Bureau of the Schola Tactica

The Guardsmen Generator

An Automated Cogitator Protocol for Generating and Processing Squads of Guardsmen

Submitted To:
Ursarkar E. Creed
Lord Castellan
8th Cadian
Segmentum Obscurus

Submitted By :
Justin R. Mansell
Acting General
482nd Cadian
Segmentum Solar

Abstract

The humble soldiers of the Imperial Guard make the faction one of the most relatable in Warhammer 40k. The fact that most guardsmen are little different from modern humans, with all of our flaws and fallibilities, gives the army a powerful narrative appeal. However, the sheer scale of typical tabletop battles makes this difficult to exploit, as most players do not have the time to write backstories for anything other than a few officers. The Guardsmen Generator is a computer program that automates the creation of arbitrary numbers of unique soldiers. Players can use the program to track injuries, rivalries, medals, and more, which the program generates automatically based on the results of actual Warhammer battles. Not only does this facilitate an emergent narrative, but it allows the analysis of many interesting statistics about life in the Imperial Guard.

Contents

1 Introduction

Miniwargamer Josh points out in his 22,000 pt Astra Militarum showcase[1] that one of the largest attractions of the Imperial Guard to Warhammer 40k players is the army's relatability. While many players enjoy putting themselves in the shoes of superhuman soldiers, maniacal fiends, or bestial aliens, anyone can imagine themselves in the boots of a guardsman. The recognizable nature of their equipment combined with fact that these guardsmen are as fallible as humans have always been adds human touch to an otherwise inhospitable setting. Even more importantly, the idea of overcoming impossible odds through teamwork is, for me, one of the most inspiring themes in Warhammer 40k, and one that the setting shares with many other war stories of both fiction and nonfiction.

My goal with this project was to add a further touch of life to the guardsmen of my Warhammer 40k collection. As someone who has always been a more narrative-minded player, the central attraction of the hobby for me is the stories it creates. Be it a lone survivor who endured to secure an objective or a squad of scions whose bravery failed them at the critical moment, winning or losing is secondary to weaving the die rolls into a tale of sacrifice and heroism, rivalry and friendship, and tragedy and triumph.

The Guardsman Generator is a computer program designed to procedurally generate arbitrary numbers unique soldiers. Users can create squads of infantry or tank crews, then generate injuries, medals, rivalries, and more, based on the results of real Warhammer 40k battles. By automating these functions, it becomes possible to track the fate of literally hundreds of soldiers, each with their own backgrounds, demeanors, ambitions, comrades, skills, and flaws. After several battles, a rich and dramatic narrative will emerge before the user's eyes – one that they can expand on as much or as little as they like.

The scope of this program also makes it possible to collect statistics that can help answer essential questions about what it is like to be in the Imperial Guard. For example, what percentage of guardsmen survive their first battle? How big is the skill difference between veterans and raw recruits? What percentage of guardsmen have earned a medal?

This report describes the Guardsmen Generator program and some of the statistics that I have collected with it. Section 2 outlines the workings of the program. Section 3 presents an example of its use. Section 4 presents statistics that I have collected from my own games as well as MiniWarGaming battle reports. Finally, the attached appendix describes the limitations and tips for using the program.

[1] `https://youtu.be/Aas4hE9xrg4`

2 Overview

The Guardsmen Generator is written in the Python programming language and consists of 3 codes:

- RandomGuardsmen.py – contains the data structures for representing soldiers, as well as many of the options for the various parts of their profiles
- NewSquad.py – generates new squads of guardsmen, tank crews, and scions
- PostBattle.py – processes the effects on a squad after each battle, based on a user input regarding the number of losses and victory points earned by the squad

2.1 Profiles

The Guardsmen Generator programs represent each soldier with a profile containing the following fields:

- Sex: this is used to generate each soldier's name, but is not tracked thereafter[2]
- Name: randomly generated based on the soldier's sex
- Age: randomized between 18 and 28, but has a random chance to be as young as 16 and as old as 65
- Rank: Cadet, Guardsman, Corporal, Lance Corporal, Sergeant, Lance Sergeant, but this can also be a custom rank entered by the user. The default rank for most soldiers is "Guardsman."
- Battles: the number of battles to which the soldier has been deployed
- Personality: a one or two word description of the soldier's demeanor
- Background: a one or two word description of the soldier's backstory. It comes in two categories: low-born (most soldiers) or high-born (most scions).
- Ambition: a short description of the soldier's greatest goal in life. It can be as simple as "get promoted" or as lofty as ... well... "survive"
- Skills: a list of talents, abilities, or traits that could make the soldier more likely to survive, or just make the them look cool

[2]Note that this can result in some questionable behavior later on if the soldier gains certain skills, since some of the "skills" include things like mustaches and beards. You may call this a bug. I call it a feature. After all, it's the beard on the inside the counts!

- Flaws: injuries, traits, or habits that are likely to get the soldier killed
- Confirmed kills: the number of kills by this soldier. These are not explicitly generated by the program, but can be entered manually by the player. The program will track the entered value thereafter. As such, it is mostly reserved for exceptional cases where a specific kill can be traced back to a specific soldier (such as a sergeant making a particularly noteworthy kill).
- Awards: a list of distinctions and medals earned by the soldier
- Friends: a list of other soldiers in the same squad who have a positive opinion of this soldier
- Rivals: a list of other soldiers in the same squad who have a negative opinion of this soldier
- Remarks: a log of notable events in the soldier's career, such as promotions and injuries

2.2 Generating Squads

The program "NewSquad.py" generates a text file when run that contains the soldier profiles for the new squad. Upon running the program, the user can specify how many soldiers should be in the squad . An optional code letter after the number also provides some options:

- C = Conscripts. All soldiers in the squad begin at the rank of cadet by default.
- V = Veterans. All soldiers in the squad begin at the rank of corporal by default and have the chance to begin with more skills than a normal guardsman.
- S = Scions. As veterans above. Additionally, soldier backgrounds are generated from the "high-born" backgrounds.
- < Anything except G > = custom squad. This generates a squad as per normal, but does not include a squad sergeant. This is useful for generating units such as heavy weapon platoons.

2.2.1 Basics

NewSquad.py repeatedly calls functions from RandomGuardsmen.py to flesh out the requested number of soldiers. First it generates the basics: name, age, sex, rank, and an initial remark stating when the soldier was deployed to active duty. Next, it

generates the soldier's personality and ambition by choosing a random entry from the list of possible demeanor's and goals stored in RandomGuardsmen.py. Following this, the program initializes the soldier's skills and flaws by randomly selecting 0-2 skills and 0-2 flaws from the lists of potential starting skills and flaws.

2.2.2 Backgrounds

After generating the basics, the next thing to be generated is the soldier's background. This can come from one of two lists: low-born, or high-born. Only scions are generated from the high-born list, but there a random chance (1%) that the program will choose a background from the "wrong" list. No one's destiny is entirely determined by the caste of their birth, and so this makes it possible (although rare) to see nobles enlisted as guardsmen, or find hive scum in the ranks of scions. The details of how this happened, of course, are left to the player's imagination.

The soldier's background also has an influence on their skills, flaws, and demeanors. Each background option has an associated list of token skills, flaws, and personalities, which represent the most common stereotypes of someone of that upbringing. For example, voidborn:

```
token_skills = ['Lucky','Strong willed','Technical knock']
token_flaws = ['Fragile','Tainted','Ship bound','Mistrusted']
token_personality = ['Superstitious','Awkward','Steely']
```

Each new soldier has a 25% chance of gaining an additional skill from their backgrounds token list, 25% for a token flaw, and 25% of a token personality (overwriting the personality generated previously). Therefore, about $1 - (1 - 0.25)^3 = 58\%$ of soldiers will possess at least one skill, flaw, or personality indicative of their background.

2.2.3 Friends and Rivals

Once all soldiers in the squad have been generated, the program randomly assigns between 0 and 2 friends and 0 and 2 rivals to each soldier. The number of friends and rivals is modified by the soldier's personality and skills. An affable cook is likely to have more friends than a cowardly slacker.

2.2.4 Squad Leader

By default, all soldiers begin at the rank of guardsmen (or corporal for veterans and scions). For squads that require a leader, the program promotes a random member of the squad to the rank of sergeant, so long as they are at least 21 years of age and have at least 1 skill. Additionally, some soldiers may have been given additional training (such as special weapons) and obtained the rank of corporal. Each soldier who is at least 18 years old has a 20% chance to begin at the rank of corporal.

The final output is a .txt file containing profiles for all of the members of the squad. The squad file is labeled with a random 3-digit number but can be renamed by the user for better organization.

2.2.5 Custom Entries

Users are free and even encouraged to customize the computer-generated soldiers. So long as the profile format is adhered to, one can edit the entries and replace them with anything they like. Want to add some confirmed kills? Do it. Have a good idea for an alternative personality descriptor? Write it. Want to make a profile for yourself and see how you'd fare in the 41st millennium? Go for it. Just make sure to avoid any fancy typesetting characters (such as stylized quotation marks), as the program can only read Unicode characters when loading the profiles. It is also important that there are exactly two spaces after the end of the last profile in the file.

2.3 Battle Results

Optimistic and naïve, the output of NewSquad.py represents a squad that is fresh out of training. Unfortunately, they won't stay that way for long.

PostBattle.py processes battle events for the squad given several inputs from the user (in order):

1. Squad: the name of the squad file (without the .txt) to be processed. The file must be located in the same directory as PostBattle.py.
2. Certain deaths: the number of soldiers the squad lost who are definitely dead, with no chance they were just wounded. It is useful for when members of the squad are felled by weapons of overwhelming power (e.g. titan weapons), or other cases (e.g. summary execution) where the soldier surviving with only an

injury would pose a paradox. If you want to see your guardsmen live, use this sparingly.

3. Normal casualties: the number of losses in the squad who may be either dead or wounded
4. Morale casualties: the number of soldiers in the squad who fled the battle
5. Victory points: the number of victory points the squad earned in the match. This determines the number of medals and promotions allotted to the squad. Alternatively, the player can set this to a value that reflects how much the squad contributed to the battle. 0 = no particular contributions to 3+ = truly exceptional performance that altered the course of the battle.

An example input could be:

```
PostBattle.py squad001 1 4 2 2
```

which indicates that squad001 had a 1 soldier who was definitely slain, 4 others who became either wounded or dead, 2 soldiers who fled, and earned 2 victory points.

2.3.1 File Backup

Before it does anything, PostBattle.py makes a copy of the original squad file. This way, if you make a mistake inputting the numbers (or don't like the results if you're being cheeky), you can always go back to the squad file as it was before being modified by PostBattle.py.

2.3.2 Certain Deaths

Certain deaths are allocated to the squad by randomly selecting members of the squad to slay. This continues until the number of certain deaths reported by the user has been met. However, not all soldiers are equally likely to survive. Upon being marked for death, each soldier has the chance to make a saving throw, representing them dodging into cover, avoiding the commissar's wrath, etc. The percentage chance of this save is calculated as follows:

$$\%\ Save = (number\ of\ skills)\text{–}(number\ of\ flaws) + (number\ of\ battles) \quad (1)$$

Since the "certain death" category is meant to represent deaths due to exceptional circumstances, a soldier's experience can only do so much. Therefore, the save is capped at a maximum of 50%. If the saving throw is passed, the soldier is not marked for death and the program continues to draw randomly from the squad. Since the

program must continue this process until it has fulfilled the number "certain deaths", it is possible that the same soldier may be selected again, in which case they must take another saving throw or be slain. The maximum number of iterations has been capped at 1000 in order to avoid the possibility of an infinite loop, which could be caused, for instance, if a soldier somehow[3] ended up with a 100% save.

2.3.3 Normal Casualties

After allocating the "certain deaths," PostBattle.py moves on to normal casualties. This proceeds similarly as above, except that each soldier's basic saving throw is capped at a maximum of 75%. Upon being selected as a potential casualty, the program also randomizes a 50/50 chance as to whether the soldier is dead or injured. The soldier then has the chance to make several more saving throws based on their profile:

- Toughness: if the soldier has the "Tough as nails" or "Hardy" skills, there is a 50% chance that they will ignore an "injured" result altogether. Similarly, if they have either the "Unshakeable Faith" or "Die Hard" skills, there is a 50% chance they will convert a "Killed" result into "Injured."
- Comrades: the soldier's comrades have a chance to tackle the soldier out of danger, or otherwise look out for them. The program loops through the rest of the squad, giving each member a 1% chance of coming to the soldier's aid. This probability increases up to 31% if the comrade is a friend of the soldier, and/or has certain skills/personalities such as "Mentor", "Trustworthy", "Observant", and the like. If a soldier passes a saving throw due to a comrade they are not already friends with, their savior automatically gains the soldier as a friend. If their savior was a rival, there is a 50% chance that the soldier will have a new found respect for them and remove them as a rival.
- Surviving against the odds: once all other saves have been made, the soldier has a 1% chance to survive due to extraordinary circumstances. The details are left to the imagination, but the soldier's fate is set to "Injured" and they immediately gain an extra skill. This counts as fulfilling a casualty.

As before, if any of these saves are passed, the soldier is returned to the squad and the program continues to select casualties at random until the number of casualties has been fulfilled.

For soldiers who are merely injured, they are temporarily removed from the squad to process the results of their injury. This is summarized by the following steps:

[3] *cough* Sly Marbo *cough*

1. There is a 10% chance they gain the "Awesome scar" skill
2. There is a 50% chance they gain a critical injury, which is randomly chosen from the following: missing eye, disfigurement, missing leg, missing arm, old wound, concussion, mostly deaf. The injury is added to the soldiers list of flaws.
3. If the soldier suffered a missing eye, leg, or arm, they may qualify for a bionic replacement. If they are a rank higher than corporal or meet a certain "bad-ass metric,"[4] then they add the appropriate bionic to their skill list and remove the associated injury from their flaws. Otherwise, if they have lost a limb or both eyes, they have become an invalid and are marked for discharge due to their injuries.
4. Soldiers not marked for discharge are returned to the squad.

2.3.4 Graveyard

Soldiers marked for death are removed from the squad and deleted. But first, their profile is copied to another file called "Graveyard.txt." This file logs all of the soldiers who have been killed. In addition to the normal entries of their profile, the log also saves the name of the squad they served in as well as a new entry called "Legacy." The legacy represents how the soldier will be remembered. Legacies range from "Anonymous sacrifice" to "Inscription on a comrade's weapon" to becoming a local deity. The impressiveness of the soldier's legacy is determined by calculating the following number:

$$Legacy\ score\ =\ 1\text{d}6 + 2 * (number\ of\ awards) + (number\ of\ friends) \quad (2)$$

2.3.5 Fled Soldiers

Morale losses are also handled in a similar way. The program randomly selects soldiers from the squad and tests their saving throw until it has generated the correct number of morale losses. The save percentage is ten times the number of battles they have participated in, but this is capped at either 25%, 50%, or 75% depending on the soldier's skills, personality, and ambition. For instance, someone with the heroic personality who wants to "die for the Emperor" will have a maximum save of 75%, whereas another with the "coward" trait will have a maximum save of 25%.

[4] $(number\ of\ battles) + (number\ of\ skills) + (number\ of\ awards)$–$(number\ of\ flaws) > 3$

As with casualties, the soldier's comrades have the chance to make them "snap out of it" and get them back in the fight. If the soldier still flees, there is a 25% chance that they will gain each member of the squad as a rival, or lose them as a friend. Soldiers that fled have a 50% chance of being demoted and a 20% chance of being either executed, sent to a penal legion, or converted into a servitor.

Finally, soldiers with the ambition "desert at the first opportunity" have a 50% chance of attempting to completely desert if they are selected as a morale casualty. The results range from being caught and executed to never being seen again.

2.3.6 Trauma

In addition to physical injuries, the program also simulates the mental strain of 41st millennium warfare. Soldiers have a random chance to gain new flaws such as nightmares and unbearable guilt, and even take on grim new personalities such as "numb" or "nihilist". The probability of this happening is related to:

- Whether the soldier lost any of their friends in the battle
- Whether the soldier fled the battle

If both of these occurred, the probability of gaining a new flaw is particularly high.

2.3.7 Promotions

Once all of the previous battle results have been generated, the program allocates promotions to the survivors. The number of promotions allotted to the squad is:

$$Number\ of\ promotions = \text{1d2–1} + 2 * (number\ of\ VP\ earned\ by\ the\ squad) \quad (3)$$

Promotions are first offered randomly to the uninjured members of the squad. If this is not enough to fill the number of promotions gained, the program then offers promotions to members of the squad who were wounded.

The highest rank tracked by the program is Lance Sergeant. If a soldier at this rank is offered a promotion, there is a 50% chance that they are promoted to an officer and granted leave to pursue additional training (see Discharge subsection). User's can customize the rank of a soldier to anything they wish (useful for embedding an officer inside of a squad), but the program will not offer them any promotions.

2.3.8 Awards

In addition to promotions, the program also automatically awards medals to surviving soldiers. The medals and the criteria for awarding them are described below.

Eagle Ordinary: awarded for heroism beyond the call of duty. To award this medal, the program calculates a "heroicness" metric for the battle, which is:

$$Heroicness = VP * (fraction\ of\ the\ squad\ that\ died) \tag{4}$$

In other words, if the squad earned a lot of points but sacrificed a lot to do it, they were probably pretty heroic. If the heroicness was > 0.6, then each member of the squad has a random chance to gain the Eagle Ordinary medal. The probability of gaining this medal ranges from 10% to 50% depending on the soldier's skills and temperament.

Macharian Cross: awarded for intelligent application of the *Tactica Imperialis*. This is awarded based on the "tacticalness" of the battle, which is:

$$Tacticalness = VP * (fraction\ of\ the\ squad\ that\ survived) \tag{5}$$

In other words, if the squad earned a lot of VP and didn't suffer many losses doing so, it is probably because they used good tactics. If the tacticalness was > 0.99, then each member of the squad has a random chance to gain this medal. The probability ranges from 1% to 30% depending the soldier's skills and temperament.

Ribbon Intrinsic: awarded to the squad for playing a pivotal role in the battle. If the squad earned 2 VP or more, every member of the squad gains this award.

Medallion Crimson: awarded to soldiers who have suffered crippling injuries. Soldiers that have suffered disfigurement or lost a limb or an eye are awarded this medal.

Triple Skull: awarded to soldiers whose units have suffered heavy losses. If more than 66% of the squad was killed, each of the survivors gains this medal.

Long Service Badge: awarded for longstanding service to the Imperial Guard. If a soldier survives 5 battles, they gain this award.

Merit of Terra: awarded for continuing to serve in the Imperial Guard despite being eligible for retirement. If a soldier is declines retirement, they gain this medal.

Many more medals exist in the Imperial Guard, but these are the ones distributed automatically by the program. As always, users can add custom medals to the "Awards" entry of soldiers they feel are deserving.

2.3.9 Skills and Flaws

Soldiers that survive have a chance to gain new skills after each battle. The probability of gaining a new skill is related to how many skills they already have. New recruits will learn quickly, while grizzled veterans will progress slower. The formula is:

$$\%\ Chance\ of\ gaining\ a\ new\ skill = 100\%/(number\ of\ skills\ they\ already\ have{+}1) \tag{6}$$

Soldiers can also spontaneously gain new flaws if their unit suffered heavy losses. The chance of gaining a new flaw is:

$$\begin{aligned}\%\ Chance\ of\ gaining\ a\ new\ flaw = [&(number\ of\ squad\ members\ who\ died){-}\\ &(number\ of\ battles\ the\ soldier\ has\ participated\ in)]/\\ &(size\ of\ the\ squad) * 100\%\end{aligned} \tag{7}$$

Soldiers cannot possess multiple instances of the same skill or flaw.

2.3.10 Discharge

Soldiers that have suffered crippling wounds and who have not been approved for a bionic are granted a discharge to a noncombat unit. They are removed from the squad and their profile is copied into a file "Alumni.txt." This file tracks all of the soldiers who have survived their time in the Imperial Guard. Soldiers who are promoted to officers are also sent here.

Furthermore, soldiers that survive 10 tabletop battles are considered to have served their duty to the Emperor and are allowed to retire. Depending on their ambition (e.g. impress the commissar, die for the Emperor, kill xenos, etc.), some soldiers may decline retirement and continue to serve. However, most will accept. Retired soldiers are also removed from the squad and copied to the alumni file.

2.3.11 New Recruits and Socializing

Upon loading the squad for processing, PostBattle.py stores the original size of the squad. If soldiers have died in the battle or been discharged, it will recruit new randomly generated soldiers to restore the squad to its initial strength. By default

these new recruits are guardsmen, but if the squad file name contains "Scions" or "conscripts," it will recruit soldiers of these types instead.

With the squad restored to full strength, each soldier has a random chance to gain new friends or rivals based on their personality. This operates in a similar manner to when the squad is first formed. The old squad file is overwritten with the new profiles.

3 Example

A typical output from the program is as follows:

```
python PostBattle.py squad655 0 3 2 1
------------------BATTLE RESULTS----------------------
Susan Powell saved Keith Esquivel
Keith Esquivel admires  Susan Powell
Cara Price was wounded.
Ricardo Orozco saved Susan Powell
Susan Powell admires  Ricardo Orozco
Heidi Hansen was killed in action.
Susan Powell saved Ricardo Orozco
Ricardo Orozco admires  Susan Powell
John Harper was killed in action.
Mary Brewer succumbed to panic
Mary Brewer fled the battle. Disgraceful!
Keith Esquivel  dislikes  Mary Brewer
Ricardo Orozco  dislikes  Mary Brewer
Ricardo Orozco shivered uncontrollably
Ricardo Orozco fled the battle. Disgraceful!
------------------------------------------------------
-------------------AFTER EFFECTS----------------------
Cara Price gained a critical wound: Disfigurement
Cara Price was awarded the Medallion Crimson for their injury
Mary Brewer suffered lingering psychological damage:  Hesitant
Ricardo Orozco suffered lingering psychological damage:  Nightmares
Susan Powell was promoted to:  Corporal
Taryn Piercy was promoted to:  Corporal
Rene Robinson was promoted to:  Lance Corporal
Mary Brewer was demoted to:  Cadet
Ricardo Orozco was Executed for desertion
```

```
Danielle Cowden gained a new skill: Nerves of steel
Keith Esquivel gained a new skill: Nerves of steel
Cara Price gained a new skill: Unshakeable faith
--------------------------------------------------
--------------------Downtime----------------------
Joe Moore joined the squad.
Elizabeth Cramer joined the squad.
Mary Hoskins joined the squad.
Rene Robinson became rivals with Cara Price
Rene Robinson became rivals with Mary Hoskins
--------------------------------------------------
```

The most interesting aspect of the program, however, are not the individuals battles, but seeing how the soldiers change over time.

The first guardsman to successfully finish their career in my summer campaign was Sergeant Fred Hudgins. Compare his retired profile with his original one.

```
Name: Fred Hudgins
Age: 26
Rank: Sergeant
Battles: 0
Personality: Gambler
Favorite food: corpse-starch
Wants to: travel the world
Skills: ['Awesome scar', 'Lightning reflexes']
Flaws: ['Coward']
Liked by: ['Anthony Catalano']
Disliked by: []
```

```
Name: Fred Hudgins
Age: 26
Rank: Lance Sergeant
Squad: squad68
Battles: 10
Personality: Numb
Favorite food: corpse-starch
Wants to: travel the world
Skills: ['Awesome scar', 'Lightning reflexes', 'Combat master',
'Ambidextrous', 'Unshakeable faith']
Flaws: ['Coward']
Liked by: []
```

```
Disliked by: []
Confirmed kills: 2
Awards: ['Triple Skull', 'Service Badge', 'Triple Skull', 'Ribbon Intrinsic',
'Winged Skull']
Remarks: ['Wounded in combat 2018-05-08', 'Wounded in combat 2018-05-30',
'Awarded the Triple Skull 2018-05-30', 'Awarded the Service Badge
2018-05-30', 'Wounded in combat 2018-06-05', 'Promoted to Lance Sergeant
2018-06-05', 'Awarded the Triple Skull 2018-06-05', 'Demoted to Sergeant
2018-06-11', 'Promoted to Lance Sergeant 2018-06-21', 'Awarded the Ribbon
Intrinsic 2018-06-21', 'Awarded the Winged Skull 2018-06-21', 'Granted
honorable discharge 2018-06-26']
```

Note that the Guardsmen Generator program was always a work in progress, and some fields (such as the background) were not introduced until later in the campaign.

4 Battle Statistics

During the summer of 2018 I played 18 Warhammer 40k tabletop battles and applied the Guardsmen Generator program to my various units, including but not limited to: infantry, weapons platoons, scions, tanks, and chimeras. In addition to the core Guardsmen Generator programs, I developed a code called "Statistics.py" to analyze aggregate statistics about these units. Following the 18 games I played in person, I also applied the program to 3 "Bearded Batrep" battle reports from Miniwargaming's Youtube channel. The plots below present a brief analysis of the statistics collected from these 21 battles.

To use Statistics.py, one must manually input the squad names to be analyzed and the number of battles fought at the start of the program.

4.1 Battles Survived

Figure 1 should not surprise anyone who is familiar with the Warhammer 40k setting. Out of all of the soldiers who fought and died in my regiment, those who were killed in their first engagement make up the single largest category. This does not necessarily mean that most soldiers die in their first battle, because notice that the sum of categories 1 through 11+ is still greater than category 0. However, the plot makes it clear that the turn-over rate for new recruits is extremely high, with fewer and fewer soldiers surviving to more veteran categories.

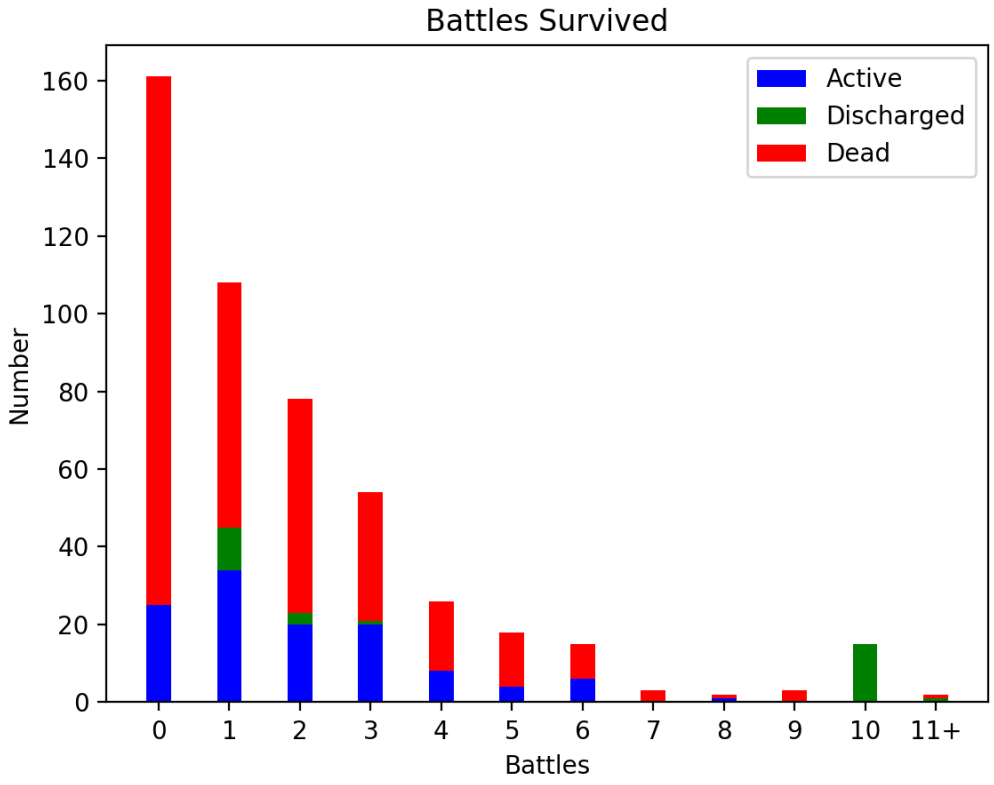

Figure 1: Distribution of battles survived.

Soldiers who survive 10 battles usually retire, which explains the peak in survivors at category 10. Discharges are also common for soldiers who survive only 1 or 2 battles. This is due to novice soldiers being denied bionics and becoming invalids. Soldiers who survive longer have proved their worth and most often are granted bionic replacements if they suffer a crippling injury.

Out of all 21 games, only 2 soldiers declined retirement. One of them, Sgt. Joseph Crump of guardsman squad 068, later died on his 13th engagement (spooky, eh?). The other, a Leman Russ commander named Bernard Gibbs, eventually retired as an officer after 13 battles.

4.2 Ranks

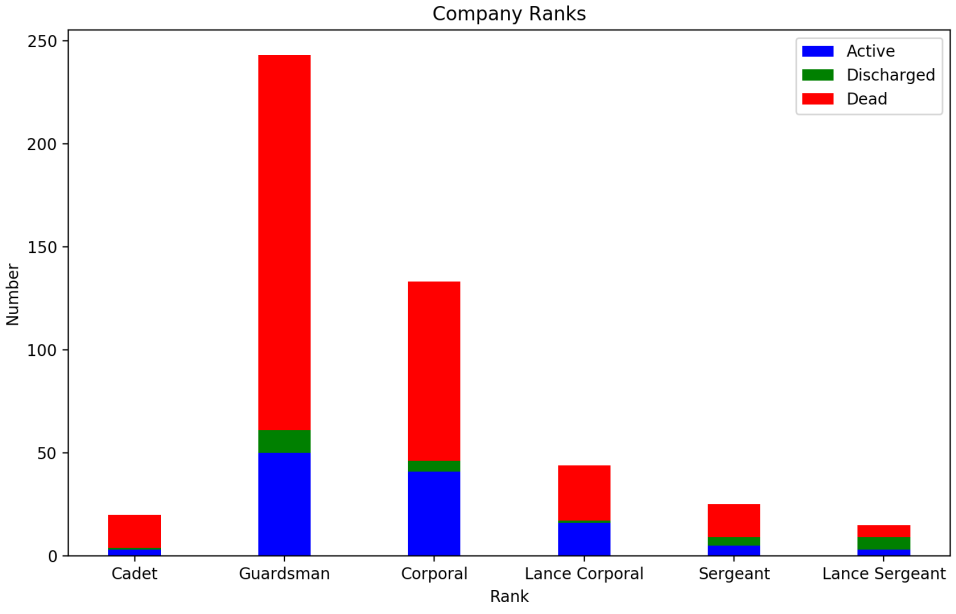

Figure 2: Distribution of ranks.

The rank distribution as illustrated by Figure 2 is also not surprising. Guardsmen are the most populous category, both in the number actively serving, and the number who were killed at that rank. I tend not to field conscript platoons, so the proportion of cadets is low and mostly made up of under-age guardsmen and soldiers who have been demoted.

By comparing the proportions of discharged versus dead soldiers of each rank, we can

gauge the survival rate of each. Clearly, most cadets are doomed, but interestingly, so are most Lance Corporals. This results from the fact that their proven mettle makes most eligible for bionics, and therefore unlikely to be discharged due to injuries. Furthermore, Lance Corporal is very much a "mid-career" rank, and those who do survive are usually promoted and exit the Imperial Guard (alive or dead) at a higher rank. Sergeants and Lance Sergeants also receive many bionics. However, soldiers at these ranks typically have a high skill-to-flaw ratio and a respectable number of battles to their record[5], making them much more likely to survive until the end of their career. Out of all of the ranks, the only one that survives more often than not are the Lance Sergeants.

4.3 Awards

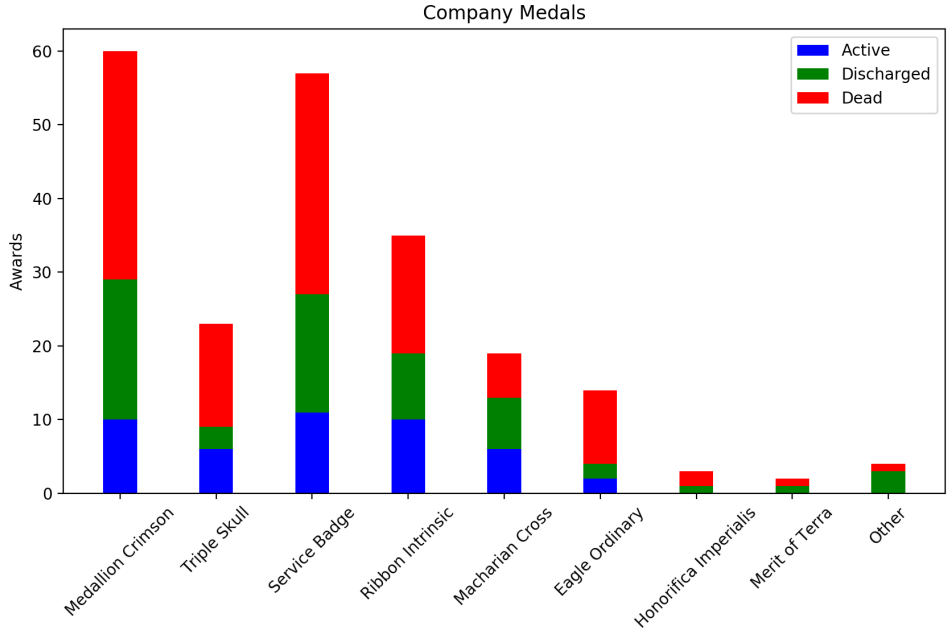

Figure 3: Distribution of awards.

The distribution of medals gives an idea of their prestige and the relative difficulty of earning each. Medallion Crimsons are "a dime a dozen," probably because the only requirement to earn one is to get hurt[6] – something that is quite easy to do on a 40k

[5]Meaning they also need to survive fewer battles to make it to retirement

[6]To clarify a particular nuance, you need to get hurt *bad*, but not so bad that you die. That's the hard part.

battlefield. Service badges are also quite common, but there nothing particularly heroic about mere survival. The Ribbon Intrinsic is the next most common. While impressive, it is awarded to every member of a squad, and so it is possible to earn one by freeloading off of some of your more heroic squad mates. The Triple Skull is a bit more rare and is mostly the purview of small squads (where the 66% casualty requirement can be reached with only a handful of deaths) or ones with particularly high turn over rates (e.g. infantry squads). Notice the low proportion of Triple Skull earners who survive to be discharged. Clearly, to earn one of these medals is to put your luck to a very lethal test. Do so, and your luck will soon run out.

In contrast, the Macharian Cross earners have a high survival rate. It turns out that good tactics are more than just prissy gossip and can keep you alive. There can be no better proof of the efficacy of the *Tatica Imperialis* and the Uplifting Primer.

Unfortunately, the same cannot be said for the Eagle Ordinary. Earning this medal requires facing the enemy head-on and leaving no man behind. The low survival rate should need no explanation. Several I even awarded posthumously.

In a similar vein, the Honorifica Imperialis – a custom-added medal awarded for legendary heroics – is rarely awarded in life. The sole living recipient was Scion Tempestor Gregory Valentine, who tied a battle against the Death Guard 14-14 by contesting an objective after single-handedly besting several plague marines.

4.4 Other Statistics

In addition to the above plots, Statistics.py also prints out the following interesting and fun statistics.

```
Number of active soldiers: 118
Number of discharged soldiers: 31
Number of fallen soldiers: 336
Company turn over rate per battle (%) 14.8
Mean skill deficit for active troops: 0.80
Mean skill deficit for dead troops: 0.45
Average number of battles survived: 1.60
Survival rate (%): 8.45
Percentage of soldiers who survive their first battle: 70.4
Percentage of soldiers who survive 3 battles: 35.2
Percentage of soldiers who survive 5 battles: 16.0
Average number of medals: 0.46
Percentage of soldiers with at least one award: 26.8
```

```
The most popular soldier is Michael Rank from squad: HWP281
The least popular soldier is Joseph Hong from squad: HWP281
The most bad-ass soldier is David Davis from squad: squad705
The least bad-ass soldier is Walter Bledsoe from squad:  squad68
The oldest soldier in the regiment is Brian Hendrickson (age 63 )
from squad squad705
The saddest soldier in the regiment is: Michael Duncan from squad
Scions475 who has lost 4 friends
```

4.4.1 Remarks

- The company turn over rate reflects the average fraction of the regiment that must be replaced by new recruits after each battle.
- The "skill deficit" is the average number of skills minus flaws. Clearly, the "survival of the fittest" is taking effect, as those who have survived are, on average, more skilled than those who did not.
- More than two thirds of soldiers will survive their first battle. However, fewer than 1 in 6 will reach the status of veteran (5+ battles).
- If we remove invalids from the survival rate percentage, we find that fewer than 1 in 20 soldiers will survive to retirement.
- Note that these statistics are aggregates; the numbers can vary significantly between squads. Units such as Leman Russ crews generally have a "high" chance of survival, while others such as infantry squads have much lower rates of survival.

5 Conclusions

The Guardsmen generator allows the user to procedurally generate large numbers of unique soldiers to provide a new layer of narrative depth to the Warhammer 40k tabletop game. By tracking just a few simple numbers during each match, the program can generate a description of the battle's events and their effects on the squad. This gives rise to an evolving narrative, and imaginative players will inevitably find a captivating story of tragedy, heroism, and sacrifice embedded in the results. The ability to process the careers of hundreds of guardsmen also enables the collection and analysis of interesting statistics about the Imperial Guard. I believe that these capabilities add a new layer of enjoyment to the Warhammer 40k

hobby, and further cement the Astra Militarum's reputation as the most relatable faction in Warhammer 40k.

6 Acknowledgement

I wish to thank the members of my local gaming group in Calgary, who took an interest in this project of mine. I omit their last names out of respect for their privacy. Thank you, Liam, Martin, Garnet, and Brendan for your suggestions and support. Many of the skills, flaws, personalities, backgrounds, and medals used by the Guardsmen Generator have been inspired by ones that appear in Fantasy Flight Games' *Only War* roleplaying game. Therefore, I would also like to thank the authors of this game for their inspiring creativity.

7 License

8 Appendix

8.1 Limitations

8.1.1 Model Ambuguity

Casualties are handled by the program in a way that is completely independent from which models the player removes as casualties on the tabletop. Therefore, during a game, it is generally not possible to single out any model as corresponding to a specific individual in the squads generated by the program. This is by design. As a campaign progresses, players will inevitably pick favorites, and the ambiguity of which model corresponds to whom leaves the control of who lives and dies firmly in the hands of an indifferent computer program. Only once the post battle results have generated can the player think back and interpret which model represented whom. Indeed, such retroactive identification was how I elected to assign "confirmed kills."

8.1.2 Imperfect Simulation

The Guardsmen Generator is designed to enhance the enjoyment of Warhammer 40k tabletop games. It is not intended to be a rigorous simulation of guardsman life in the 41st millennium. Most, if not all, of the formulas used to control the program are arbitrary, and each has been subjected to a certain amount of tweaking in order to strike a balance between portraying the grim nature of 40k and still getting to see a soldier retire honorably every once in a while.

8.2 Tips

8.2.1 Original Copies

When generating a new squad, I suggest keeping a copy of the original roster. Later on, you can compare your soldiers with their original profiles to see how they have changed.

8.2.2 Guardsman of the Week

After every couple of games, I selected a notable soldier from one of my squads and fleshed out a paragraph or two about their story. I posted these on my local wargaming groups' Facebook page as "Guardsman Spotlight."

8.2.3 Vehicle Casualties

I used squads of 6 to represent Leman Russ crews. Since vehicles use hull points in 8th edition 40k, I set the casualties in the program based on how many hull points were lost in the game. For example, if the tank survived with less than half of its hull points, I input 2 casualties. If less than a quarter remained, 4 casualties. If the tank exploded upon being destroyed, I set the entire crew as "certainly dead."

8.2.4 Unit Transfers

Soldiers need not serve their entire career in a single squad. Have fun transferring them between units. Promote deserving soldiers from guardsmen units to command squads. You can also field the same squad as a different unit on the battlefield between games. For example, if the average number of battles survived in a guardsmen squad is 3 or more, I usually field that squad as a veteran squad on the tabletop.

8.2.5 Player Characters

Invite your friends to submit characters to sprinkle into your squads, then send them updates on how their characters are doing. This can even become a sort of low-key RPG if you let the players make career decisions for their characters. If they make it to squad leader, you could even give them control of that squad on the tabletop.

8.2.6 Single and Mixed Gender Regiments

By default, RandomGuardsmen.py generates roughly equal numbers of men and women in each squad. The ratio of men to women can be controlled by editing the "m2f" parameter at the start of the RandomGuardsmen.py file. This parameter represents the probability that a generated soldier is male. For an all-female unit, set it to 0. For an entirely male unit set it to 1.

8.2.7 Modifications

Though the Guardsmen Generator is intended for use with the Astra Militarum faction, it could be readily adapted to other factions and settings. Computer-savvy users can accomplish this by editing the names of ranks, skills, backgrounds, and other fields in the program files.