

ddt - RCI

Generated by Doxygen 1.8.9.1

Wed Mar 18 2015 15:01:02

Contents

1	Data Structure Index	1
1.1	Data Structures	1
2	File Index	3
2.1	File List	3
3	Data Structure Documentation	5
3.1	args_parse_options Struct Reference	5
3.1.1	Field Documentation	5
3.1.1.1	argc	5
3.1.1.2	argv	5
3.1.1.3	startup_data	5
3.2	peer_data Struct Reference	5
3.2.1	Field Documentation	6
3.2.1.1	id	6
3.2.1.2	node	6
3.2.1.3	service	6
3.2.1.4	socket	6
3.3	startup_data Struct Reference	6
3.3.1	Field Documentation	6
3.3.1.1	dest_size	6
3.3.1.2	destination	6
3.3.1.3	family	6
3.3.1.4	ringport	6
3.4	transversal_data Struct Reference	7
3.4.1	Field Documentation	7
3.4.1.1	ext_addr	7
3.4.1.2	id	7
3.4.1.3	peer_pred	7
3.4.1.4	peer_succ	7
3.4.1.5	reg	7
3.4.1.6	ring	7

3.4.1.7	serv_arranq	7
3.4.1.8	socket_with_new_node	7
3.4.1.9	startup_data	7
3.4.1.10	t	7
3.4.1.11	u	7
4	File Documentation	9
4.1	include/common.h File Reference	9
4.2	include/interface.h File Reference	9
4.2.1	Function Documentation	9
4.2.1.1	interface	9
4.2.1.2	print_ui	10
4.3	include/net_common.h File Reference	10
4.3.1	Function Documentation	10
4.3.1.1	getsockaddr	10
4.4	include/net_tcp.h File Reference	10
4.4.1	Function Documentation	10
4.4.1.1	connect_tcp	10
4.4.1.2	createserver_tcp	11
4.5	include/net_udp.h File Reference	11
4.5.1	Function Documentation	11
4.5.1.1	createsocket_udp	11
4.5.1.2	getudpdest	11
4.6	include/options.h File Reference	11
4.6.1	Function Documentation	12
4.6.1.1	parse_options	12
4.7	include/ring.h File Reference	12
4.7.1	Function Documentation	12
4.7.1.1	join_ring	12
4.8	include/trata_message.h File Reference	12
4.8.1	Function Documentation	12
4.8.1.1	check_message	12
4.8.1.2	dist	12
4.8.1.3	preenche_predi_info	12
4.8.1.4	trata_mensagem	12
4.8.1.5	verifica_se_responsavel	12
4.8.1.6	write_message_tcp	12
4.9	src/interface.c File Reference	12
4.9.1	Detailed Description	13
4.9.2	Macro Definition Documentation	13

4.9.2.1	_XOPEN_SOURCE	13
4.9.3	Function Documentation	13
4.9.3.1	interface	13
4.9.3.2	print_error	13
4.9.3.3	print_join_l	14
4.9.3.4	print_join_s	14
4.9.3.5	print_search	14
4.9.3.6	print_ui	14
4.10	src/main.c File Reference	14
4.10.1	Detailed Description	14
4.10.2	Function Documentation	14
4.10.2.1	main	14
4.11	src/net_common.c File Reference	14
4.11.1	Macro Definition Documentation	15
4.11.1.1	_XOPEN_SOURCE	15
4.11.2	Function Documentation	15
4.11.2.1	getsockaddr	15
4.12	src/net_tcp.c File Reference	15
4.12.1	Macro Definition Documentation	16
4.12.1.1	RCI_BACKLOG	16
4.12.2	Function Documentation	16
4.12.2.1	connect_tcp	16
4.12.2.2	createserver_tcp	16
4.13	src/net_udp.c File Reference	16
4.13.1	Function Documentation	16
4.13.1.1	createsocket_udp	16
4.13.1.2	getudpdest	17
4.14	src/options.c File Reference	17
4.14.1	Detailed Description	17
4.14.2	Macro Definition Documentation	17
4.14.2.1	_XOPEN_SOURCE	17
4.14.3	Function Documentation	17
4.14.3.1	parse_options	17
4.14.3.2	usage	17
4.15	src/ring.c File Reference	18
4.15.1	Macro Definition Documentation	18
4.15.1.1	RCI_MSGSIZE	18
4.15.2	Function Documentation	18
4.15.2.1	join_ring	18
4.16	src/trata_message.c File Reference	18

4.16.1	Function Documentation	18
4.16.1.1	check_message	18
4.16.1.2	dist	18
4.16.1.3	preenche_predi_info	19
4.16.1.4	trata_mensagem	19
4.16.1.5	verifica_se_responsavel	19
4.16.1.6	write_message_tcp	19
Index		21

Chapter 1

Data Structure Index

1.1 Data Structures

Here are the data structures with brief descriptions:

args_parse_options	Estrutura com os argumentos para a função parse_options	5
peer_data	Estrutura com a informação referente a um par	5
startup_data	Estrutura com os dados a ser recolhidos pela linha de comandos	6
transversal_data	Estrutura com as referências necessárias aos vários módulos do projecto	7

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

include/common.h	
Cabeçalho com declarações das estruturas de dados transversais aos vários módulos do projecto	9
include/interface.h	
Cabeçalhos e protótipos associados às funções relacionadas com a interface com o utilizador	9
include/net_common.h	
Cabeçalhos e protótipos associados às funções relacionadas com a rede mas não associadas a nenhum protocolo em específico	10
include/net_tcp.h	
Cabeçalhos e protótipos associados às funções relacionadas com o protocolo TCP	10
include/net_udp.h	
Cabeçalhos e protótipos associados às funções relacionadas com o protocolo UDP	11
include/options.h	
Cabeçalhos e protótipos associados ao módulo de processamento das opções de linha de comandos	11
include/ring.h	
.	12
include/trata_message.h	
.	12
src/interface.c	
Interface com o utilizador	12
src/main.c	
Ficheiro principal do projecto	14
src/net_common.c	
Ficheiro responsável pela implementação das funções relacionadas com rede mas não associadas a nenhum dos protocolos utilizados	14
src/net_tcp.c	
Ficheiro responsável pela implementação das funções de rede associadas ao protocolo TCP .	15
src/net_udp.c	
Ficheiro responsável pela implementação das funções de rede associadas ao protocolo UDP .	16
src/options.c	
Ficheiro responsável pela interpretação dos argumentos passados via linha de comandos . .	17
src/ring.c	
.	18
src/trata_message.c	
.	18

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_mensagem](#) (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_mensagem (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_mensagem (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_mensagem (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

`_XOPEN_SOURCE`

`interface.c`, 13
`net_common.c`, 15
`options.c`, 17

`argc`

`args_parse_options`, 5

`args_parse_options`

`argc`, 5
`argv`, 5
`startup_data`, 5

`argv`

`args_parse_options`, 5

`check_message`

`trata_message.c`, 18
`trata_message.h`, 12

`connect_tcp`

`net_tcp.c`, 16
`net_tcp.h`, 10

`createserver_tcp`

`net_tcp.c`, 16
`net_tcp.h`, 10

`createsocket_udp`

`net_udp.c`, 16
`net_udp.h`, 11

`dest_size`

`startup_data`, 6

`destination`

`startup_data`, 6

`dist`

`trata_message.c`, 18
`trata_message.h`, 12

`ext_addr`

`transversal_data`, 7

`family`

`startup_data`, 6

`getsockaddr`

`net_common.c`, 15
`net_common.h`, 10

`getudpdest`

`net_udp.c`, 16
`net_udp.h`, 11

`id`

`peer_data`, 6
`transversal_data`, 7

`include/common.h`, 9

`include/interface.h`, 9

`include/net_common.h`, 10

`include/net_tcp.h`, 10

`include/net_udp.h`, 11

`include/options.h`, 11

`include/ring.h`, 12

`include/trata_message.h`, 12

`interface`

`interface.c`, 13

`interface.h`, 9

`interface.c`

`_XOPEN_SOURCE`, 13

`interface`, 13

`print_error`, 13

`print_join_l`, 13

`print_join_s`, 14

`print_search`, 14

`print_ui`, 14

`interface.h`

`interface`, 9

`print_ui`, 9

`join_ring`

`ring.c`, 18

`ring.h`, 12

`main`

`main.c`, 14

`main.c`

`main`, 14

`net_common.c`

`_XOPEN_SOURCE`, 15

`getsockaddr`, 15

`net_common.h`

`getsockaddr`, 10

`net_tcp.c`

`connect_tcp`, 16

`createserver_tcp`, 16

`RCI_BACKLOG`, 16

`net_tcp.h`

`connect_tcp`, 10

`createserver_tcp`, 10

`net_udp.c`

`createsocket_udp`, 16

`getudpdest`, 16

`net_udp.h`

`createsocket_udp`, 11

`getudpdest`, 11

- node
 - peer_data, 6
- options.c
 - _XOPEN_SOURCE, 17
 - parse_options, 17
 - usage, 17
- options.h
 - parse_options, 12
- parse_options
 - options.c, 17
 - options.h, 12
- peer_data, 5
 - id, 6
 - node, 6
 - service, 6
 - socket, 6
- peer_pred
 - transversal_data, 7
- peer_succ
 - transversal_data, 7
- preenche_predi_info
 - trata_message.c, 18
 - trata_message.h, 12
- print_error
 - interface.c, 13
- print_join_l
 - interface.c, 13
- print_join_s
 - interface.c, 14
- print_search
 - interface.c, 14
- print_ui
 - interface.c, 14
 - interface.h, 9
- RCI_BACKLOG
 - net_tcp.c, 16
- RCI_MSGSIZE
 - ring.c, 18
- reg
 - transversal_data, 7
- ring
 - transversal_data, 7
- ring.c
 - join_ring, 18
 - RCI_MSGSIZE, 18
- ring.h
 - join_ring, 12
- ringport
 - startup_data, 6
- serv_arranq
 - transversal_data, 7
- service
 - peer_data, 6
- socket
 - peer_data, 6
- socket_with_new_node
 - transversal_data, 7
- src/interface.c, 12
- src/main.c, 14
- src/net_common.c, 14
- src/net_tcp.c, 15
- src/net_udp.c, 16
- src/options.c, 17
- src/ring.c, 18
- src/trata_message.c, 18
- startup_data, 6
 - args_parse_options, 5
 - dest_size, 6
 - destination, 6
 - family, 6
 - ringport, 6
 - transversal_data, 7
- t
 - transversal_data, 7
- transversal_data, 7
 - ext_addr, 7
 - id, 7
 - peer_pred, 7
 - peer_succ, 7
 - reg, 7
 - ring, 7
 - serv_arranq, 7
 - socket_with_new_node, 7
 - startup_data, 7
 - t, 7
 - u, 7
- trata_message.c
 - check_message, 18
 - dist, 18
 - preenche_predi_info, 18
 - trata_mensagem, 19
 - verifica_se_responsavel, 19
 - write_message_tcp, 19
- trata_message.h
 - check_message, 12
 - dist, 12
 - preenche_predi_info, 12
 - trata_mensagem, 12
 - verifica_se_responsavel, 12
 - write_message_tcp, 12
- trata_mensagem
 - trata_message.c, 19
 - trata_message.h, 12
- u
 - transversal_data, 7
- usage
 - options.c, 17
- verifica_se_responsavel
 - trata_message.c, 19
 - trata_message.h, 12

write_message_tcp
 trata_message.c, [19](#)
 trata_message.h, [12](#)