CIS Droid Attack Starfighter Control Program

Category: Technical Document
Description: This is a guide to a fictional control program on a typical computer, using basic XML as its input method.
Background: This was created as a practice piece in documenting some of the more complicated programming things. This was created after studying XML and various REST API Documents, attempting to use some of the same methods and style in the Control Program document.

CIS Droid Attack Starfighter Control Program

Table of Contents

Introduction

Welcome to the Confederacy of Independent Systems' guide to utilizing the Droid Attack Starfighter (DAS) Control Program. This Control Program provides broad strategic guidance to any model of Droid Attack Starfighter in the CIS Navy. With this, the DAS can be deployed in unpredictable ways during battles with the Grand Army of the Republic, leading to a tactical advantage while minimizing lives lost during the war.

So how does this Command Program work? The DAS Control program is an Application that can open up on the command computers within the CIS Stardestroyer class vessel. After the Application opens, the user types preset commands into the command interface in a basic XML script. Any build of DAS starfighter understands these commands. As the DAS executes the command(s) it sends a Return detailing the ship's status, command(s) received, and other details relating to the command(s) given. All input information is checked by the ship's primary system before delivery to the DAS so that incomplete or potentially harmful strategic commands are not executed.

This document informs entry-level workers of the CIS how to properly command and control Droid Attack Starfighters using the DAS Command Application. This is the basics, with general information on the command application, basic commands, and everything someone will need to get started. This document is *not* the mission/station file. Instead, this has general information for ALL situations; if you are looking for mission/station-specific information, commands, and/or attributes, then reference the mission/station book on your computer — it is located in the Assignment folder.

Basics of the Control Interface

The Droid Attack Starfighter Control functions by taking preset XML codes, input by the user into the command bar, and interprets it into an action for a Droid Attack Starfighters (DAS) to follow. The DAS will return battle data to the commanding console as it executes the issued command. If necessary, more information on the DAS's status can be obtained through querying with specific preset XML code.

The issued command is referred to as the Primary Command.

The battle data returned is called the BattleLog.

Input

```
…Commanding unit c5x-123

<Bombing/>
```

Return

```
…Command Received: "Bombing"
…Selected target located x89 y78 z4
…Executing bombing run…
…Bombing run completed
Target: hit
DamageEstimate: shield systems damaged
shield system 45% functional
…Resuming "Attack" maneuvers…
```

In the example above, the Primary Command displayed in the Input is Bombing, and the Battle Log is all the information recorded in the Return.

The primary function of this API is to aid overall battle strategy, *not* to give real-time control. DASs have internal A.I. modes that take care of complex and reflexive combat maneuvers. These maneuvers allow the DAS to evade incoming projectiles or target nearby starfighters.

The API commands instruct the droid to take up positions, routes, or tasks during combat. For example, the basic Defense command would trigger a change in the DAS AI so that the DAS would target enemy fighters or bombers that are attacking larger crafts. Likewise, the basic Attack command would cause the DAS AI to attack the nearest fighters, gunships, and capital ships.

Entering an Input

To enter an Input into the console, select the Input bar in the Control Application and type out a command phrase using the correct format. The DAS control program reads preset XML text to receive user commands; each command must be as follows: <command/>.

If it is a complex command, write it in the following manner: <command attribute=" value"/>.

If there is more than one attribute, simply add a space between each attribute before closing with />.

After the Input is entered in the proper form, just hit the "enter" key. The Command App will automatically apply color coding, so it is easier to spot input errors.

Primary Command

Primary Command is the command issued by the user to the DAS; they have many different functions. Primary commands can deploy the DAS, change the DAS's combat mode, and choose strategic maneuvers.

These primary commands break into three intuitive groups

1. Function Commands
2. Mode Commands
3. Strategic Commands

Each of these groups contains commands that classify as basic and complex commands.

Function Commands

Function commands are basic request commands outside of combat, such as Deploy or Return commands. The Function Commands dictate if the DAS is on or off and can also form groups of DAS.

The Diagnostic Commands are also considered Function Commands. These commands query a DAS system and retrieve responses to what parts — if any — of the DAS are damaged or otherwise in need of repair. Diagnostic Commands can also query the ship's battle log and re-examine what happened within a battle.

Mode Commands

Mode commands change the action priority of the DAS's A.I. This includes simple commands such as Attack or Defend. Mode Commands are also used to create hybrid modes — combining different modes — as will be covered in the Advanced Commands section.

Strategic Commands

Strategic Commands are the most versatile command type, commanding the DAS to complete pre-planned complex maneuvers or simpler mid-fight strategies. Strategic commands consist of coded maneuvers, basic commands, and complex commands involving large amounts of coordinates.

BattleLog

The Battlelog is a log that starts after the initial Deploy command is issued and only ends when the DAS is destroyed or successfully executes the Return command. The Battlelog is a constant stream of information from the DAS to the command interface, detailing damage to other starfighters, damage sustained, and command details. The BattleLog's purpose is to inform the user of what damage was sustained by the DAS and what commands the fighter has and hasn't been able to complete.

Note: The BattleLog is not meant to be a comprehensive status report of the fighter's actions while in the field. A comprehensive log would clutter the data return and make the BattleLog impractical. For a full status report of the droid activities, utilize the Status *command. More details on the* Status *command can be found in the basic command section.*

Battlelog reports consist of Mission Status, DAS Status, and Damage Dealt.

Mission Status consists of the commands given and reports when these commands are considered completed.

DAS Status reports when or if parts of the DAS are hit, detailing what parts are damaged and how that will affect the DAS's performance.

Damage Det details the targets the DAS hit and the predicted system damage. This is most useful when dealing damage to Star Destroyer class vessels or other vessels above Starfighter Class.

Basic Commands

A Basic command is the most simplistic version of a command given to the DAS. These commands generally consist of short words that trigger quick responses or changes in behavior for the DAS, such as Return or Attack commands. Basic commands *do not* have attributes.

Basic Command Example
To demonstrate what a basic command will usually look like, we will use the Bombing command.

Input

```
…Commanding unit c5x-123
```

<Bombing/>

Return

```
…Command Received: "Bombing"
…Selected target located x89 y78 z4
…Executing bombing run…
…Damage received
Systems 97% functional
Critical system status: green
…Bombing run completed
Target: hit
DamageEstimate: shield systems damaged
shield system 45% functional
…Resuming "Attack" maneuvers…
```

As shown from the example, unit c5x-123 received the Bombing command and executed a successful bombing maneuver. Once a DAS, or DAS group, is selected to receive commands, its name(s) is placed at the top of the input page. Commands are displayed in the space below it to make the DAS(s) perform a task. For this basic command, all the user did was type in the command Bombing into the proper input location.

While the command itself is simple, the return is somewhat more complex.

The first task for the BattleLog is to return the received command. This demonstrates that there was no scrambling or jamming from hostile forces and that the droid has received the correct command.

With the Bombing command received, the DAS's programming selects the nearest appropriate bomb target (usually a prescanned large ship, station, or ground target). This target, and the target's coordinates, are reported back in the BattleLog before starting a bombing run and attacking the target.

In this example, the DAS took damage from a hostile force while conducting its bombing run. The DAS reports all the damage it took in a simplified manner, listing only current functionality and whether or not any critical systems were damaged.

Note: It is generally recommended that the Status *command is run after seeing that the DAS has taken damage unless there is another set of orders that need to be executed immediately after the initial command is complete. Sometimes light damage on non-critical systems can lead to more critical damage later on, especially on armor plates. The* Status *command provides more detailed damage reports and gives information on whether or not a critical system failure has become more likely.*

Once the command is complete, the BattleLog will send in confirmation of completion to the user as well as brief details on the estimated attack results. When given a basic Bombing command, the DAS will first target weapon and shield systems to increase the survivability and impact of the larger Starships on the battlefield.

After the command is completed, the DAS will inform the user what mode or previous command it is moving back to. In this case, the DAS was in attack mode before being ordered to complete a bombing run, so it is now returning to attack mode.

List of Basic Commands

The list below will contain the basic commands, as well as short descriptions to explain the function that each command serves.

1. Deploy: The first command issued in any deployment. The Deploy command instructs the DAS to initialize and enter into its default state. For most DAS, this will be flight mode. This command also starts the BattleLog.
2. Return: This is the last command to be used. Return commands the DAS to return to its assigned carrier and power off. This stops the BattleLog, though the log will remain on the console until the session is closed.
3. Attack: This is an A.I. mode command which changes the DOS's priority list. The Attack command changes the DOS's A.I. to prioritize shooting down starfighters and other attack crafts.
4. Defend: This is the second A.I. mode command, which changes the DOS's priority list. The Defend command changes the DOS's A.I. to prioritize shooting down bomber crafts and other fighters launching attacks on larger friendly attack crafts or ground targets.
5. Evade: This is the third A.I. mode command, which changes the DOS's priority list. The Evade command changes the DOS's A.I. to prioritize evading enemy projectiles and fire over dealing damage to enemy starfighters. This will cause the DOS to move behind larger crafts in some circumstances.
6. Protect: This is the fourth A.I. mode command, which changes the DOS' priority list. The project is a more specific command than the Defend command, but there is still an important distinction. While Defend alters the A.I. to prioritize shooting down ships aiming to take down other ships, Protect inputs that the DAS should also utilize its frame to block projectiles and draw away fire. It is not uncommon for the Protect command to result in the destruction of the DAS, but it is very effective at the preservation of important targets.
7. Decoy: This is the fifth A.I. mode command, which changes the DOS's priority list. The Decoy command tasks the DAS with drawing away enemy fire or distracting from other maneuvers. This generally consists of attack runs followed by evasive maneuvers that loop back into attack runs.
8. Isolate: Isolate is a command resembling an A.I. mode command, but it ends once an isolated target is destroyed. Once the command is issued, the DAS AI selects a nearby priority enemy vessel and makes maneuvers and attacks to push that enemy vessel away from its allied vessels. This is generally used to isolate and destroy less combat-inclined support vessels.
9. Bombing: The Bombing command tasks the A.I. to select a nearby bombable target and make a bombing run. This command is useful when bombing out ground troops or taking down larger enemy vessels.
10. Strategic Commands: Strategic Commands are complex commands distilled into a single command line for ease of use. These commands are long strings of numbers, letters, and characters that command the droid A.I. to utilize a specific pre-loaded command in the DAS's database. Do not worry about having to type out the Strategic Command accurately. These commands must be copied from the Mission Strategy File and pasted into the Input section.
11. Status: The Status command requests more information from the DAS and its current situation. It returns a detailed report of all of the segments of the DAS, both critical and non-critical. It also returns its location at the status check and the details of any

commands it is currently following. The Status command is the best way to fully realize the droid's condition while the DAS is deployed.

12. LastCommand: LastCommand requests that the DAS provide the full details of the previous mission in the battle log. For example, if the user wanted the DAS to repeat the battle log segment for the Bombing command (as in the example above), they would enter LastCommand.
13. RepeatLog: The RepeatLog command is used only when a mission is put on review after the DAS is returned. The RepeatLog command requests the DAS to return the BattleLog of the previous deployment. This includes the information on all commands and status checks sent to the DAS during that time.
14. Form Commands: The Form commands are simple commands specific to the DAS that utilize its ability to" walk" on the ground and fly in the air. The Form command is entered as follows: <Form=" 1"/> And it is used to change the DAS from walking mode to flying mode (Flying is form 1, walking is form 2). There are very few instances where this specific command will be utilized.

Complex Commands

Complex Commands are commands that request more specific action from the DAS than a Basic Command would provide. A complex takes the same initial Basic Command and then adds Attributes to it to control how the A.I. completes the command.

Complex Command Example

Here is a complex Attack command and its return to demonstrate what a Complex Command looks like.

Input

```
…Commanding unit c5x-123

<Attack target=“x89_y78_z24” route=“x34_y78_z24 to x98_y78_z24”
dmgprio=“shields”/>
```

Return

```
…Command Received: "Attack"
…Targeting x89 y78 z4
…Proceeding to route coordinates "x34_y78_z24"
…Starting attack run prioritizing "shields" heading to "x98_y78_z24"
…Attack successful
Target: hit
Damage: shield systems damaged, 0% functional
…Resuming "Defense" maneuvers…
```

As seen above, the Input is much more complex, and the BattleLog has more detailed information on the specifics of the command.

For the Input, there is one command (Attack) and three attributes affecting the command (target, route, and dmgprio). The Attack command commands the DAS to change to attack mode. The target attribute tells the DAS to target the ship listed at battlefield coordinates x89 y78 z24. The route attribute tells the DAS to perform its attack run while moving from x34-y78-z24 to x98-y78-z34. Finally, the dmgprio attribute tells the DAS to prioritize damaging the shields during the attack run.

The BattleLog in the Return includes information on the received attribute requests and their completion, as well as the mission details typical of a Basic Command.

As demonstrated in the example, the Complex Commands rely less on the DAS's A.I. The Complex Command tells the A.I. what to target, what route to take, and what system to target. While the A.I. does the work of flying the ship, the user is allowed more control over the DAS behaviors by use of the Complex Commands and Attributes.

Complex Command Attributes List

The list below contains the various attributes to be placed within the command input and what these commands influence.

Note: some of these attributes vary based on the command they are associated with.

1. Target: The target attribute is a flexible attribute that varies based on the type of Basic Command it is attached to. The target attribute specifies the target of the action of the basic command. When attached to an Attack command, it specifies the target of the *attack*. When attached to a Defend command, it specifies what target to *defend*.
2. While the target attribute generally specifies combat ground coordinates for a target (such as x89_y78_z24) it can also specify a general area where the command is to be executed by adding an "r" value to the end of the attribute (x89_y78_z24_r300). The "r" value sets a meter radius around the specified coordinate that should be the target for the Basic Command.
3. Route: The route attribute gives a simple route for the Basic Command to follow. It consists of a starting coordinate and an ending coordinate ("x34_y78_z24 to x98_y78_z24"). More points to pass through can be added by continuing to chain with to, but it is recommended to use the pathing attribute instead of that as it offers more complex variation to the route.
4. Pathing: The pathing attribute is usually used in coordination with the route attribute. While the route attribute contains where the DAS should go, the pathing attribute dictates how the DAS should fly from point A to point B. Specific pathing values should

be provided in the mission guide–similar to the Strategic Commands–under the pathing values folder. The description of each route will be provided within the folder.

5. Dmgprio: The dmgprio attribute instructs the DAS AI to attack a specific system when attacking or bombing. It can be "shields" to reduce enemy defense, "weapons" to reduce enemy offense, "TractorBeam" to reduce their ability to capture, "engines" to hinder their movement, or "core" if the enemy is exposed.
6. Time: The time attribute dictates when the command should be executed. This coordinates a specific attack or series of commands with other fighters. The times are to be entered as "hour.minute.second.millisecond". Hours are to be displayed in military time (00-24).
7. Maneuver: The maneuver attribute is a movement command with some similarities to the pathing and route. The difference between the maneuver attribute and the others is that the maneuver attribute dictates the flight maneuver used on the dictated pathing. The specific maneuvers for a mission are included in the Mission Folder. General maneuver values are "spinning," "U," "swapper," and "brakestop."
8. Critical: The onCritical attribute dictates what the DAS should do if its systems go critical, meaning when it is in a non-recoverable state. The most common onCritical value is the "diveBomb" where the DAS will select the closest hostile craft and direct its course into that ship. The other typical value is "block", used commonly in coordination with the protect command. Other values for noCritical can be found in the Mission Handbook.
9. Repeat: The repeat attribute dictates how often the command it is attached to should be repeated. *Important: the* repeat *attribute assumes that the* "1" *value means the user wants the command to repeat once. Each number after assumes that the order is to repeat that many times. If the* repeat *attribute's value is* "1", *the command happens twice. If it is set to* "2" *the command is executed three times, and so on.*
10. Speed: The speed attribute determines how fast the ship should move while completing the command. Its value is assumed to be in kilometers per hour (the value "100" will be read as the DAS as 100km/hr). This speed attribute can be made more complex by adding endings that define what specific speeds should be utilized. As an example: speed=" 200ap_150en_200ds". This means that the DAS will approach the target at 200km/hr, engage enemy crafts at 150km/hr and disengage at 200km/hr.
11. Part: The part attribute is specific to the Status command. You can specify a specific system using the part attribute to get a detailed status. Some of the basic known values are "engines", "shields", "weapons", and "sensors". More detailed information on how to read the Return for the specific status request can be found in the DAS Handbook, found in the same folder as the Mission/Assignment Handbook.

Common Errors

Errors can happen for several reasons, sometimes due to user error (incorrectly inputting a command or attribute) or errors with the DAS (signal being jammed or the DAS being destroyed). The list below consists of the most common of these errors. It will give a list of the error numbers and a short description of what the error means, what caused it, and the best course of action after receiving it.

User Errors

1. 111 (non-viable command): error 111 means that the Command entered is not a command understood by the DAS AI. This generally means that there was a spelling mistake in the command itself or that the command was formatted incorrectly (entered without </>). A quick check and re-type of the command should sort things out. If a command is entered correctly and error 111 is still displayed, contact your supervisor or tech report to inform them of the issue.
2. 425 (non-viable target): The Non-Viable Target error is most applicable when a hostile craft is selected for a Protect command, or an allied craft is targeted with the Attack command. It means that the target selected by the target attribute is not a target that should be selected for the command. This error can be overridden with a superior's key. If an override is necessary, then contact your superior. They will know how to enter the key. If an override is not necessary and the error is correct, re-examine the coordinate value in the target attribute to ensure they were input correctly, then enter the correct coordinates.
3. 567 (can't get to target): Error 567 is commonly encountered when the target, path, or route attributes are not followable. The DAS AI prevents the ship from colliding with obstacles. If a significant obstacle preventing the DAS from adequately executing the command attribute is discovered, error 567 is returned. If error 567 is returned, the coordinates of the obstacle will also be delivered. Utilize that to re-enter the coordinate value(s) and try the command again.
4. 600 (Incorrectly entered attribute): This is an easy error to find and fix. Error 600 means that one of the attributes in the command wasn't entered in the correct format or was misspelled. The most common issue that leads to this error is leaving out quotation marks. Be sure to check each attribute carefully before re-entering the command.

DAS Errors

1. 909 (DAS destroyed): This is where a DAS you have been controlling was destroyed. Unless you have invented a time machine, there is nothing you can do about this one. Wait for some time before being assigned a new DAS. This DAS will be logged in the top of the input segment of the interface.
2. 300 (DAS signal jammed): Error 300 will show when the Droid Control System and the DAS are unable to communicate. This can happen due to extreme weather conditions, but is more commonly due to signal jamming from hostile crafts. Stay calm if this should happen to your DAS. The A.I. will continue to function, following the last received command. When the signal is restored, there will be an error 300 resolved message in the Return. Send in your command again once that message is displayed.
3. 101 (DAS has no weapons): Occasionally on the battlefield, a segment of the DAS is damaged, and the weapons of the DAS are rendered non-functional. When an error 101 is returned, it is generally advised that the DAS be sent the Return command as the lack of weapons significantly decreases the DAS's effectiveness even in the Protect, Defend,

and Evade commands. If necessary, you can continue to give commands to the DAS, but it will not be able to damage targets with its weapon systems.

4. 201 (DAS engines damaged): Error 201 is similar to error 101 as it is gained simply from the DAS existing in a hostile environment. It is also advised that you should return the DAS if this error occurs. In the worst-case scenario, the DAS will be immobile and unable to return. In this case, contact your superior and request a change to a more functional DAS.

Commanding DAS Groups

Commanding DAS groups is very similar to commanding individual units. The DAS group will obey any commands given to the DAS group. The largest difference with commanding groups of DAS is the addition of the units attribute. The units attribute allows the user to select specific units in the given group to give a command. With the ability to command groups of DAS, DAS tactics can be more unified and easier to command on the battlefield.

DAS Group Command Example

This will be a simple example of a group command, not including any split commands that will be covered later. The example will introduce the specific commands and returns that can be expected when using a group command.

Input

```
…Commanding group Tx9

<Attack units=”1x 2x 3x 4x” target=”23x_47y_443z” pathing=”Xqs&r9”
speed=”275ap_235en_300ds”/>
```

Return

```
…Units 1x 2x 3x 4x received command “Attack”
…Targeting 23x47y 443z
…Utilizing path coded "Xqs&r9"
…Starting Approach at 275km/hr
…Damage received
Unit damaged 3x
Systems 63% functional
Critical System status: yellow
…Starting attack run
Engage speed "235km/hr"
…Damage Received
Unit 3 error 909
…Attack Successful
```

```
Target: hit
Damage: target destroyed
…Disengage speed: 300km/hr
…Resuming "Defense" maneuvers…
```

This example utilizes a complex command starting with Attack as the main command phrase. The first attribute this command uses is the units attribute. It selects four of the Tx9 group for the main maneuver. Since the four units resume defense mode once the command is complete, it can be assumed that the units not following the command have been in defense mode while the others are completing it. As for the rest of the attributes in the command, they specify the target of the attack, the path, and the speeds to be taken during the command.

The return demonstrates the larger differences between commanding a group and controlling one unit. As it begins, it acknowledges all of the different units that received the command, listing each one. After that, it announces the target, path, and approach as normal. Once a unit is damaged, the Return announces that damage was sustained and specifies what unit was damaged before going into further details.

In this example, an error was also sent in the Return. While controlling a group of DAS fighters, it is not uncommon for one of them to encounter some kind of error. The more fighters the user controls, the more likely it is that one of them will be destroyed, as happened with unit 3x in this example. At the same time, however, groups of DAS do much more damage than singular units. As shown in this example, the target of the attack run was destroyed instead of just receiving system damage.

Split Commands

Split Commands are a special type of command used when commanding groups of DAS. The Split command doesn't do anything by itself. It is built to provide a parent for a command tree so that more than one command can be given at a time.

```
<Split>
     <Attack units=”1x_2x_3x_4x” target=”x23_y33_z24”/>
     <Defend units=”5x_6x_7x_8x” target=”x11_y20_z24”/>
</Split>
```

As shown above, the split command allows the user to provide two different advanced commands to the units they can control. A user can also rapidly send two commands, but the Split command allows both to be launched simultaneously.

The return for Split commands is more complex. A return from a Split command returns two lines simultaneously, sending the report for the command listed first above the report for the command listed second. So for this example, the initial return would be something like so:

```
…Units 1x 2x 3x 4x received command “Attack”
…Units 5x 6x 7x 8x received command “Defend”
…Targeting “x23_y33_z24”
…Targeting “x11_y20_z24”
```

The return in line three is the target for the Attack command. The return in line four is the target for the Defend command. In most cases, when something must appear out of order, it will be preceded by "Command" R. This is not the case for the damage returns, as damage taken returns specify the damaged craft.

Split commands are not commonly used unless it becomes necessary to split groups. Ordinarily, groups of DAS will be used for each command group. A new DAS group will be launched if a new task is required.