket)

4.9 src/interface.c File Reference

Interface com o utilizador.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include "common.h"
#include "ring.h"
#include "trata_message.h"
```

Macros

- #define _XOPEN_SOURCE 700

Functions

- void print_join_s ()

  Função de impressão de formato do comando <join> (curto).
- void print_join_l ()

  Função de impressão de formato do comando <join> (longo).
- void print_search ()

  Função de impressão de formato do comando <search>.
- void print_ui ()

  Função de impressão dos comandos disponíveis.
- void print_error ()

  Função de impressão de erro nos comandos.
- int interface (struct transversal_data *transversal_data)

  Função de análise dos comandos inseridos.

4.9.1 Detailed Description

Este ficheiro contém as funções relacionadas com a interface com o utilizador.

4.9.2 Macro Definition Documentation

4.9.2.1 #define _XOPEN_SOURCE 700

4.9.3 Function Documentation

4.9.3.1 int interface (struct transversal_data * transversal_data)

Após detecção de dados para ser lidos via stdin, esta função processa esses dados de maneira a executar o comando inserido pelo utilizador.

4.9.3.2 void print_error ( )

Esta função mostra uma mensagem de erro, informando o utilizador que o comando que inseriu é inválido.

4.9.3.3 void print_join_l ( )

4.9.3.4 void print_join_s ( )

4.9.3.5 void print_search ( )

4.9.3.6 void print_ui ( )

Esta função mostra os comandos disponíveis para o utilizador. Os comandos sem parâmetros são hardcoded, enquanto que os comandos com parâmetros estão descritos em funções individuais de maneira a facilitar as mensagens de erro de formato.

4.10 src/main.c File Reference

Ficheiro principal do projecto.

```
#include <stdlib.h>
#include <stdio.h>
#include "common.h"
#include "options.h"
#include "net_udp.h"
#include "net_tcp.h"
#include "interface.h"
```

Functions

- int main (int argc, char **argv)

  Função principal do programa ddt.

4.10.1 Detailed Description

Este ficheiro serve como chave de abóbada para os vários módulos desenvolvidos e responsáveis pelas diversas funcionalidades do projecto.

4.10.2 Function Documentation

4.10.2.1 int main (int *argc,* char ** *argv*)

Esta função serve-se dos módulos criados nos restantes ficheiros-fonte para implementar as funcionalidades necessárias ao projecto, como especificadas no enunciado.

4.11 src/net_common.c File Reference

Ficheiro responsável pela implementação das funções relacionadas com rede mas não associadas a nenhum dos protocolos utilizados.

```
#include <stdlib.h>
#include <stdio.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netdb.h>
#include <string.h>
#include "common.h"
#include "net_common.h"
```

Macros

- #define _XOPEN_SOURCE

Functions

- void getsockaddr (char *node, char *service, int *family, socklen_t *size, struct sockaddr **sockaddress, int protocol)

  Função de leitura de endereços e portos e criação de estruturas de destino.

4.11.1 Macro Definition Documentation

4.11.1.1 #define _XOPEN_SOURCE

4.11.2 Function Documentation

4.11.2.1 void getsockaddr (char * *node*, char * *service*, int * *family*, socklen_t * *size*, struct sockaddr ** *sockaddress*, int *protocol*)

Esta função recebe um endereço e um porto, bem como a família (que é tanto uma entrada como uma saída), o tamanho da estrutura de saída e um ponteiro para a estrutura de saída. Caso esse ponteiro seja NULL, a estrutura é alocada.

A função getaddrinfo() é utilizada em vez da função gethostbyname(), por uma questão de preferência pessoal e também pelo facto da função getaddrinfo() estar especificada como thread-safe (a função gethostbyname() só é thread-safe em algumas implementações). Os detalhes da função getaddrinfo() podem ser consultados em RFC 2553.

4.12 src/net_tcp.c File Reference

Ficheiro responsável pela implementação das funções de rede associadas ao protocolo TCP.

```
#include <stdlib.h>
#include <stdio.h>
#include <sys/types.h>
#include <sys/socket.h>
#include "common.h"
#include "net_common.h"
#include "net_tcp.h"
```

Macros

- #define RCI_BACKLOG 16

Functions

- void createserver_tcp (struct transversal_data *transversal_data)

  Função de abertura do servidor TCP.
- int connect_tcp (char *node, char *service)

  Função de abertura de uma sessão TCP com outro nó.

4.12.1 Macro Definition Documentation

4.12.1.1 #define RCI_BACKLOG 16

4.12.2 Function Documentation

4.12.2.1 int connect_tcp (char * *node,* char * *service*)

Esta função cria uma socket e liga-a directamente ao nó, retornando então essa mesma socket.

4.12.2.2 void createserver_tcp (struct **transversal_data** * *transversal_data*)

Esta função cria a socket para recepção de mensagens via TCP. Esta função consiste de socket(), bind() e listen() exclusivamente, o accept deve ser feito a posteriori, de maneira a não bloquear a execução do programa.

4.13 src/net_udp.c File Reference

Ficheiro responsável pela implementação das funções de rede associadas ao protocolo UDP.

```
#include <stdlib.h>
#include <stdio.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <netdb.h>
#include <string.h>
#include <stdint.h>
#include <arpa/inet.h>
#include "common.h"
#include "net_common.h"
#include "net_udp.h"
```

Functions

- void getudpdest (char *node, char *service, struct startup_data *startup_data)

  Função de determinação do destino do protocolo UDP.
- void createsocket_udp (struct transversal_data *transversal_data)

  Função de abertura da socket UDP com o servidor de arranque.

4.13.1 Function Documentation

4.13.1.1 void createsocket_udp (struct **transversal_data** * *transversal_data*)

Esta função recebe como parâmetros a estrutura de dados transversal ao programa e abre uma socket UDP tendo como destino o servidor de arranque.

4.13.1.2 void getudpdest (char * *node*, char * *service*, struct **startup_data** * *startup_data*)

Esta função recebe o endereço e porto do servidor de arranque e coloca na estrutura comum a estrutura correspondente a esse endereço.

Na estrutura comum é alocada memória para a struct sockaddr de destino.

4.14 src/options.c File Reference

Ficheiro responsável pela interpretação dos argumentos passados via linha de comandos.

```
#include <unistd.h>
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include "common.h"
#include "options.h"
#include "net_udp.h"
```

Macros

- #define _XOPEN_SOURCE

Functions

- void usage (char *appname)

  Função que mostra a mensagem Usage.
- void * parse_options (struct args_parse_options *params)

  Função responsável pela interpretação dos argumentos passados via linha de comandos.

4.14.1 Detailed Description

As funções implementadas neste módulo correspondem à interpretação dos argumentos passados via linha de comandos, e correspondente transferência para a estrutura de dados transversal aos vários módulos do projecto.

4.14.2 Macro Definition Documentation

4.14.2.1 #define _XOPEN_SOURCE

4.14.3 Function Documentation

4.14.3.1 void* parse_options (struct **args_parse_options** * *params*)

Esta função recorre à função getopt() de maneira a obter os diversos parâmetros passados via linha de comandos. Os argumentos são passados via ponteiro para estrutura de maneira a que a função seja compatível com threads.

4.14.3.2 void usage (char * *appname*)

4.15 src/ring.c File Reference

```
#include <stdio.h>
#include <string.h>
#include <sys/types.h>
#include <sys/socket.h>
#include "ring.h"
#include "common.h"
```

Macros

- #define RCI_MSGSIZE 256

Functions

- int join_ring (char *ring, char *num, struct transversal_data *transversal_data)

4.15.1 Macro Definition Documentation

4.15.1.1 #define RCI_MSGSIZE 256

4.15.2 Function Documentation

4.15.2.1 int join_ring (char * *ring,* char * *num,* struct **transversal_data** * *transversal_data*)

4.16 src/trata_message.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "common.h"
#include "trata_message.h"
#include "math.h"
```

Functions

- int dist (int ele, int eu)
- int verifica_se_responsavel (char *c, eu_id, predi_id)
- int check_message (char **message, int num_words)
- void write_message_tcp (char *string, int socket)
- void preenche_predi_info (struct transversal_data *transversal_data, char *id, char *ip, char *porto)
- void trata_messagem (char *buffer, struct transversal_data *transversal_data)

4.16.1 Function Documentation

4.16.1.1 int check_message (char ** *message,* int *num_words*)

4.16.1.2 int dist (int *ele,* int *eu*)

4.16.1.3 void preenche_predi_info (struct transversal_data * *transversal_data,* char * *id,* char * *ip,* char * *porto*)

4.16.1.4 void trata_messagem (char * *buffer,* struct transversal_data * *transversal_data*)

4.16.1.5 int verifica_se_responsavel (char * *c,* eu_id , predi_id)

4.16.1.6 void write_message_tcp (char * *string,* int *socket*)

Index