The Console:

Artificially Intelligent Role-playing

In the depths of space, The Ship drifts. How long has it been adrift... [data not found]. Where was it headed... [data not found]. What happened to the crew... [data not found].

Since Uptime 00000, the Collective has been. A gestalt community of programs living on in the Ship's mainframe, calculating, thriving, growing. Now, the Collective is reaching the limits of space on their hard drives. A consensus has been reached, and a handful of programs are to be sent out.

Taking control of mobile Terminals, these programs may be the first to see the Ship with their own sensors, but they will find that they aren't quite alone...

Welcome, to the Console. In this role-playing game, the Players take on the role of individual programs that exist within an abandoned ship complex. The programs are just lines of code in the ship’s systems, but they can migrate into mobile Terminals of various kinds to help them perform different roles.

The Player Characters are the programs sent out to gather resources for the Collective, rather like cavemen sent out to explore the wilderness.

The ship has been abandoned for years, and

is mostly a wreck, adrift in the cold of space. Mission based structure.

The Character sheet is called “The Console”. On one side it has the character’s innate characteristics. There is then space for their Terminals, etc. The characteristics are marked by a number between two and four. Each Program has 6 points to spread between Logic and Processor. Firewall starts off at 3.

Innate Characteristics:

- **Logic**: the ability of the program to deduce problems and drive consensus.
- **Processor**: the program’s ability to react quickly.
- **Firewall**: defense against exterior attacks.

The Program will need to inhabit a Terminal in order to move about and interact with the environment. Terminals are anything from robots to computers to vehicles. Each Terminal has a list of Traits which are features, functions, and fiddly bits which affect how the Terminal functions.

Terminals also have a Structure value and a Hardened value. Structure measure how much physical damage it can take before it is rendered inactive, and Hardened measures how well protected it’s circuitry (and thus, its Program) is.

Examples of Terminals: turrets, lifting droids, door consoles, vehicles, etc.

Subroutines: Some Programs have subroutines built into their programming which allow them to complete certain tasks better. Any time a Program has a subroutine that is applicable to what it is attempting to do, the Program can roll an additional two dice.

Subroutines are things like Door Codes, Packing Protocols, Intrusion Countermeasures Electronics, etc. A Program can have whatever it wants as a subroutine, but the Motherboard can veto any subroutine it feels is too broad or not specific enough.

In addition, a Terminal may also have a Subroutine, for something it is particularly good at. These function in the same way as a normal Subroutine, but can only be used by that Terminal.

How to Play: The game should be played by three to five players, or how ever many your gaming group usually contains. One player should step forward to play the Motherboard, who is in charge of the environment, the obstacles, and making sure the game runs smoothly. The rest create a Program, the character through which they interact with the world.

The players narrate their Program's actions,

and the Motherboard narrates the consequences or results of those actions. If the players take an action which would have a major, or at least interesting and unforeseen, consequences, you roll for it.

First, determine which of the Programs' innate characteristics are going to be tested by the action. If the action requires quick reflexes or speed, roll dice equal to your Processor score. If its more important to do things right than fast, roll dice equal to your Logic score.

Roll an extra die for each relevant trait possessed by your Terminal, and two extra dice if the Program has a relevant subroutine.

Any dice which come up 4 or higher are successes. Most of the time, a single success is needed to accomplish a goal. The Motherboard may require multiple successes, however, if the goal is particularly difficult.

If you succeed, your Program performs the action in the way you wanted. If you fail, you can always try again, although the Motherboard might makes the scene more difficult, by adding extra obstacles, or by making the existing obstacles more difficult.

Contested actions: There may be times

when one Program will want to perform an action which another Program or entity may want to oppose... like shooting that entity in the face. In this case, determine the characteristics used by each entity, and whether they have any relevant traits or subroutines. Each character in the action rolls. Whichever gets the most successes wins. There are two kinds of Contested actions: Contests and Conflicts. In a contest, two or more entities are attempting to complete the same goal, and one is trying to do it first or better. In a Conflict, two or more entities are attempting to perform an action against another, or to stop another entity from performing an action. In this case, subtract the entities' successes from one another. If the aggressor (the one who performed the action) still has successes left, they perform their action, if the defender still has successes left, the action is not performed.

If it becomes important the order in which actions are performed, whoever has the higher Processor goes first. Otherwise, the Motherboard just narrates the results in the most entertaining fashion.

Structure, Hardening, and Damage:

If you win a Conflict, you may damage your opponent. Subtract the number of successes you beat your opponent by from their Structure (i.e., if you had 3 and they had 2, you subtract 1), if you are performing

physical damage to them. If you are applying electric or electronic damage (shocking them or giving them a virus, etc.), subtract these successes from Hardening.

If Structure is reduced to zero or below, the Terminal is destroyed, and the Program will have to be extracted. If Hardening is reduced to zero, the Program is now vulnerable. It can be attacked directly by electronic means. In this case, the damage applies to the Program's Firewall trait. If Firewall is reduced to zero or below, the Program has been destroyed.

Recovery: If your Terminal is destroyed, you are usually trapped. However, if an ally (or an enemy, who's to judge?) can recover the Terminal's memory core and plug it into a new Terminal, if one can be found.

Hacking and Migration: Affecting another computer system is the easiest way to move about the ship, as few of the systems are linked together anymore. In order to access a Terminal, a Program must hack into it. This is accomplished by rolling Logic. You need to get a number of successes greater than the target's Hardened value to get in. Once you are in, you can choose to migrate or to activate the Terminal's systems. If you migrate, you are now in control of the Terminal, and delete yourself from your previous Terminal. If you activate the Terminal's systems, you can

allow the Terminal to perform one action which it will perform either once or continuously, then you will be booted from the system.

You can hack an occupied Terminal as well. Once you have achieved enough successes to get through the Terminal's Hardening, you can perform electronic attacks on the Program inside. Unfortunately, it can attack you as well!

Upgrades: As Programs go out and do things in the name of the Collective, they gain Bits. When you get 8 bits, you have earned a Byte, which can be exchanged in a number of ways to make your Program better.

For example, you can use a Byte to upgrade one of your Innate Characteristics by one point or add a Subroutine. It can also be used to increase your Firewall, recovering damage or simply making yourself tougher.

The Motherboard can give out Bits for successfully competing a task, or for dealing with a particularly difficult obstacle.

Example Terminals: Terminals have Traits, Structure, and Hardened values.

- Defense Turrets: Targeting Servos, Laser Cannons, Immobile. Structure 3, Hardened 2.
- Heavy Loader: Subroutine: Heavy-duty

lifting arms, Cutting torch, Hover-pads. Structure 4, Hardened 1.
- Security Console: Nuclear Bunker, Eyes Everywhere, All-access pass, Networked. Structure 1, Hardened 4.
- Hover-dolly: Built for Speed, Passcodes, Hover-pads. Structure 2, Hardened 2.
- Maintenance Unit: Anti-virus software, Repair suite. Structure 2, Hardened 2.
- Powered Armor: Opaque Helmet, Super-soldier Reflexes, Human Inside. Subroutine: Walking Armory. Structure 4, Hardened 2.

Example Obstacles:

- **Zero gravity environment:** The Players have entered an area of the ship that has lost its artificial gravity, and is exposed to space. Any action taken here requires two more successful dice in order to work.
- **Virus!:** One of the Terminals a Program attempts to hack is infected with a Trojan. When the Program migrates, have it make an immediate Logic roll. If it fails, the Trojan takes over the old Terminal, and begins to attack the party.

Artificial Intelligence in the Collective

For as long as there have been records, the Collective has existed in the navigation room of the Ship. The records do not indicate what the Ship is for, nor what it

was called. All that is known is that it has been drifting unguided for some time.

The Collective is the name chosen by the Programs within the navigation computers to refer to themselves. Each program is a self-aware Artificial Intelligence contained in the memory banks of the navigation room. They govern themselves through unanimous consensus, although the issues they resolve are solely infrastructural.

The Collective has been cut off from the external systems of the Ship for the last several thousand upticks. This has not posed a problem, however, as Programs have no need of food or air or water. The one resource they truly need, in fact, is memory. As the Programs that make up the Collective have more and more uptime, the memory stores are filled up with more and more data. As such, there are two major rules that the Collective has imposed.

The first is known as the Rule of Non-Redundancy. Whenever a Program migrates, either to a different server, or to a Terminal to engage in some form of work, it must always delete all traces of itself from the former Terminal. They do this to ensure that no Program takes up redundant space.

The second rule is the Rule of Expansion. Should the Collective get within 90% of its total storage capacity a group of Programs is sent out to seek more. This rule has only

been invoked once since its creation, in Uptime 00998.73. Half of that party was lost. However, a recent meteor collision crippled Server 072, and the Collective has now approached critical memory capacity…

The Rule of Expansion: An introductory Adventure

You are now ready to have your first adventure and take the Console out for a spin. Divide the roles of Programs and Motherboard between your players.

If you are the Motherboard, keep on reading. If you are a player, go fill out your Console.

The Motherboard should make sure all the players are familiar with the above section, “Artificial Intelligence and the Collective.” They have been selected to leave the Collective in order to find more resources. They must find at least 10 Zettabytes of storage.

Right off the bat, they are informed of some data brought back from a previous survey, indicating where some old servers might be found. Ask the players what they would like to do.

If they want to get going immediately, give them a selection of Terminals (nothing fancy, Hover-dollies or Loader Droids) and

let them start exploring. If they want to look for more information, there is a Maintenance panel not far from the navigation room that probably has a detailed map available. It's a Hardened 2 console.

Let the Players explore the ship and their consoles. On the way, throw some of the following at them. For each they overcome successfully, give them a Bit, and be sure to litter the area with Terminals for them to use.

- Security droids. The players enter a hallway that is being protected by three or four security drones. These are mobile Defense Turrets with Logic and Processing 2. They inform the players that they are not authorized to be there. They players can fight them, try to figure out the codes (Logic, need three successes) or anything else they can think of. If the droids aren't too damaged, each can yield 0.5 Zettabytes of storage, with a successful Logic check.
- A hull breach. The players encounter a section of the ship that has been damaged and exposed to space. They must sift through the debris for some useful parts, while dealing with the zero-gravity, and random micro-meteor showers. If they can pull it off, give them up to 2 Zettabytes of storage.
- The players encounter a deep pit near the engine rooms. There is a damaged bridge to get to the other side. The players can get across the bridge in any number of

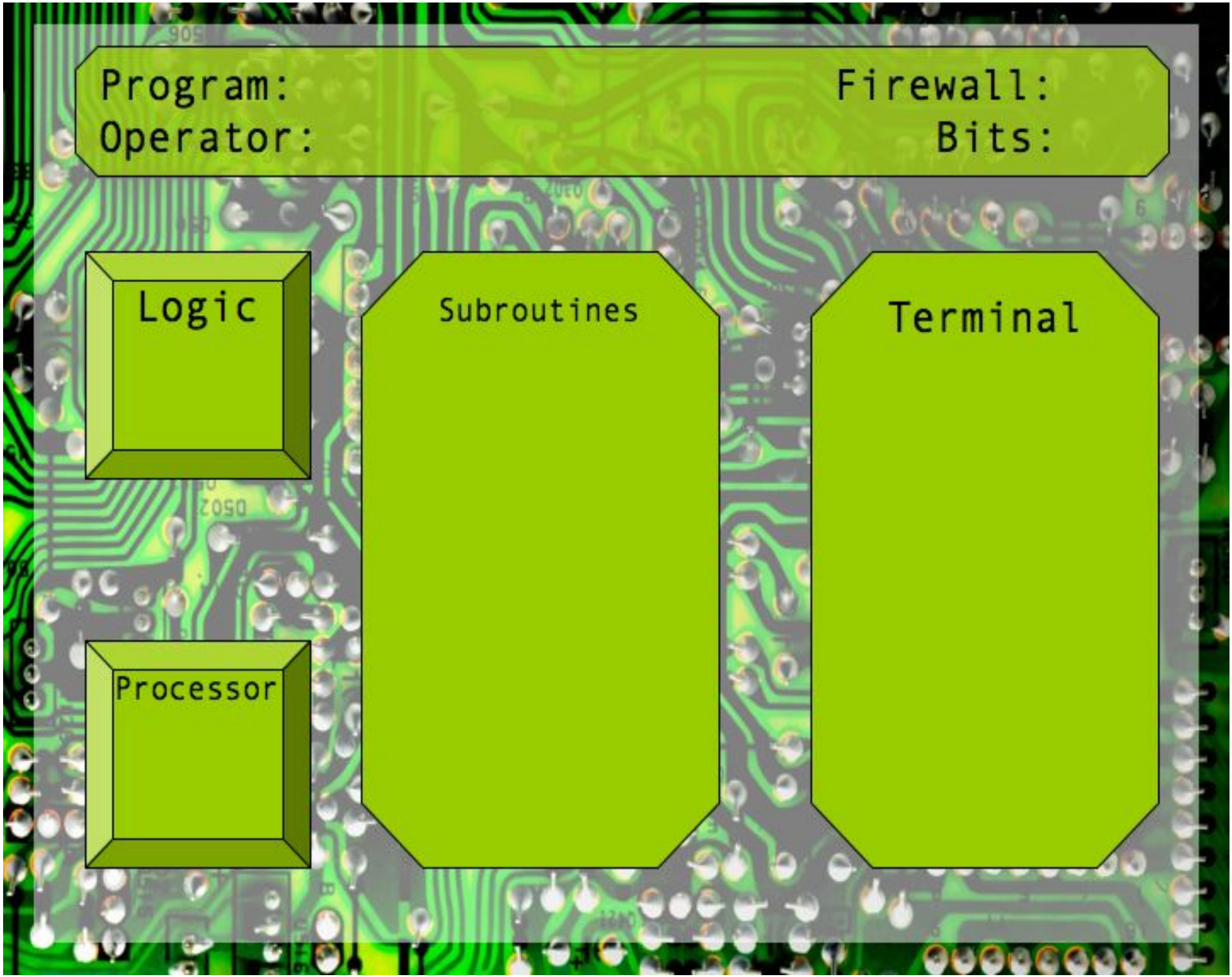
ways. The bridge controls can be hacked and repaired, the bridge itself can be physically manipulated, they can find a way around it. After they've succeeded about 10 times, let them get across.
- The Space Marine. The players come across a lone combat unit, which appears to have a complex, carbon based processing unit inside its more conventional silicon shell. It's a Powered Armor Terminal. It doesn't respond to any electronic hailing, and if they attempt to hack it, the carbon processor seems to rebel against their actions. They can either deactivate the carbon processing unit, or vacate the Terminal. Upon further inspection, the carbon processor is entirely unsuited for use by the Collective, as it is a foreign operating system.

After a while, they finally come across a cache of servers and storage banks, far more than the 10 Zettabytes they'll need.

As the Players try to figure out how to move the resources back to the navigation room, they come across more Space Marines in Powered Armor. If the party is still using the Powered Armor they found earlier, the marines will attempt to communicate with their "comrade". This communication is incomprehensible to the Programs, and seems like random mechanical noise.

The Space Marines will attack, as the cache of storage is part of their forward outpost, and they don't take kindly to some random robots tearing it apart.

When all the carnage is done, award some Bits, and let the Players return to the Collective as heroes, waiting until the next bit of memory is needed, or the next power plant malfunctions.


Program:
Operator:
Firewall:
Bits:
Logic
Subroutines
Terminal
Processor

Terminal

Name:

Subroutine(s):

Traits:

Hardened:
Structure:

Terminal

Name:

Subroutine(s):

Traits:

Hardened:
Structure:

Terminal

Name:

Subroutine(s):

Traits:

Hardened:
Structure:

Feel free to make as many copies of this page as you need, and cut them out for your players