NAME
DCP: ADI_alloc_cmd, ADI_alloc_attr, ADI_alloc_string, ADI_receive, ADI_acknowledge, ADI_free, ADIR_initiate_session, ADIR_alloc_cmd, ADIR_alloc_response, ADIR_alloc_ready, ADIR_alloc_message, ADIR_alloc_capinfo, ADIR_send, ADIR_free - PRODUCT_NAMEX library control

SYNOPSIS
#include <dcp.h>
#include <adi.h>
#include <ov_lib.h>

struct ADI_command *ADI_alloc_cmd(enum ADI_cmd command);

void ADI_alloc_attr(struct ADI_attrlist **list, char *name, char *value);

void ADI_alloc_string(struct ADI_stringlist **list, char *string);

struct ADI_command *ADI_receive(char *buffer, time_t timeout);

void ADI_acknowledge(struct ADI_command *cmd);

void ADI_free(struct ADI_command *cmd);

int ADIR_initiate_session(char *hostname, char *instance, char *client, int flags);

struct ADIR_command *ADIR_alloc_cmd(enum ADI_cmd command);

struct ADIR_command *ADIR_alloc_response(enum ADI_response_type type, char *errtoken, struct ADI_command *cmd);

struct ADIR_command *ADIR_alloc_ready(enum ADIR_ready_type type, char *reason);

struct ADIR_command *ADIR_alloc_message(enum ADIR_msg_severity sever, enum ADIR_msg_whom who);

struct ADIR_capinfo ADIR_alloc_capinfo(struct ADIR_command *cmd);

void ADIR_send(struct ADIR_command *cmd);

void ADIR_free(struct ADIR_command *cmd);

DESCRIPTION
A drive control program (DCP) uses abstract drive interface (ADI) routines to receive commands from the MLM server, and ADI response (ADI/R) routines to send responses and requests back to the MLM server. ADI and ADI/R routines are described below.

ADI_alloc_cmd() allocates memory for an ADI_command structure for the task specified by command, and returns a pointer to that ADI_command structure in memory.

ADI_alloc_attr() allocates memory for an attribute name and value pair, and links the attribute-value pair into the ADI_attrlist structure list.

ADI_alloc_string() allocates memory for string and links this into the ADI_stringlist structure list.
ADI_receive() parses the ADI command string in buffer, provided the command arrives before timeout expires, and returns a pointer to an ADI_command structure. Usually timeout is set to the polling interval or OV_TIMEOUT_STANDARD (120 seconds).

ADI_acked() informs the server that the DCP received the incoming ADI_command structure cmd previously returned by ADI_receive().

ADI_free() frees up memory allocated for the ADI_command structure for the cmd previously returned by ADI_receive().

ADIR_initiate_session() begins a session with the MLM server running on hostname for a specific instance of the DCP running on client, with optional flag settings.

ADIR_alloc_cmd() allocates memory for an ADIR_command structure for the task specified by command, and returns a pointer to that ADIR_command structure in memory.

ADIR_alloc_response() allocates memory for an ADIR_response structure of ADI_response_type type (accepted, unacceptable, success, error, cancelled), possibly with error indication errtoken, in reply to the ADI_command structure specified by cmd, and returns a pointer to the resulting ADIR_command structure in memory.

ADIR_alloc_ready() allocates memory for an ADIR_ready structure of ADI_ready_type type (yes, no, lost, broken), with a device-dependent reason string if not ready, and returns a pointer to the resulting ADIR_command structure in memory.

ADIR_alloc_message() allocates memory for an ADIR_message with a given severity (emergency, alert, critical, error, warning, notice, information, debug), intended for somebody who (operator, admin, all) reads the message, and returns a pointer to the resulting ADIR_command structure in memory.

Given an existing ADIR_config command structure as the cmd parameter, ADIR_alloc_capinfo() returns drive capability information inside a linked list of ADIR_capinfo structures.

ADI_free() frees up memory allocated for the ADI_command structure for the cmd previously returned by ADIR_alloc_cmd() or equivalent.

DIAGNOSTICS
In the event of failure, these routines generally return null values and set system error codes as defined in the <errno.h> include file.

SEE ALSO
umsh(1M), AAPI(3), CAPI(3), LCP(3)