Athena Analyst's Guide

Athena S&RO Simulation, V3

February, 2012

William H. Duquette
Robert G. Chamberlain
Jet Propulsion Laboratory

Table of Contents

Models

1. INTRODUCTION

This document presents the models and related constructs implemented in version 3.1 of the Athena Stability & Recovery Operations (S&RO) Simulation. The models are described in sufficient detail to allow implementation; neither the derivations nor the implementation itself are in the scope of this document. Readers are advised to read this section and Section 2 (Athena Concepts) first, and then the other sections as desired. In addition, the model-related sections of the *Athena User's Guide* should be read before this document, as topics which are fully and adequately discussed there are not repeated here.

1.1 Overview

The Athena simulation is a decision support tool designed to allow a skilled[1] analyst to consider the intended and unintended consequences of various courses of action that might be taken during Stability & Recovery Operations. Athena is a descendant of the Joint Non-kinetic Effects Model (JNEM), but includes many new models. In addition, where JNEM is a federated simulation intended for use in training, Athena is a stand-alone single-user application intended for use in analysis.

The intent of Athena's models is first to capture and make explicit a wide variety of first-order causal links, each of which makes sense on the face of it, and then present the second-, third-, and n^{th}- order consequences of events while preserving the causal chain.

1.2 Other Documents

Documentation on using Athena may be found in the on-line help; invoke the Athena Simulation, and select Help Contents from the Help menu. Additional documentation may be found in the "docs" directory of the Athena build tree; open "docs/index.html" in a web browser, and follow the links.[2] The documentation is included with the installed software. Documentation can also be obtained directly from the Athena project; contact `William.H.Duquette@jpl.nasa.gov`.

Documentation in the "docs" directory include:

Athena User's Guide

This document combines a high-level introduction to the Athena models and philosophy with information about using the Athena application and hints on how to assemble Athena scenarios.

---

[1] I.e., knowledgeable about both the problem domain and Athena's models.

[2] When installed on the Windows operating system, this documentation is also available from the Windows start menu.

Athena Rules

This document describes the events and situations (drivers) that affect group attitudes, and the Driver Assessment Model (DAM) rule sets that assess attitude change.

Mars Analyst's Guide

Athena is built upon a software infrastructure layer called Mars. Models implemented in Mars, including the Generalized Regional Attitude Model (GRAM), are documented in the *Mars Analyst's Guide* (MAG), which may be found in the Athena documentation tree.

1.3 Changes for Athena 3.1

- Added actors and their strategies (goals, tactics, and attached conditions)
- Added the notion of *belief systems*; comparison of belief systems is the basis for the model of inter-group and group/actor relationships.
- Added a model of actor support, influence, and control of neighborhoods.
- Added a model of Essential Non-Infrastructure Services, which uses a new paradigm for driving attitude change.

Previous versions of Athena relied heavily on the analyst to make changes and provide inputs as simulation time advanced; essentially, the analyst was a stand-in for the federation of simulations that provided a stream of input events for JNEM. With the actor/strategy model in this version of Athena, it is now possible for Athena to run for months of simulated time with no external events. Rather, the stream of events is provided by the actors executing their strategies.

2. ATHENA CONCEPTS

Athena's models describe, within a region of interest, the significant political, social and economic actors, the actions they take, the effects of those actions on the civilian population of the region, the corresponding changes in political support for the actors, and the resulting effects on the stability and control of the region. Within this feedback loop Athena tracks actor assets, military force levels and activities, civilian attitudes, regional demographics, and the regional economy. Information flows between actors and civilians are modeled implicitly.

This section gives an overview of Athena and its parts, and of the basic concepts that Athena uses. The discussion is kept to a high level; see Sections 3 and following for the detailed models.

2.1 Model Parameters

Many of the Athena models contain numeric parameters that are used to calibrate and tune the results of the models. Athena stores these values as *model parameters* in the *model parameter database*. Although not part of an Athena scenario proper, the model parameters are accessible to the Athena user, who can view them and set them on a scenario-by-scenario basis. When such a parameter appears in a model description, below, a footnote will give its name in the model parameter database. See the *Athena User's Guide* for more information.

2.2 Simulated Time

Athena uses the following measures of simulated time.

Athena's clock measures time in integer *days* since time 0. The day is the smallest time interval with which Athena is concerned; simulation time always advances day-by-day. Days are also sometimes referred to as time *ticks*.

Time 0 is mapped to a calendar date by a *start date* set by the user. Athena then outputs simulated time as either some number of integer days or as a zulu time string based on the start date.

Athena is a time-step simulation; many computations take place at each time tick. Some models are triggered every so many ticks; these trigger points are called *tocks*. For example, economics and attrition are assessed every seven days (by default); thus, we say that the economics tock and the attrition tock are each one week.

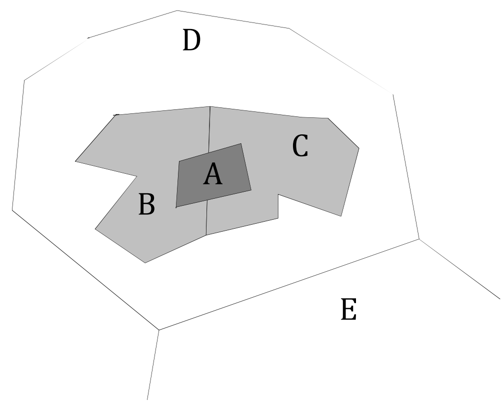
2.3 Geography

Each Athena scenario is associated with a geographical region called the Athena *playbox*, which is divided into polygonal areas called *neighborhoods*. (Indeed, the playbox is simply the collection of neighborhoods.)

2.3.1 Neighborhoods

Neighborhoods are simply a way of dividing the playbox into a number of reasonably homogeneous areas, and may be of any size: country, province, city, town, zip code, or neighborhood proper. Almost everything that happens in Athena happens in the context of a particular neighborhood.

Geographically, neighborhoods are defined as polygons on a map, using some appropriate coordinate system.[3] Neighborhoods may stack, e.g., a city may be a neighborhood within a larger province, and the city may contain several neighborhoods. When stacked, two neighborhoods are said to *nest* if the first is completely contained within the second, and to simply *overlap* if one is stacked on another but is not properly nested. In the diagram below, for example, A is an urban area surrounded by suburban areas B and C; all three lie within D, a county, which abuts E, another county. B and C are nested in D, and A overlaps B and C.



Note that locations within an inner neighborhood are not also part of the outer neighborhood; in the diagram above, D effectively has a hole cut out of it by B and C, and A cuts sections out of both B and C. Consequently, if a neighborhood is completely tiled by nested neighborhoods, it can be omitted as it contains no locations. If D's surface were entirely covered by suburbs, for example, there would be nothing left of D and no reason to define D within Athena.

Overlapping neighborhoods should be avoided, as they cause visualization problems—when a neighborhood is drawn transparently so that a map is visible underneath, the borders of any neighborhoods it overlaps are also visible, and this can be confusing. In the diagram, then, the borders of B and C should go around A, rather than A being overlaid on top of them.

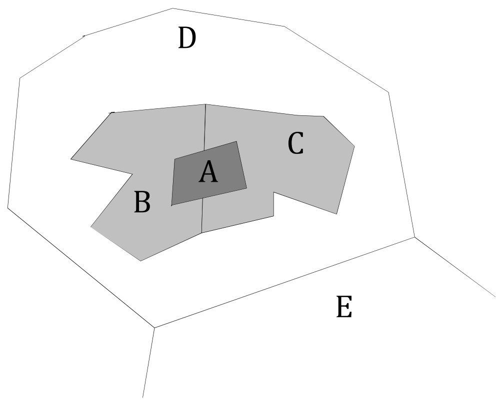
In practice, however, stacked neighborhoods of either kind are rarely used.

---

[3] Athena uses neighborhoods as homogeneous bins; map coordinates and distances are of interest only for visualization and have no effect on the model results. This may change in future versions.

2.3.2 Neighborhood Proximity

In Athena, simulation events take place within neighborhoods, and affect the people in the neighborhoods. An event taking place within a neighborhood can have ripple effects in other neighborhoods; the geographic spread of these ripples depends on how nearby other neighborhoods are presumed to be, socially and psychologically rather than geographically. This psycho-social nearness of one neighborhood to another is called *neighborhood proximity*.



There are four proximity levels: *here, near, far,* and *remote*. In the above diagram, for example, suppose that A is the capital city. From A's point of view, A is *here*, B and C are probably *near* A, and outlying area D is *far* from A. Neighborhood E is *remote*. In other words, a person in A would be concerned by violent events in A, might worry that violent events in B or C could spread to A, might be mildly concerned with riots out in the boondocks of D, and might not think at all of events in E.

Proximity need not be symmetric. A sees D as *far*, but D might see A as *near* because of its prominence in the country as a whole.

2.3.3 Local vs. Non-Local Neighborhoods

The playbox usually consists of a single contiguous region, e.g., a country, part of a country, or a set of adjacent countries. It is possible, however, to define *non-local* neighborhoods, representing countries (or simply staging areas) in other parts of the world. For example, the United States could be represented as a neighborhood, both as a place to stash US troops and as a way to track US civilian attitudes with regard to actions elsewhere in the world.[4]

However, such neighborhoods of convenience are outside the economy of the region of interest, also called the *local region*. Thus, neighborhoods are flagged as being *local* or *non-*

[4] Note that Athena does not directly model the effect of foreign actions on US civilian attitudes, but "magic" attitude inputs could be used for this purpose.

local, based on whether or not they participate directly in the economy of the local region. If the region of interest is Pakistan, for example, we might include portions of India and Afghanistan in the playbox but exclude them from the local economy.

2.4 Actors

An *actor* is an individual or group of individuals that functions as a significant decision-maker in the politics, economy, and society in the playbox. The relevant actors in a scenario depend on the scope of the scenario. If the action takes place in a single city, the mayor of the city (or those he represents) is likely to be an actor. If the playbox consists of an entire country, individual city mayors might not matter.

Actors can be of different levels of abstraction in the same scenario. If the United States is intervening in the playbox, the "US" will be an actor, even if the playbox represents a portion of a province and most of the other actors are specific individuals. And if the US leads a coalition, the coalition might be represented as a single actor or as a number of actors, depending on what is needed for the scenario.

2.4.1 Strategies: Goals, Tactics, and Conditions

Every actor can have zero or more *goals*. A goal is either a state of affairs that the actor wishes to bring about, or a state of affairs that the actor wishes to maintain. For example, the actor might wish to take control of a particular neighborhood. A goal consists of a set of Boolean predicates called *conditions*; the goal is said to be *met* if all of the conditions are true, and *unmet* otherwise. (Conditions are usually said to be true or false, but are also sometimes said to be met if true and unmet if false.)

The actor's strategy for reaching his goals is specified as a prioritized list of *tactics*. Each tactic represents an action the actor can take. Tactics require the use of assets (money and personnel), and as these are scarce the actor cannot do everything he might want to do.

The actor might have any number of *conditions* attached to a given tactic. The actor will consider executing a tactic only if all of the attached conditions are true. For example, tactics will often have an attached condition specifying the goals that the tactic supports.

2.4.2 Support, Influence, and Control

In general, most of the actors' goals will ultimately involve who is in control of the neighborhoods in the playbox. Note that we are speaking of power on the ground, not of governments or elections. The real power in Pakistan, for example, is not the Pakistani government as such, but the Pakistani military. In some provinces, control lies with the tribes who live there, and the central government is largely ignored.

To this end, we say that actors receive *support* from people in neighborhoods. Actors can use this support for themselves, or lend it to other actors. Actors with enough support in a neighborhood, whether direct or indirect, are said to have *influence* in that neighborhood. When an actor gains sufficient influence, he is said to be in *control* of the neighborhood. These concepts will all be made precise in Section 4.

2.4.3 Stability

Given that we know who is in control in a neighborhood, we may also wish to know whether the neighborhood is *stable*. When the United States pulls out of a country where it has intervened, we would like to ensure that the regime we leave behind is both friendly and stable.

As yet, we do not have a model for computing stability; this is a topic for future work.

2.5 Groups

The people in the playbox are divided into *groups*, of which there are three kinds: civilian groups, force groups, and organization groups.

2.5.1 Civilian Groups

Civilian groups represent the population of the playbox, i.e., the people who actually live in the neighborhoods. This population may be broken into groups by ethnicity, religion, language, social class, political affiliation, or any other demographic criteria the analyst deems necessary. Civilian groups are similar to the "market segments" used to target advertising: a group is a collection of people who may be assumed to have similar biases, interests, and behaviors due to their demographic similarity.

Each civilian group resides in some neighborhood, and each neighborhood must have at least one resident group.

Each civilian group's population is represented implicitly in the Demographics model (Section 10), broken down in various ways: the total population, the number of consumers, the number of workers, the subsistence agriculture population, and so forth. In addition, all civilians are also represented in the Ground model in the form of civilian units. People present in units are usually referred to as *personnel*, rather than as *population.* There is no real difference between the two, however; units are simply an aid to visualization of where people are and what they are doing.[5]

Athena models civilian groups in detail, tracking the attitudes of each group as the group's members are affected by a variety of events and situations.

---

[5] This is a significant difference from JNEM, where *personnel* and *population* are quite distinct topics.

Civilian groups are sometimes referred to as *neighborhood groups* because they reside in neighborhoods.

2.5.2 Force Groups

Force groups represent military forces, such as the U.S. Army, and other groups whose purpose is to apply force in support of policy. There are five kinds of force group; the force group type affects the degree to which a force group's units are able to project force vs. other force groups. The types are as follows:

- Regular military, e.g., the U.S. Army
- Paramilitary, e.g., SWAT teams and other combat-trained police units
- Police, e.g., normal civilian police
- Irregular military, e.g., militias
- Criminal, e.g., organized crime
-

A force group may be *uniformed* or *non-uniformed*; this affects the tactics the force group may use. In an S&RO situation, it is assumed that uniformed and non-uniformed forces usually use asymmetric tactics: the uniformed forces (typically regulars, paramilitary, or police) are hunting for cells of non-uniformed forces (typically irregulars), and the non-uniformed forces are attempting to whittle down the uniformed forces by means of IEDs and hit-and-run attacks.

2.5.2.1 Mobilization, Deployment, and Assignment

Every force group is owned by an actor; when that actor executes his strategy each week, he may mobilize, demobilize, deploy, and assign the forces under his command.

Force group personnel moved into the playbox are said to be *mobilized*; when removed from the playbox, they are said to be *demobilized.* Once in the playbox, all personnel must either be *deployed* to a particular neighborhood or demobilized. Once in a neighborhood, personnel may be *assigned* to do particular activities during the course of the week.

2.5.3 Organization Groups

Organization groups represent organizations that are present in the playbox to help the civilians. There are three kinds: Non-Governmental Organizations (NGOs), International or Inter-Governmental Organizations (IGOs), and Contractors (CTRs). NGOs are groups like the Red Cross or Doctors Without Borders who do humanitarian relief, development, and so forth. IGOs are international organizations like UNESCO. Contractors are commercial firms who are doing development work in the playbox at the behest of some actor.

Like force groups, every organization group must belong to some actor; this may be a real actor, or a fictive actor that exists only to own and direct one or more organization groups. Organization personnel are mobilized, deployed, and assigned using the same mechanisms as force groups, but can only be assigned a limited set of activities.

2.5.4 Force, Security, and Volatility

Civilian and force groups, and to a much lesser extent organization groups, have the ability and willingness to project and use force. Athena analyzes the balance of forces in each neighborhood, taking into account the populations and personnel present in the neighborhood, the types of each, and the ability of each group and unit type to project force. As the result of this analysis Athena computes the *security* of each group in each neighborhood. Closely related to security is *volatility*, a measure of the likelihood of random violence within the neighborhood.

A force or organization group's security in a neighborhood will determine which activities it can perform, if any.

2.6 Modeling Areas

Athena's models are loosely grouped into a handful of modeling areas. The models themselves will be described in detail in the body of this document; this section gives a brief description of each area and the models within it.

2.6.1 Ground

The most basic area is the Ground area. It includes the neighborhoods, groups, and units, as described above, and also the following specific models:

- Force, security, and volatility
- Group activity analysis, including activity coverage and the resulting Activity Situations (actsits).
- The Athena Attrition Model (AAM)
- Environmental Situations (ensits)
- Services

2.6.2 Demographics

The Demographics area tracks the civilian population of the playbox by group and neighborhood, and breaks down each group's population in a variety of ways. Populations change as civilian lives are lost due to collateral damage and direct attrition, and as civilians are displaced to other neighborhoods. The Demographics model provides population statistics, e.g., the number of consumers and the size of the labor force, to the Economics model, and creates demographic situations (demsits) in response to the rest of Athena.[6]

[6] In Athena 3.1 there is only one demsit type, Unemployment; see the *Athena Rules* document for details.

2.6.3 Attitudes

The Attitudes area is responsible for tracking the effects of events and situations (collectively known as *drivers*) on the attitudes of civilian groups. Attitudes currently include satisfaction of a group with respect to a variety of concerns, and cooperation of civilian groups with force groups.

The engine responsible for tracking attitudes and changes to attitudes is called the Generalized Regional Attitude Model (GRAM); it is documented in great detail in the *Mars Analyst's Guide*, which is delivered with Athena.

The Driver Assessment Model (DAM) is responsible for assessing the effects of each driver of attitude change, and giving related inputs to GRAM. DAM primarily consists of a large collection of rule sets; each rule set is devoted to one particular kind of driver, e.g., civilian casualties or presence of a force group. The DAM rule sets are described in the *Athena Rules* document. A wide variety of drivers already exist: activity situations, environmental situations, demographic situations, and civilian and organization casualties; in addition, the user may create their own *Magic Attitude Drivers* (MADs), in effect writing DAM rules on the fly.

2.6.4 Politics

The Politics area covers actors and their strategies and the computation of actor support, influence, and control.

2.6.5 Economics

The Economics area tracks employment and the production of goods and services in the local region. The economy changes in response to changes in neighborhood demographics and production capacity, and (via the Demographics model) drives the new Unemployment situation. In the future, it will also track the Black Market and actor incomes and expenditures.

2.6.6 Information

The Information area covers information flow to and among the civilian population, including propaganda and other information operations. At present, information is handled implicitly, primarily in the spread of indirect satisfaction and cooperation effects in GRAM, and in the use of group cooperation in the Attrition model and similar places.

2.7 Simulation States and the Advancement of Time

The Athena simulation has a number of states. When a new scenario is created, Athena is in the Scenario Preparation or **PREP** state. In this state, the analyst creates neighborhoods, groups, and so forth, and sets up their initial attributes.

When the analyst is satisfied with the scenario, the scenario is *locked*. At this time, Athena:

- Initializes GRAM.
- Computes demographics and force security.
- Calibrates the economics model.
- Executes actors’ strategies.
- Executes any simulation orders scheduled to be executed at time 0.
- Enters the **PAUSED** state.
-

In the **PAUSED** state the analyst may modify actor’s strategies, and execute and schedule many other simulation orders. Then, the analyst may ask the model to run time forward by some number of days. The simulation state changes to **RUNNING**, and time advances day by day.

During each day, Athena performs the following steps:

- Updates the demographics and activity statistics.
- Assesses attrition and the economy (if it is the correct tock)
- Assesses environment, activity, and demographic situations.
- Advances GRAM.
- Saves a variety of historical data, for later plotting.
- Updates the simulation time by one day.
- Executes actor’s strategies (if it is the correct tock)
- Executes any scheduled events:
  - Spawning or resolving of environment situations
  - Scheduled orders
- Returns to the **PAUSED** state if the stopping time has been reached, allowing the analyst to make changes.

It is useful to think of these various computations as being performed at different times of day:

- Midnight: the simulation time is advanced by one day.
- Morning: strategies are executed, activities are staffed, and scheduled orders are executed.
- Noon: if PAUSED, the analyst can make changes interactively.
- Evening: the activities and events of the day are assessed for their attitude implications, and GRAM is advanced.
- Midnight: the simulation time is advanced by one day.

In short, the simulation pauses in the middle of the day. As a result, if a situation begins at time t the user will first see it at time $t + 1$.

3. STRATEGIES AND STRATEGY EXECUTION

Section 2.4 gives a brief overview of actors and their strategies: goals, tactics, and conditions. This section goes into more detail about each of these.

The specific tactics and conditions supported by Athena are discussed in detail in the *Athena User's Guide*.

3.1 Agents

An *agent* is an entity that can have a strategy in Athena. At present there are two kinds of agent: actors, as described in Section 2.4 and below, and the "SYSTEM" agent. In the future, civilian groups might also be agents.

3.1.1 The SYSTEM Agent

The SYSTEM agent represents the Athena simulation as a whole, and allows Athena to use the strategy mechanism to execute actions which are conditioned on the state of the simulation but which are not associated with any particular actor. By assigning tactics to the SYSTEM agent, the analyst can cause a number of effects, including stopping the simulation when particular conditions are met, and displacing civilians to other neighborhoods. However, the SYSTEM agent is not an actor; it has no assets and plays no role in the politics of the region.

3.2 Actors

An actor is an individual or group of individuals that functions as a significant decision maker within the playbox. Actors control assets, and can use them to take action. Actors can control neighborhoods, and lend money and political support to other actors. An actor's actions are determined by the actor's strategy, which consists of goals, tactics and conditions.

3.2.1 Assets: Cash

Each actor has the following attributes:

Name	**Description**
income	Income in dollars per week.
cash-on-hand	Cash on hand, in dollars: the money the actor has immediately available to fund tactics. Unspent cash is carried over to the next week.
cash-reserve	Cash reserve, in dollars. The actor can reserve funds for later use.

At time 0, each actor starts with a particular amount of cash on hand and in reserve, and a starting income. Each week the actor receives his income, and possibly also funding from other actors; this money appears in his *cash-on-hand*. As part of his strategy, the actor may choose to spend his *cash-on-hand*, transfer a portion of it to or from his *cash-reserve*, or simply let it ride until the next week.

Money is spent by executing tactics; the cost of executing a tactic depends on the tactic type, and on the parameters of the specific tactic. For example, deploying troops to a neighborhood for a week incurs a weekly maintenance cost per person deployed.

3.2.2 Assets: Personnel

Each actor can own force groups and organization groups, and then mobilize, deploy, and assign personnel belonging to these groups (Section 2.5.2.1) by executing the relevant tactics. In addition, the presence of personnel belonging to an actor's groups in a neighborhood contributes to the actor's support in that neighborhood.

3.3 Conditions

A condition is a Boolean predicate about some aspect of Athena and its models which can be attached to a goal or tactic to control strategy execution. Athena defines a variety of types of condition; for example,

- Does actor *a* control neighborhood *n*?
- Is actor *a*'s cash-reserve greater than $1,000,000?
- Is group *g*'s mood less than -40.0?

A condition can be evaluated, returning a true or false value. The condition's truth value depends on the condition type, the condition's parameters, and on the current state of the simulation.

Each condition has a *state*, one of "normal", "disabled", and "invalid". Disabled and invalid conditions have no value and are ignored by Athena during strategy execution.

The set of potential condition types is vast, and is expected to grow over time. Those currently implemented in Athena are documented in detail in the *Athena User's Guide.*

3.4 Goals

Actors have goals they would like to achieve. In Athena, a goal belongs to an actor and is defined by a set of one or more conditions representing some state of affairs in the simulation; the goal is *met* if all of the attached conditions are true, and is *unmet* otherwise.

A goal can represent either a state of affairs that the actor would like to bring about, or a state of affairs that exists and that the actor would like to preserve. In the former case, the actor will execute tactics when the goal is unmet; in the latter case, when the goal is met.

Each goal has a *state*, either "normal" or "disabled". Disabled goals are neither met nor unmet, and are ignored during strategy execution.

3.5 Tactics

A tactic is an action that the actor can choose to take, possibly in support of one or more goals. For example, an actor can deploy troops, assign group activities, set rules of engagement, fund essential services, and support other actors.

Athena defines a number of kinds of tactic; these are referred to as *tactic types*. The complete set is described in detail in the *Athena User's Guide*.

A tactic may have a cost in dollars, personnel, or both, depending on the tactic type and its parameters. Dollars spent on a tactic are consumed. Personnel used by a tactic are unavailable for use by other tactics during the same week, but may be reassigned the following week (unless killed or demobilized). If the required assets are not available, the tactic cannot be executed.

Each tactic has a *state*, one of "normal", "disabled", or "invalid". Tactics that are disabled or invalid are ignored when strategies are executed.

Conditions may be attached to tactics; a tactic will be considered for execution only if all of the attached conditions are met. A tactic can therefore be executed in support of a goal or goals by attaching a condition that is true when the goal or goals is met or unmet.

Note that it is quite possible for an actor to take actions that are counter-productive to his goals. Athena itself has no idea which tactics (if any) will bring about any particular goal or set of goals, nor does it attempt to determine optimal strategies for a set of goals. Rather, it is aimed at modeling the decision makers in the region, along with their limitations and prejudices.

3.6 Strategy Execution

The actor's prioritized list of tactics represents his strategy for achieving his goals. At each strategy tock, (nominally one week)[7], the following algorithm is used:

[7] Perhaps this should be one day; and tactics should have a length in days. Tactics persist until their length has been exceeded. Some things happen quickly and others slowly, and some things have great momentum. We will eventually need a way to handle that.

Load working data (see below)
Evaluate all goals.
For each agent:
    For each of the agent's tactics in priority order:
        If the tactic's dependent conditions are met:
            Attempt to execute the tactic:
                If insufficient resources are available:
                    Skip this tactic.
                Otherwise:
                    Expend the resources.
                Execute the tactic.
Save working data.

This process is called strategy execution. All agents execute their strategies at once, at the beginning of the week; the executed tactics actually play out over the course of the week.

3.6.1 Working Data

As actor A executes his strategy, he allocates assets to different uses. As the process goes on, Athena must track how much A has spent, and how much he has left. This presents an issue, because other actors have some insight (via conditions) into A's resources—the number of troops that A has in the playbox for example, and what they have been doing. Consequently, Athena has to manage what each actor knows, and when he knows it:

- Every actor knows the state of the simulation at the **start** of strategy execution.
- Every actor knows the decisions he has made **during** strategy execution, and what resources he has left.
- No actor knows anything about any other actor's decisions until **after** strategy execution is complete.

In other words, bright and early the first morning of the week, all of the actors go to their offices, look at the reports from the past week, determine their orders for the following week, and promptly at 8 AM (so to speak) they send them out to their subordinates and go back to watching soap operas for seven days.

Therefore, at the beginning of strategy execution, Athena loads each actor's "working data"—the actor's view of those things he can change by his tactics. The conditions are designed, where appropriate, to query the actor's working data when they apply to the actor himself, and the state of the simulation when they apply to other actors. At the end of strategy execution, the working data is saved, changing the state of the simulation. Thus, each actor knows and can see his own decisions, but not those of his fellows.

3.7 Roads Not Taken

This section records ideas we chose not to implement, with our reasons for not doing so. Some of them we may choose to reconsider in the future, so it seems worthwhile recording them.

3.7.1 Goal Extensions

Our original notion of goals was that a goal was like a condition, embodying a Boolean expression, but with additional semantics. We would have a variety of goal types. Over time, though, it became clear that we wanted condition types for all of the same Boolean expressions as we had goals, and we redefined goals as a collection of conditions and discarded the notion of goal types altogether.

The following are some extensions to the original notion of goal.

Progress Metrics: A progress metric is a measure of the actor's progress toward a goal, probably expressed as a percentage. This would be nice to have, but is hard to do in practice, likely requiring significant modeling for each goal type. For example, considering the goal of controlling a neighborhood: what is a suitable metric for progress? One could relate the actor's influence in the neighborhood to that of other actors, which is in some sense a distance measure…but it doesn't give much insight into the degree of action required by the actor to gain control, or how much work remains to be done. It might take a great deal of slogging (providing services, information operations, and so forth) to gain that last few percent of influence; or a great deal of influence might be gained by one lightning stroke. In short, naïve models are likely to be unsatisfactory, and informed models might be unsatisfactory as well.

Activation Conditions: As defined above, the actor pursues goals that are both enabled and unmet, and the analyst controls the goals' *state* flags. It might be desirable to allow activation conditions to be attached to goals just as they are to tactics; the goal would be enabled only if all activation conditions were met. For example, Goal G1 might be enabled if:

- Goal G2 has been met.
- Goal G3 has not been met.
- The simulation time is less than six months after time 0.
-

This would allow the actor to pursue chains of goals, and to decide to pursue one goal in preference to another. Note, though, that allowing goals to depend on whether other goals are met or unmet could cause the condition evaluation network to contain cycles; we would need to detect and prevent these.

Actions on Success/Failure: A goal succeeds when its condition is met; Ed Upchurch has suggested that a goal fails if an end-time is reached before the goal succeeds. Certain actions could be taken on goal success or failure, such as activating another goal. Note that some of the actions Ed suggests would also be handled by activation conditions, as defined above.

3.7.2 Goal Prioritization

As described above, the analyst sets the priority of each of the actor's tactics; the tactics are then executed in priority order, as resources and conditions permit. Originally we considered a different approach:

- Actor a's goals are prioritized from most important to least important.
- Every tactic is associated explicitly with one or more goal.
- The tactics associated with a goal are in priority order for that goal.

Then, the algorithm is as follows:

For each actor *a:*
    Let the *plan* be the empty list.
    For each of a's goals G, in priority order,
        For each tactic T associated with G, in priority order:
            If T is already in the *plan*, skip it.
            If T's conditions are not met, skip it.
            If T's cost exceeds the available resources, skip it.
            Reduce a's resources by T's cost.
            Add T to a's *plan*.
    For each tactic T in a's *plan*,
        Execute tactic T.

This is a very natural approach, where you start with goals and then move on to the tactics required to achieve them. Unfortunately, it yields unrealistic results because it assumes that each goal in the list dominates all subsequent goals: the actor will do everything possible to meet the first goal before moving onto the second, and so on. In reality, an actor might wish to push forward towards several goals at once, using the most important tactic for each, before moving on to less important (or less effective) tactics.

By prioritizing the tactics instead of the goals, and attaching "Goal Unmet" conditions to them, we allow the analyst to decide which tactics the actor is likely to use in support of his goals, and to order them to reflect the actor's likely very complex priorities—instead of trying to write a model to do that, and getting it wrong.

4. RELATIONSHIPS AND CONTROL

Athena models the relationships between groups and between groups and actors. This section describes how relationships are defined and how they vary over time. In addition, it describes the closely related issue of actor support and influence in neighborhoods, and how influence leads to neighborhood control. More particular, this section addresses the following topics:

- Inter-group relationships, also called "horizontal" relationships.
- Group/actor relationships, also called "vertical" relationships.
- How vertical relationships vary with time.
- Support for actors by groups in neighborhoods.
- Support for actors by other actors.
- The influence of actors in neighborhoods due to the support they receive.
- The circumstances under which control of a neighborhood shifts from one actor to another (or to or from no actor at all).
- What happens when control of a neighborhood shifts.

4.1 Relationships and Affinity

Athena defines two kinds of relationship, horizontal relationships and vertical relationships. In each case, a relationship is expressed as a number X where $-1.0 \leq X \leq +1.0$. The magnitude of the number indicates the strength of the relationship and the sign indicates whether the related parties are friends or foes. We usually assume that an entity's relationship with itself is 1.0, and that a relationship of -1.0 is pathological; the practical range is more like $-0.6 \leq X \leq +1.0$, though this is not a hard and fast rule.

In Athena 3, all relationships are based on an affinity A computed by comparing the belief systems of the entities involved, where $-1.0 \leq A \leq +1.0$. The definition of belief systems and affinities is found in the *Mars Analyst's Guide*. An affinity greater than zero implies that the two entities tend to have compatible beliefs; an affinity less than zero implies that the two entities tend to have incompatible beliefs. Both actors and civilian groups have belief systems, and all pairs of them have affinities. (Force and organization groups inherit the affinities of their owning actors.)

Athena uses *quality* data types to relate numeric values to symbolic constants. The `qaffinity` data type is used with both affinity and relationship values; it is defined as follows:

Narrative	Symbol	Value	Range
a supports *b*	SUPPORT	0.8	+0.7 < *value* ≤ +1.0
a likes *b*	LIKE	0.4	+0.2 < *value* ≤ +0.7
a is indifferent to *b*	INDIFF	0.0	–0.2 < *value* ≤ +0.2
a dislikes *b*	DISLIKE	–0.4	–0.7 < *value* ≤ –0.2

a opposes b	OPPOSE	−0.8	−1.0 ≤ *value* ≤ −0.7

4.2 Horizontal Relationships

The horizontal relationship between group f and group g is denoted R_{fg}. This is the familiar group relationship from earlier versions of Athena and from JNEM. As in earlier versions, it is a static, unchanging value that reflects the amity or enmity between the pair of groups. It is not symmetric; it will often be the case that $R_{fg} \neq R_{gf}$. Athena 3 bases the relationship on group affinity as described in this section.

4.2.1 Force and Organization Group Affinities

Force and organization groups have neither belief systems nor affinities. They are, however, owned by actors, which do. For the purposes of computing relationships, we assume that force and organization groups inherit the belief systems and affinities of their owning actors.

To put it another way, each group gets its affinities from its associated *belief system entity*, where:

- The belief system entity for each force and organization group is the actor that owns the group.
- The belief system entity for each civilian group is the group itself.
-

Thus, when we speak of the affinity A_{fg} of group f for group g, we are really speaking of the affinity of group f's belief system entity with group g's belief system entity.

4.2.2 Computing Horizontal Relationships

The horizontal relationship between two groups is then defined as follows:

$$R_{fg} = \begin{cases} 1.0 & \text{where } f = g \\ A_{fg} & \text{otherwise} \end{cases}$$

Basing relationships on affinity provides a straightforward way to allow relationships to vary dynamically: if a group's belief about a topic changes (perhaps due to information operations), its relationships will change. The GRAM model, however, does not support dynamic group relationships at this time, so this remains for future work.

4.3 Vertical Relationships

The vertical relationship V_{ga} is a number that determines the extent of group g's opposition to or support for actor a. As with R_{fg}, $-1.0 \leq V_{ga} \leq +1.0$; unlike R_{fg}, the vertical relationship is unidirectional. We are measuring g's opposition to or support for a; a's opposition to or support for g will be revealed by a's choice of tactics.

The foundation for V_{ga} is the affinity A_{ga} of g for a based on their belief systems, where force and organization groups inherit their belief systems and affinities from the owning actors, as stated above. V_{ga} is static when g is a force or organization group; for civilian groups, V_{ga} is dynamic, varying over time according to a's actions and g's circumstances. Groups expect things from actors, and the relationship will depend on how well the actors meet those expectations.

This section explains how $V_{ga}(t)$ is computed as the simulation advances.

4.3.1 Force and Organization Groups

Force and organization groups are owned by actors, and share the relationships of their actors. When g is a force or organization group owned by actor b, we define $V_{ga}(t)$ as follows:

$$V_{ga}(t) = \begin{cases} 1.0 & \text{when } a = b \\ A_{ba} & \text{otherwise} \end{cases}$$

That is, a force or organization group has a perfect relationship with its owner, and its vertical relationship with any other actor is simply the affinity of its owner for that actor. Because belief systems are static in this version, $V_{ga}(t)$ is constant when g is a force or organization group.

4.3.2 V_{ga} at Time t

The vertical relationship of civilian group g with actor a is a dynamic quantity that varies according to a's actions and g's circumstances. The relationship at time t is denoted $V_{ga}(t)$, and is given by the following equation:

$$V_{ga}(t) = BV_{ga}(t_{control}) + \Delta V_{mood}(t) + \Delta V_{eni}(t) + \Delta V_{tactics}(t) + \Delta V_{beliefs}(t)$$

where

$t_{control}$	=	The time at which control of group g's neighborhood last changed. At simulation start, $t_{control} = 0$.
$BV_{ga}(t_{control})$	=	The base vertical relationship as of $t_{control}$.
$\Delta V_{mood}(t)$	=	The change in V due to changes in group g's mood
$\Delta V_{eni}(t)$	=	The change in V due to a's provision of Essential Non-Infrastructure (ENI) services (Section 8)
$\Delta V_{tactics}(t)$	=	The change in V due to a's choice of tactics
$\Delta V_{beliefs}(t)$	=	The change in V due to changes in a's and g's belief systems

Note that the symbols listed above depend implicitly on g and a.

In other words, at any given point in time the vertical relationship between civilian group g and actor a is some base value plus a number of deltas; the deltas are determined by rule sets, which are described below.

At any given time, a neighborhood is either uncontrolled or controlled by a single actor. As described below, control of a neighborhood can shift; for example, actor a can take over from actor b. When control shifts in neighborhood n, this establishes a new baseline for the vertical relationships of the civilian groups residing in n. The time that control last shifted is denoted $t_{control}$, and the new baseline is denoted $BV_{ga}(t_{control})$.

The following sections will describe each component of this equation.

4.3.3 Magnitudes and the Scale Function

Several of the rules in the following sections make use of the following magnitude symbols, which are also used throughout the attitude rules:

XXXXL–	XXXL–	XXL–	XL–	L–	M–	S–	XS–	XXS–	XXXS–
-30.0	-20.0	-15.0	-10.0	-7.5	-5.0	-3.0	-2.0	-1.5	-1.0
XXXS+	XXS+	XS+	S+	M+	L+	XL+	XXL+	XXXL+	XXXXL+
1.0	1.5	2.0	3.0	5.0	7.5	10.0	15.0	20.0	30.0

Each magnitude represents a percentage movement from a current position to an extreme, e.g., M+ represents a 5% movement from the current value of a variable toward the maximum extreme, and L– represents a 7.5% movement from the current value of a variable toward the minimum extreme.

For vertical relationships, this scaling function is defined as follows:

$$\text{scale}(M,B) = \begin{cases} \left|\dfrac{M \times (1-B)}{100}\right|, \text{where } M \geq 0 \\ \left|\dfrac{M \times (1+B)}{100}\right|, \text{where } M < 0 \end{cases}$$

where M is the magnitude of the change and B is the base value of the relationship. This scaling will be presumed wherever the magnitude symbols are used.

4.3.4 The Base Vertical Relationship, BV_{ga}

The base vertical relationship, $BV_{ga}(t)$, is the value to which deltas are applied to compute $V_{ga}(t)$. The base vertical relationship of civilian group g with actor a at time 0, $BV_{ga}(0)$, is simply the affinity of the group for the actor:

$$BV_{ga}(0) = A_{ga}$$

The base vertical relationship is recomputed whenever control of g's neighborhood changes:

$$BV_{ga}(t_{control}) = V_{ga}(t^{-}_{control}) + \Delta V_{control}$$

where

$t^{-}_{control}$	= The time just prior to the change of control
$V_{ga}(t^{-}_{control})$	= The vertical relationship as of $t^{-}_{control}$
$\Delta V_{control}$	= Change in V due to change of control

The value of $\Delta V_{control}$ depends on the vertical relationship between g and a just prior to the change in control, i.e., $V_{ga}(t^{-}_{control})$, and on the change in a's status in the neighborhood, as shown in the following table:

$V_{ga}(t^{-}_{control})$	a is now in control but was not	a is still not in control	a was in control but now is not
SUPPORT	L+	M–	L–
LIKE	M+	S–	M–
INDIFF	0	0	0
DISLIKE	M–	0	XS–
OPPOSE	L–	0	S–

The symbols L+, M+, etc., have the usual interpretation, as described in Section 4.3.3. Thus, when $V_{ga}(t^{-}_{control})$ has the symbolic value "LIKE" and actor a has just gained control of group g's neighborhood, $\Delta V_{control}$ will be

$$\Delta V_{control} = \text{scale}(\text{M+}, V_{ga}(t^{-}_{control}))$$

4.3.5 Computing ΔV_{mood}

This term computes the delta in V_{ga} due to changes in the mood of group g since the last time control of g's neighborhood changed. The delta depends on whether or not actor a is blamed for the change, which depends on whether or not actor a is in control of g's neighborhood. The magnitude of the delta (subject to the usual scaling) is shown in the following table:

Group g's mood is:	Actor a is in control	Actor a is not in control
Much worse than at the last change of control: $S_g(t) - S_g(t_{control}) \leq -30$	XL–	M+
About the same as at the last change of control: $-30 < S_g(t) - S_g(t_{control}) < 30$	0	0
Much better than at the last change of control: $30 < S_g(t) - S_g(t_{control})$	XL+	M–

A more aggressive model could have the controlling actor's support continue to erode so long as the group's mood is depressed, e.g., the magnitude shown in the table could be viewed as a daily or weekly slope rather than as a temporary delta. This would require modeling vertical relationships rather like attitudes are modeled in GRAM.

Experience may show that this model is too simple; in the right circumstances, group g could very well blame some actor other than the controlling actor (or in addition to the controlling actor) if that actor were seen as the cause of their problems. Further modeling may be needed in future releases.

4.3.6 Computing ΔV_{eni}

This term computes the delta in V_{ga} due to actor a's provision of Essential Non-Infrastructure (ENI) services to group g. This rule set is described in Section 8.2.5.

4.3.7 Computing $\Delta V_{beliefs}$

Note: Beliefs do not change dynamically in this version of Athena; hence, this model will not be implemented until some future release.

This term computes the change in V_{ga} due to changes in the beliefs of group g, actor a, or both, since the last time control of g's neighborhood changed.

Given group g and actor a,

$$\Delta V_{beliefs}(t) = A_{ga}(t) - A_{ga}(t_{control})$$

In other words, just as $V_{ga}(t)$ begins with affinity, so it is directly affected by changes in affinity.

4.3.8 Computing $\Delta V_{tactics}$

Note: This term has not yet been modeled; it will not be implemented until a later version of Athena.

This term computes the delta in V_{ga} due to actor a's use of tactics that are inconsistent with group g's beliefs; the delta is likely to be larger if the tactics are also inconsistent with actor a's stated beliefs, as this can be seen as a betrayal of principle.

An actor's tactics reflect the actor's beliefs. However, an actor's belief system, as defined in Athena, contains those beliefs the actor wishes *to be seen* as having. Thus, one way to model this would be to change the actor's expressed beliefs if the actor uses a tactic that is incompatible with them. If so, we might not need an explicit model for $\Delta V_{tactics}$; the desired effect will simply be part of $\Delta V_{beliefs}$. The difficult part is likely to be the link between the actor's tactics and the actor's beliefs, given that belief system topics are defined as part of the scenario rather than being known *a priori*.

4.4 Actor Support and Influence

An actor receives *support* in a neighborhood from the groups in the neighborhood, and also possibly from other actors. Given enough support, the actor can gain *influence* in the neighborhood, and perhaps even control the neighborhood. This section defines the terms "support" and "influence" as they are used in Athena.

4.4.1 Direct vs. Derived Support

An actor's political influence in a neighborhood derives from the support he receives from the groups in the neighborhood. An actor may make use of this support himself, or he may choose to give his support to another actor. We will call the support an actor receives directly from groups his *direct support*, and support an actor receives from other actors his *derived support*.

4.4.2 Direct Support

Actor *a*'s direct support in *n* is determined by the number of people who favor him, the strength of their favor (as determined by the vertical relationship between them and the actor) and their ability to move and work within the neighborhood (as determined by their security). Thus,

$$DirectSupport_{na} = \sum_{\substack{g \in n \\ V_{ga} \geq V_{min} \\ SF_{ng} > 0}} V_{ga} \cdot SF_{ng} \cdot \frac{P_{ng}}{P_n}$$

where

$DirectSupport_{na}$	=	The fraction of *n*'s population that is willing and able to support *a*.
P_{ng}	=	The number of people belonging to *g* in *n*. Group *g* can be a civilian, force, or organization group.
P_n	=	The total number of people of all groups in *n*.
V_{ga}	=	The vertical relationship between group *g* and actor *a*.
V_{min}	=	The minimum relationship needed to qualify as "support", nominally 0.2.[8]
SF_{ng}	=	Group *g*'s security factor in neighborhood *n*, as computed from its security using a Z-curve.

The security factor is a number that indicates the fraction of a group that is able to actively support actors given the group's security level. The security of group *g* in neighborhood *n* is an

[8] Model parameter: `control.support.vrelMin`, nominally 0.2.

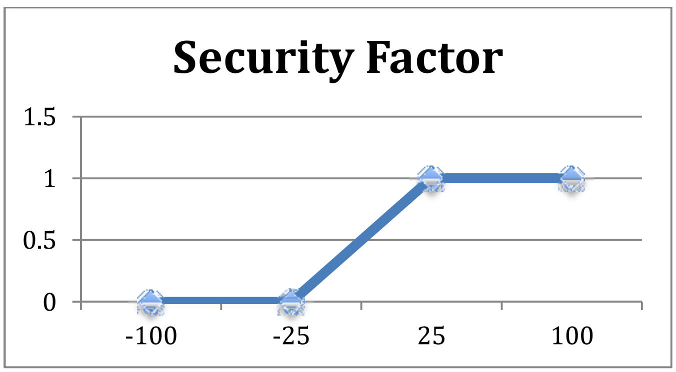
integer in the range $-100 \leq Security_{ng} \leq +100$; the number is interpreted according to the following scale:

Bin	Range
High	25 < *security* ≤ 100
Medium	5 < *security* ≤ 25
Low	-25 < *security* ≤ 5
None	-100 < *security* ≤ -25

We compute the security factor from the security level as follows:

$$SF_{ng} = Z_{security}(Security_{ng})$$

where $Z_{security}$ is a Z-curve:[9]



This Z-curve is based on the assumption that groups with a security of "None" (-25 or less) can't support anyone, and once a group has a security of "High" (25 or more) it is supporting as much as it can. Increasing security in between increases the group's support per person directly, as shown. The $Z_{security}$ parameters are defined in the model parameter database, and so can be changed to reflect different assumptions.

4.4.3 Derived Support

Actor *a* may do one of three things with his direct support in neighborhood *n*:

- He may use it himself, to try to take control in the neighborhood. In this case, we say that *a* supports himself in *n*.

[9] Model parameter: `control.support.Zsecurity`

- He may give it to another actor *b*. In this case, we say that *a* supports *b* in *n*, and that *b* receives derived support from *a*.

- He may choose not to use it at all, that is, he may be *apolitical*. For example, organization groups in Athena must be owned by an actor, but that actor will not usually involve himself in neighborhood politics. He might receive support from the groups in a neighborhood, but will not use that support to take control in the neighborhood. In this case, we say that *a* supports no one in *n*.

Note that derived support is not transitive. If *a* supports *b*, and *b* supports *c*, that does *not* mean that *a* supports *c*. Rather, it means that *b* gets *a*'s direct support, and *c* gets *b*'s direct support.

An actor chooses which actor to support in two ways. First, each actor has an actor that he "usually supports"; this may be **SELF**, **NONE**, or the name of a specific actor. This value is set as part of the definition of the actor during scenario preparation; it can be changed by the analyst as the simulation runs. In the absence of other input, the actor will support the given actor (or none) in all neighborhoods. Thus, for example, an actor that owns humanitarian organization groups like the Red Cross would support **NONE** in all neighborhoods, and would thus not be included in the political process.

Next, an actor can execute the **SUPPORT** tactic[10] to explicitly support another actor. The **SUPPORT** tactic has the following inputs:

- The actor to support
- A list of neighborhoods in which to support him.

The effect of the **SUPPORT** tactic lasts until the next strategy execution, when it can be repeated.

Thus, the political relationships between the actors depend entirely on the actor's strategies. This is a general rule—**an actor is what an actor does**, and hence **relationships between actors are always mediated by the actors' strategies**.

4.4.4 Total Support

Actor *a*'s total support in *n*, also called his *effective support* in *n*, is computed as follows:

$$Support_{na} = \sum_{b \text{ supports } a} DirectSupport_{nb}$$

[10] See the *Athena User's Guide* for specifics about tactic types.

Note that while $DirectSupport_{na}$ can be interpreted as the fraction of the population of n that supports a, $Support_{na}$ can not. It is simply a score used to compute influence.

Some observations:

- If every actor supports himself, then $Support_{na} = DirectSupport_{na}$ for all a.
- If no actor supports actor a, including himself, then $Support_{na} = 0$, even if $DirectSupport_{na} > 0$.
- If an actor receives derived support, then his $Support_{na}$ can be greater than 0, even if he gives his direct support away or chooses not to use it.
- If an actor gives his direct support away and receives no derived support, his $Support_{na}$ will be 0.

The third bullet is particularly interesting: an actor can be "in control" of a neighborhood because of support from some other powerful actor, whether he wants to be or not.

4.4.5 Influence

Actors have influence in a neighborhood in proportion to their support relative to other actors. In particular, if only actor A has support in neighborhood n, then only A will have influence in n. We require an actor to have a minimum of *SupportThreshold* support[11] to have any influence; this prevents an actor with a trivial (though positive) degree of support from taking control of a neighborhood in the absence of other candidates.

Let C_n be the set of actors a for which $Support_{na} > SupportThreshold$. Actor a's influence in neighborhood n is then

$$Influence_{na} = \begin{cases} \dfrac{Support_{na}}{\sum_{b \in C_n} Support_{nb}} & \text{when } a \in C_n \\ 0.0 & \text{otherwise} \end{cases}$$

Note that $0.0 \leq Influence_{na} \leq 1.0$ and that $\sum_{a \in C_n} Influence_{na} = 1.0$ unless C_n is the empty set, in which case all influence values in n are 0.0. Influence is thus a zero-sum game; an actor can increase his influence only at the expense of some other actor.

[11] Model parameter: `control.support.min`, nominally 0.1.

4.5 Control of a Neighborhood

When the residents of neighborhood *n* see actor *a* as being responsible for causing or fixing the current state of affairs in *n*, then *a* is said to be in control of neighborhood *n*. For *a* to be in control in *n* might involve an official governmental position, or it might simply be a *de facto* status, as when a military commander or warlord takes a neighborhood by force. As shown in Section 4.3, the civilians in *n* will hold *a* responsible for increases or decreases in their mood, and also for the provision of ENI services.

The actor in control of *n* will usually be the actor that has sufficient influence[12] to dominate *n*; however, this is not always the case. Suppose actor *A* has a dominating degree of influence, and is in control of *n*. *A'*s rival, actor *B*, moves troops into *n*, causing a crisis. *B'*s action decreases security levels in *n* for everyone but *B'*s own troops, thus eroding *A'*s influence significantly. Although *A* may no longer be able to dominate the neighborhood, the civilians will continue to hold *A* responsible until he either becomes irrelevant (because he is dominated by *B*) or *B* is able to dominate the neighborhood altogether. If *A* manages to drive *B'*s forces out of *n*, he can maintain control throughout the crisis, even though his actual degree of influence may have been very low at times.

Thus, we define the following rules. Suppose, first of all, that actor *a* is in control of neighborhood *n* before we re-assess each actor's influence in *n*. Who is in control after the re-assessment is determined by applying these rules:

1. If $Influence_{na} > ControlThreshold$, then a remains in control.[13]
2. If $Influence_{na} \leq ControlThreshold$ but $Influence_{na} \geq Influence_{nb}$ for all b, then a no longer dominates the neighborhood, but remains in control.
3. If $Influence_{nb} > Influence_{na}$ for some actor b:
    a. If $Influence_{nb} > ControlThreshold$, then actor b now dominates the neighborhood and is in control. **Control has shifted.**
    b. Otherwise, actor a has clearly lost control, being dominated by b, but no other actor dominates the neighborhood. No actor is in control of neighborhood n. **Control has shifted.**
4. If no actor is in control of neighborhood n, and some actor b gains influence such that $Influence_{nb} > ControlThreshold$, then naturally b is now in control. **Control has shifted.**

In the future we might want to take time into account: if actor *a* is in control but no longer dominates the neighborhood (i.e., *a* has influence less than the *ControlThreshold*), then after some period time he is no longer deemed to be in control even if no other actor has gained

---

[12] We will make this notion more precise below.
[13] Model Parameter: `control.threshold`, nominally 0.5.

control. Similarly, actor b might need to dominate the neighborhood for some period of time to gain control.

4.5.1 When Control Shifts

When control shifts:

- Support for actors is now relative to the new political situation. $BV_{ga}(t_{control})$ is computed, leading to new values for V_{ga}, $Support_{na}$, and $Influence_{na}$.

- The change in control is a monitored event that has an effect on the attitudes of every civilian group resident in the neighborhood.

The attitude effects are assessed by the CONTROL rule set, which is documented in the *Athena Rules* document.

5. FORCE ANALYSIS

Athena analyzes the forces present in each neighborhood along with the neighborhood demographics to determine the following quantities for each group and neighborhood:

Security: A group's level of security in a neighborhood, measured as a score from -100 (very dangerous) to +100 (very safe). A group's security determines the kinds of things the group can safely do within the neighborhood.

Volatility: The volatility of a neighborhood is the likelihood of spontaneous violence within the neighborhood, measured as a score from 0 (least volatile) to 100 (most volatile).

Force: A group's force in a neighborhood is its physical ability to use its assets to control that neighborhood through force. The major component of force is military strength.

5.1 Measuring Force

In general:

- The primary component of group g's force in neighborhood n is group g's assets in the neighborhood. In Athena, a group's assets are its people.

- Friendly military personnel in neighborhood n, i.e., those with a positive relationship with g, should increase g's force in proportion to their numbers.

- Friendly civilians in n should also increase g's force, but by a smaller amount per person than soldiers or militia. However, an aggressive demeanor and a negative mood ($S_g < 0$) can increase a civilian group's force multiplier.

- Friendly contractor personnel in n may also increase g's force, but personnel from friendly IGOs and NGOs are expected to remain neutral in case of conflict and do not contribute to the force of other groups.

- Friendly personnel in nearby neighborhoods should also increase g's force, but to a smaller extent than personnel in neighborhood n.

- Also of interest is the force aligned against group g, from the groups that oppose group g. And just as g can call in its friends from nearby neighborhoods, so can g's enemies.

Using these concepts we build up a notion of force which can be used to define neighborhood volatility, and then a group's security within the neighborhood.

First, group g's own force in neighborhood n is

$$Q_{ng} = \text{ceiling}(F_{ng} \times personnel_{ng})$$

where

F_{ng}	=	The force multiplier for group g in neighborhood n. This value is defined differently for civilian groups than other groups.
$personnel_{ng}$	=	The number of group g's personnel that are present in neighborhood n.

Force Multiplier for Civilian Groups: For civilian group g in neighborhood n, the force multiplier F_{ng} is defined as follows:

$$F_{ng} = a \times D_g \times M_g$$

Here, a is a multiplier, nominally 0.01, which reflects that only a small fraction of the population will be available to participate in a fracas at any given time.[14] D_g is a multiplier based on the demeanor of g.[15] M_g is a multiplier based on group g's mood, S_g. It is assumed that the more aggressive the demeanor and the worse the mood, the more likely it is that civilians will use force to aid their friends and hinder their foes:

$$D_g = \begin{cases} 1.5 & \text{if demeanor is Aggressive} \\ 1.0 & \text{if demeanor is Average} \\ 0.3 & \text{if demeanor is Apathetic} \end{cases}$$

$$M_g = 1 - b \cdot \frac{S_g}{100}$$

Here, b is a factor that determines how strongly a group's mood contributes to its force.[16] If b is 0.2, for example, the effect of mood will range from 0.8 (when group g is perfectly satisfied) to 1.2 (when group g is perfectly dissatisfied).

Force Multiplier for Force and Organization Groups: For force or organization group g in neighborhood n, the force multiplier F_{ng} is intended to account for the effectiveness and demeanor of the unit's personnel:

$$F_{ng} = E_g \times D_{ng}$$

[14]Model parameter: `force.population`
[15]Model parameter: `force.demeanor.*`
[16]Model parameter: `force.mood`

D_g is defined above. E_g, the unit force multiplier, should ideally depend on the type of the unit and the nature of its assets. In Athena, however, all units are the same and have no assets other than personnel, and so E_g is constant for each group. It is defined as shown in the following table:[17]

Characteristics of Unit *i*			E_g
Unit's group type is Organization (ORG)	Group's ORG type is:	NGO	0
		IGO	0
		Contractor	2
Unit's group type is Force (FRC)	Group's force type is:	REGULAR	25
		PARAMILITARY	15
		POLICE	10
		IRREGULAR	20
		CRIMINAL	8

Effect of unit activities: At the January 2007 meeting of the JNEM Rules Committee, it was suggested that a unit's activity could affect its force multiplier. Although this is an intuitively attractive notion, it may contain an implicit circularity. Our model presumes that a unit can successfully perform an activity given sufficient security. If the unit's extra contribution to force due its activity depends on its ability to successfully perform the activity, then we need to have computed security to determine the unit's contribution to force...yet we need the unit's contribution to force in order to compute security.[18] Consequently, the committee ruled that a unit's activity need not affect its force.

However, it is becoming clear that a unit's activity needs to affect security in a different way: a unit doing peacekeeping should have a different effect on the security of other groups than one conducting combat operations, for example. How this will factor in is yet to be determined.

Effect of Group Relationships: Friends and enemies are determined by their horizontal relationship, R_{fg}, where friends have a positive relationship and enemies have a negative relationship. We assume that group *f* will assist or oppose group *g* according to the strength of the relationship from *f*'s point of view. For convenience we define

---

[17] Model parameters: `force.orgtype.*`, `force.forcetype.*`

[18] RGC: The egg had to come before the chicken. That is, whether the unit has enough security to conduct an activity has to be based on the security before it started doing the activity or it could not have started. However, rather than introduce a new time lag (which we could do), I recommend we merely look at the previous value. With this rationalization, it is no longer circular, but inevitably sequential. It would become circular if the activity lowered the unit's effectiveness. I claim that should be taken into account when establishing the security threshold for that activity.

$$R^{+}_{fg} = \begin{cases} R_{fg} & \text{where } R_{fg} > 0 \\ 0 & \text{otherwise} \end{cases}$$

$$R^{-}_{fg} = \begin{cases} |R_{fg}| & \text{where} R_{fg} < 0 \\ 0 & \text{otherwise} \end{cases}$$

Then,

$$LocalFriends_{ng} = \sum_{f} Q_{nf} \cdot R^{+}_{fg}$$
$$LocalEnemies_{ng} = \sum_{f} Q_{nf} \cdot R^{-}_{fg}$$

Note that R_{fg} need not be symmetric; if we were to replace R_{fg} with R_{gf} in $LocalFriends_{ng}$, then we'd be assuming that group *f* would assist group *g* based on the strength of *g*'s feeling for *f*, which might not be at all the same thing as *f*'s feeling for *g*.

Next, we take neighborhoods into account:

$$Force_{ng} = LocalFriends_{ng} + h \cdot \sum_{m \text{ near } n} LocalFriends_{mg}$$

$$Enemy_{ng} = LocalEnemies_{ng} + h \cdot \sum_{m \text{ near } n} LocalEnemies_{mg}$$

Here, *h* is a factor, nominally 0.3, that reduces the effect of friends and enemies in nearby neighborhoods.[19] The phrase "*m* near *n*" denotes those neighborhoods *m* whose $Proximity_{mn}$ is *near*, that is, those neighborhoods *m* whose inhabitants regard neighborhood *n* as being nearby. As with R_{fg}, $Proximity_{mn}$ need not be symmetric.

In order to create scores, we'll need to normalize these values by the total force in the neighborhood:

$$TotalForce_{n} = \sum_{g} Q_{ng} + h \cdot \sum_{m \text{ near } n} \sum_{g} Q_{mg}$$

Then,

[19] Model parameter: `force.proximity`. An appropriate value of *h* should be selected during exercise construction to reflect the criteria used for selection of neighborhood boundaries. Normally, a playbox with large neighborhoods would have a smaller value for *h* than one with small neighborhoods.

$$\%Force_{ng} = \frac{Force_{ng}}{TotalForce_n} \times 100$$

$$\%Enemy_{ng} = \frac{Enemy_{ng}}{TotalForce_n} \times 100$$

5.2 Volatility

The volatility of a neighborhood is the likelihood of spontaneous violence within the neighborhood. Volatility depends on the balance of forces in the neighborhood, and is a key component of security. For example, an ORG group may hesitate to go into a neighborhood with high volatility, even if the ORG group is on friendly terms with all of the parties present in the neighborhood, simply because the chance of getting caught in a cross-fire is too high.

The volatility of neighborhood *n* can be regarded as a measure of the number of individuals potentially in conflict in neighborhood *n*, using the force values defined above:

$$Conflicts_n = \sum_g Enemy_{ng} \times Force_{ng}$$

$Conflicts_n$ ranges from 0 to ${TotalForce_n}^2$ because it counts every conflict from the point of view of each of the parties involved. Scaling to the range 0 to 100 yields

$$Volatility_n = \frac{Conflicts_n}{TotalForce_n^2} \times 100$$

5.3 Security

A group's security in a neighborhood determines the kinds of activities the group can perform within that neighborhood. Different levels of security are needed for different kinds of operations. The group's level of security depends on the assets present in the neighborhood, including both those belonging to the group and those belonging to friendly and unfriendly groups, as well as on the nature of both assets and groups and the volatility of the neighborhood. In general:

- The presence of friendly forces should increase group *g*'s security in neighborhood *n*.
- The presence of enemy forces should decrease group *g*'s security in neighborhood *n*.
- Increased volatility should decrease group *g*'s security in neighborhood *n*.

The security of group *g* in neighborhood *n* is defined as follows:

$$Security_{ng} = \frac{\%Force_{ng} - \%Enemy_{ng} - v \cdot Volatility_n}{100 + v \cdot Volatility_n} \times 100$$

where v is the volatility scaling factor, nominally 1.0, which can be used to reduce the effects of volatility on security.[20] The denominator scales $Security_{ng}$ so that it ranges from -100 to +100.

For use in rules, $Security_{ng}$ will often be translated to a symbolic value, as shown in the following table:

Range	Symbol
$25 < Security_{ng} \leq 100$	High
$5 < Security_{ng} \leq 25$	Medium
$-25 < Security_{ng} \leq 5$	Low
$Security_{ng} \leq -25$	None

For some uses, $Security_{ng}$ will be converted to a multiplicative factor using a Z-curve.

[20] Model parameter: `force.volatility`

6. EFFECTS OF UNIT ACTIVITIES

An actor's strategy can deploy group personnel to neighborhoods, and then assign those personnel activities. The end result is a collection of units that contain the personnel and are tagged with the relevant activities.

6.1 Force Presence and Activities

Force units present in a neighborhood can affect civilian attitudes in a neighborhood in two ways: by their mere presence, and by engaging in other activities.

Force Presence: A force group's presence in a neighborhood is measured as the total number of personnel in units that belong to the group and are present in the neighborhood.[21]

Force Activities: A unit assigned an activity is assumed to carry out the activity if it can possibly do so. Each activity requires that the acting group have a minimum level of security in the neighborhood. The following table lists the force activities and the nominal minimum security level for each activity.[22]

Force Activity	Minimum Security Required	Shifts	Coverage Function
Presence	n/a	1	25/1000
Checkpoint	Low	1	25/1000
CMO -- Construction	High	1	20/1000
CMO -- Development (Light)	Medium	1	25/1000
CMO -- Education	High	1	20/1000
CMO -- Healthcare	High	1	20/1000
CMO -- Industry	High	1	20/1000
CMO -- Infrastructure	High	1	20/1000
CMO -- Law Enforcement	Medium	1	25/1000
CMO -- Other	High	1	20/1000
Coercion	Medium	1	12/1000

21 Ideally, the nature of each unit would also be taken into account, at least at the level of "boots, wheels, or tracks," i.e., the number of personnel on foot, in wheeled vehicles, and in tracked vehicles. Tanks tear up the pavement, for example, and thus should have an increased negative effect on QOL. At present, Athena does not model units to this level of detail.

22 Model parameter database, `activity.FRC.*`

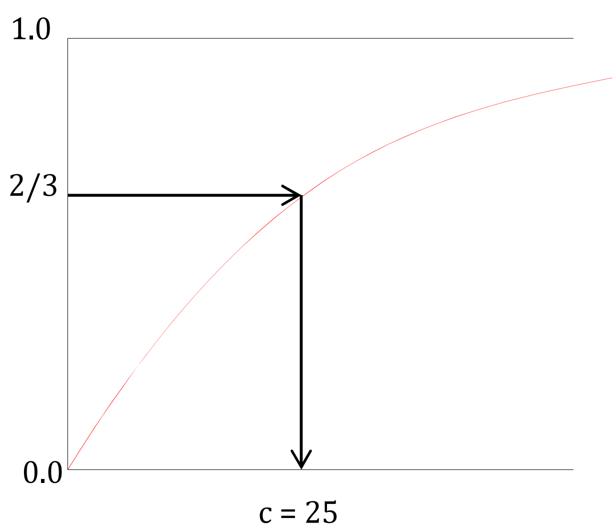
Force Activity	Minimum Security Required	Shifts	Coverage Function
Criminal Activities	Medium	1	10/1000
Curfew	Medium	1	25/1000
Guard	Low	1	25/1000
Patrol	Low	1	25/1000
PSYOP	Medium	1	1/50000

Thus, a force group can enforce a curfew or provide law enforcement in a neighborhood only if the group's security is at least Medium in that neighborhood. The extent to which a group is engaging in an activity in a neighborhood is measured by the total number of personnel in units that:

- Belong to the group
- Are present in the neighborhood
- Are assigned to do the activity
- Are on the current shift (the number of personnel available at any given time is the total divided by the number of shifts defined for the activity type)

This figure is automatically 0 if the minimum security requirement is not met.

Coverage Fractions: Whenever a group is present or conducting an activity in a neighborhood, the extent to which its presence or activities affect the local population is termed its *coverage fraction* for the activity. The coverage fraction is a value from 0 to 1; it is computed by comparing the group's personnel (present, or conducting an activity) with the total population of the neighborhood, using a function like that shown below.



In this example, if the group has deployed 25 troops per 1000 people in the neighborhood then the coverage fraction is 2/3, rising exponentially toward 1.0 as the troop density goes toward infinity. The coverage fraction determines how much of the potential satisfaction change is gained or lost by the group's presence or activity. The x-axis is usually troops per 1000 population, as shown, but can vary; for PSYOP, it is troops per 50,000 population. Thus, the two parameters required to compute a coverage fraction are {*c*, *d*} where *c* is the troop density at the 2/3rds point, and *d* is the population denominator, e.g., 1000 or 50,000. The coverage function is often specified in the form c/d. In the example shown above, then, the coverage function may be specified as either 25/1000 or {25, 1000}. The actual function is as follows. First, the troop density is

$$TD = \frac{p \cdot d}{Population_n}$$

where *p* is the number of personnel engaged in the activity. Then the coverage fraction is

$$CF = 1 - e^{-\frac{TD \cdot \ln 3}{c}}$$

Example: Force group *g* has 750 troops in neighborhood *n*, which has a total population of 40,000 people:

p = 750

$Population_n$ = 40,000

The coverage fraction parameter, *c/d*, is nominally 25/1000 for mere presence. The troop density for group *g* is therefore

$$TD = \frac{p \cdot d}{Population_n} = \frac{750 \cdot 1{,}000}{40{,}000} = 18.75$$

The coverage fraction is therefore

$$CF = 1 - e^{-\frac{18.75 \cdot \ln 3}{c}} = 1 - e^{-0.824} = 1 - 0.44 = 0.56$$

A force group *g*'s coverage fraction for activity *a* in neighborhood *n* is denoted $Coverage_{nga}$.

Composite Coverage Fractions: If two force groups were cooperating in enforcing a curfew, we would expect the curfew to suppress civilian activities based on the total personnel used in enforcing it, independent of the fact that two groups are involved. In cases like these it is useful to employ a *composite coverage fraction*. This fraction can be computed based on the total troop density across all relevant groups; it can also be computed from each group's coverage fraction for the activity, as follows:

$$Coverage_{na} = 1 - \prod_{g} (1 - Coverage_{nga})$$

6.2 Organization Activities

Units belonging to organization groups have no effect on civilian satisfaction due to their mere presence. Organization units may perform the following activities; the effect of these activities is modeled in the same way as force activities are, with these distinctions:

- The minimum security required for an activity depends primarily on the organization type, and not on the activity. NGOs and IGOs require "high" security, while CTRs require "medium" security.[23]

- Organization units can perform only a subset of the activities that a force unit can perform.

Organization Activity	Minimum Security Required	Shifts	Coverage Function
CMO -- Construction	High/Medium	1	20/1000
CMO -- Education	High/Medium	1	20/1000
CMO -- Healthcare	High/Medium	1	20/1000
CMO -- Industry	High/Medium	1	20/1000
CMO -- Infrastructure	High/Medium	1	20/1000
CMO -- Other	High/Medium	1	20/1000

6.3 Civilian Activities

Civilian units may perform the following activities; the effect of these activities is modeled in the same way as force activities, with the distinction that security is irrelevant.

Civilian Activity	Minimum Security Required	Shifts	Coverage Function
Displaced Persons	n/a	1	25/1000
In Camp	n/a	1	n/a

[23]Model parameter: `activity.ORG.*`

Units assigned the "Displaced Persons" activity are assumed to contain population driven from their homes and living with and among the civilian population in the area in which they find themselves. Coverage is computed for these persons in the normal way.

Units assigned the "In Camp" activity are assumed to be displaced persons who have been settled in a refugee camp. Such persons are ignored, for two reasons. First, such camps are often established in out of the way places, where there are no local civilians to be affected. Second, the attitudes of the displaced persons themselves are of interest; but GRAM does not yet allow populations to move around in this way.

6.4 Activity Situations

When a group has an activity with coverage greater than 0.0 in a neighborhood, we say that an *activity situation* exists in that neighborhood. The effects of these situations on local attitudes are assessed by the DAM rules; see the *Athena Rules* document. Note that no activity situation is created for the "In Camp" activity.

7. ENVIRONMENTAL SITUATIONS

Environmental situations are on-going circumstances in a neighborhood that affect all of the civilians resident in the neighborhood. Environmental situations have a coverage fraction, just as activity situations do; however, for environmental situations the coverage fraction is set by the analyst when the situation is created. The complete set of environmental situation types (and the accompanying DAM rules) are given in the *Athena Rules* document.

8. SERVICES

A service is something provided to the civilians, the level of which affects civilian attitudes.

8.1 Overview

At present Athena models one kind of service, Essential Non-Infrastructure (ENI) services (Section 8.2). We expect additional services to be handled in a similar way; this section presents an overview of the general concept.

8.1.1 Services vs. Environmental Situations

Athena has historically modeled services (e.g., power and water) using environmental situations, or "ensits" (Section 7). In the ensit model, the service is presumed to be provided unless the related ensit exists, in which case the service is out. There is usually a satisfaction penalty when service lapses, a satisfaction gain when service is restored, and a satisfaction drain so long as the service is out. This model derives from JNEM's five-day training exercises, where ensits were presented as problems for the training audience to solve—the training audience is rewarded for speedy resolution, and no situation lasts for a long time because the time horizon is so short.

As it turns out, ensits are not well-suited for Athena time horizons. In addition, they do not allow actors to have a role in providing services to the civilians—something an actor does to influence, benefit, and ingratiate the civilians, and to make them dependent on the actor's leadership.

Consequently, we have developed the notion of a provided service. Actors may be directly responsible for providing the service, or may indirectly facilitate provision of the service.[24]

8.1.2 Levels of Service

A service is provided to the civilians by one or more actors. It can be provided to a greater or lesser extent, and the primary driver of attitude and relationship change is whether the provided level of service meets or fails to meet the expectations of the civilians. In particular, a service has:

- An Actual Level of Service (*ALOS*): the amount of the service currently being provided. This is a continuous variable which rises and falls over time. The natural units for the level of service will of course vary from one kind of service to another.

---

[24] In future versions, actors may also be able to interfere with the provision of a service.

- A Required Level of Service ($RLOS$). If the $ALOS$ is less than the $RLOS$, serious hardship results.

- An Expected Level of Service ($ELOS$). The $ELOS$ will approach the $ALOS$ over time. In general, the population grows happier when the $ALOS$ exceeds the $ELOS$, and unhappier when the opposite is true.

- A Saturation Level of Service ($SLOS$). When the $ALOS$ reaches saturation, the population has all the service they can use; increases beyond SLOS cause no net gain in satisfaction.

For simplicity's sake, we normalize all of these variables to the Saturation Level of Service. That is, $SLOS$ is scaled to be 1.0, and the other variables are all represented as fractions of $SLOS$. As a result,

$$SLOS = 1.0$$

$$0.0 \leq RLOS \leq 1.0$$
$$0.0 \leq ELOS \leq 1.0$$
$$0.0 \leq ALOS$$

Note that we allow $ALOS$ to exceed the saturation level; the actors might indeed provide a superfluous quantity of the service, although they get no benefit from doing so.

8.1.3 Service Cases

As an aid to analysis of attitude and relationship effects, we define a number of cases. At any given time, with respect to a particular group g in a particular neighborhood n, one of the four following cases obtains:

- **Case R−**: The actual level is less than the required level. The people are hurting, and the expected level doesn't matter.

- **Case E−**: The actual level is at least the required level, but is less than expected. The people will be unhappy.

- **Case E**: The actual level is close to the expected level. The people are getting what they expect.

- **Case E+**: The actual level is well above the expected level. The people are getting more than they expect, and they like it. (Note that the expected level can approach but never exceed saturation.)

8.2 Essential Non-Infrastructure (ENI) Services

Among the tactics actors can use when trying to achieve their goals is funding of those essential civilian services that do not require significant amounts of infrastructure.[25] To avoid repeating that mouthful every time we want to refer to them, we are calling them "essential non-infrastructure" or "ENI" services, for short. Civilian groups' support for actors will be affected and their satisfaction levels may change. This section describes how to compute those changes. The model takes into account that the actors will usually be competing.

ENI services include such things as welfare payments, courts, public education, and any other service for which funding alone adequately describes what is needed to "make it happen." The model can accommodate services that require other assets as well as money if those other assets are used in strict proportion to the funding, but details of how to do so are not suggested here. Athena will not execute any tactic an actor cannot afford (in terms of any asset).

Modeled services do not include provision of water, food, electrical power, or other utilities; shortages of those services are modeled as *ensits*, which can be mitigated by tactics that specify appropriate force group activities. Nor do they include law enforcement, which can be provided by assigning the CMO-Law Enforcement activity.

Other essential services that require substantial amounts of infrastructure, such as public transit and postal service are not covered by this model, nor are the rather distant governmental services of defense, central banking, and law-making.

8.2.1 Service vs. Funding

Because ENI services by definition do not require investment in infrastructure, we assume that the amount of service provided at any given time depends on the total funding for ENI services at that time, summed over all actors' contributions. In particular, let

$$A_g = \left(\frac{F_g}{P_g}\right)^{\beta_r}$$

where

g	=	A civilian group to whom ENI services are provided.
A_g	=	The actual level of service for group g, as a fraction of the saturation level of service.

[25] Services that require infrastructure are significantly more complicated; and moreover, each kind of service is likely to require its own unique infrastructure model. Thus, for this version we have limited ourselves to the simplest case.

P_g	=	The funding required to saturate the supply of ENI services to group g, in \$/week.
F_g	=	Total funding of ENI services for group g, in \$/week.
β_r	=	Shape parameter[26] for the service vs. funding curve, where $\beta_r > 0$. Values less than 1.0 imply economies of scale. A value of 1.0 indicates that service is directly proportional to funding.
r	=	The urbanization level of the neighborhood in which g lives, one of ISOLATED, RURAL, SUBURBAN, or URBAN.

The saturation level of funding, P_g, depends on the quantity of service required per person in the group and the number of people in the group:

$$P_g = POP_g \times S_r$$

where

POP_g	=	The population of group g.
S_r	=	The per capita cost of providing saturated ENI services in a neighborhood with urbanization r.[27]

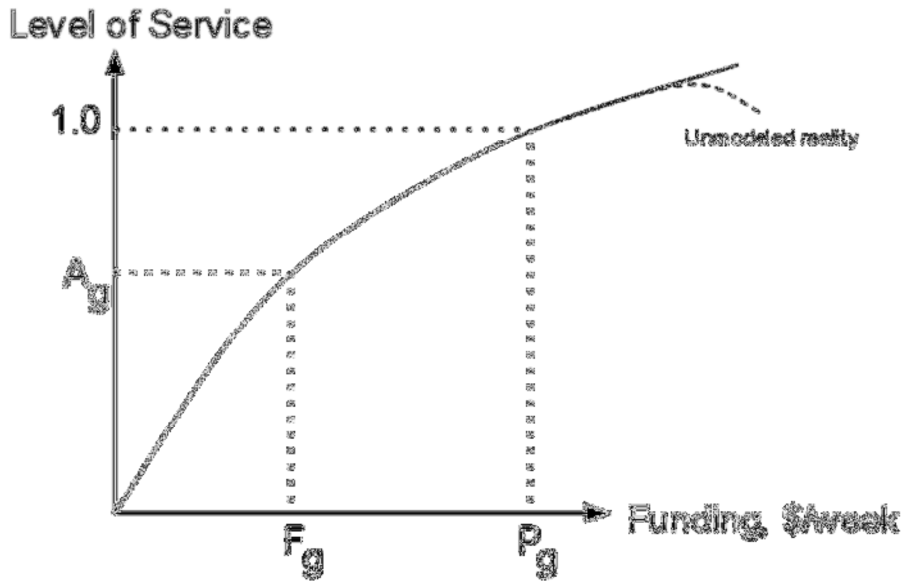
The nominal values for S_r are as follows:

Urbanization	$\boldsymbol{S_r}$
URBAN	0.40
SUBURBAN	0.20
RURAL	0.10
ISOLATED	0.01

The resulting cost curve is shown in the figure, below. As drawn, it indicates that there are economies of scale ($\beta_r < 1.0$). Note that very large funding levels can be expected (in the real world) to show diminishing returns; the model can be modified to reflect that expectation if necessary, but the anticipated operating range is well below saturation.

[26] Model Parameter: `service.ENI.beta.`*`urbanization`*, nominally 1.0 for all urbanization levels.

[27] Model Parameter: `service.ENI.saturationCost.`*`urbanization`*.



8.2.2 Funding by Individual Actors

We assume that an actor can provide ENI services in any neighborhood in which he has at least minimal direct support.[28] Thus, multiple actors can choose to provide ENI services to a particular group. The total funding for ENI services to group g is then simply

$$F_g = \sum_a F_{ga}$$

where

F_{ga}	=	Actor a's funding for ENI services to group g.

Actors choose how much funding to provide for ENI services using the FUNDENI tactic, which is described in the *Athena User's Guide*.

8.2.3 Expected Level of Service

We compute the expected level of service, X_g, by exponential smoothing from the actual level of service and the previous expected level, capping the expectation at the saturation level:

$$X_g = X_g^- + \alpha \times \left(\min(A_g, 1) - X_g^-\right)$$

[28] Model parameter: `service.eni.minSupport`, nominally 0.0.

where

X_g	=	Group g's expected level of service
X_g^-	=	Group g's expected level of service at the previous week.
A_g	=	The actual level of service received by group g.
α	=	The exponential smoothing parameter, where $0 \leq \alpha \leq 1$

The value of α can be thought of as the reciprocal of the average age of data in weeks, i.e., if $\alpha = 0.5$ then the expectation is based on two weeks of data. Greater values of α imply that expectations adjust more quickly than lower values. If α is 1, the expected level of service will simply be just the actual level of service; the civilians will expect whatever they get. If α is 0, then the expected level of service never changes.

We assume that civilians more rapidly grow accustomed to good fortune than to hardship, and consequently the value of α depends on whether or not A_g exceeds X_g^-:

$$\alpha = \begin{cases} \alpha_A & \text{if } A_g > X_g^- \\ \alpha_X & \text{if } A_g \leq X_g^- \end{cases}$$

Here, α_A and α_X are model parameters[29], nominally 0.5 and 0.25 respectively.

The initial value for X_g at time 0 is simply the actual level of service A_g at time 0. Thus, the civilians initially expect what they are currently getting.

8.2.4 Satisfaction Effects

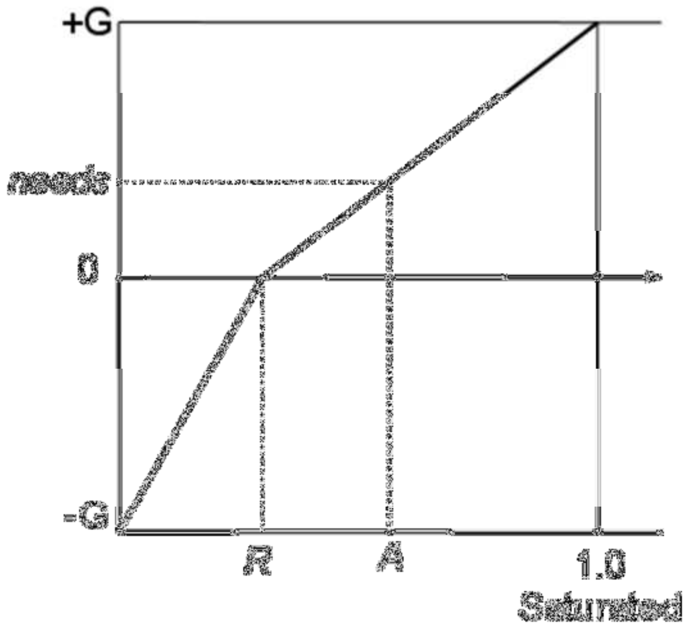
The rate of change of civilian satisfaction due to the availability of services depends on two major considerations: (1) The difference between the actual level of service and the civilians' expectation of service and (2) how the actual level compares to minimum (required) and maximum (saturation) levels. In general, satisfaction levels should go down if the civilians are receiving less than the minimum level of service. If the civilians are receiving at least the minimum level of service, then satisfaction should go down if they are receiving less than they expect, and should go up if they are receiving more than they expect.

To capture these two effects, we define two factors: the needs factor, $needs_g$, and the expectations factor, $expectf_g$. These two factors are then used in the ENI rule set to assess satisfaction effects; see the *Athena Rules* document for the specifics.

[29] Model parameters: `service.ENI.alphaA`, `service.ENI.alphaX`.

8.2.4.1 The Needs Factor

The $needs_g$ factor measures the degree to which the actual level of service received by group g is less than or greater than that required for survival. It is based on the piecewise-linear function shown in the following figure:



In this diagram,

$needs_g$	=	The "needs" factor for group g.
A_g	=	The actual level of service received by g.
R_r	=	The required level of service for residents of neighborhoods with urbanization r, where r is the urbanization of g's neighborhood.[30]
G_N	=	A model parameter,[31] the gain on $needs_g$, nominally 2.0.

The nominal values for $\boldsymbol{R_r}$ are shown in the following table:

[30] Model parameter: `service.ENI.required.urbanization`.

[31] Model parameter: `service.ENI.gainNeeds`.

Urbanization	$\boldsymbol{R_r}$
URBAN	0.60
SUBURBAN	0.40
RURAL	0.20
ISOLATED	0.00

The function is computed using the following algorithm:

If $A_g = R_r$,
  Let $needs_g = 0.0$.
If $A_g \geq 1.0$,
  Let $needs_g = G_N$.
If $A_g < R_r$,
  Let $needs_g = G_N \times \frac{A_g - R_r}{R_r}$.
If $A_g > R_r$ (the only remaining case),
  Let $needs_g = G_N \times \frac{A_g - R_r}{1 - R_r}$.

The result is a number between $-G_N$ and $+G_N$, or nominally between -2 and 2.

The current model presumes that providing service in excess of saturation has no additional effect. This is unlikely. Providing excess service might decrease satisfaction with respect to AUT, for example. On the other hand, it might not really be possible to provide service in excess of saturation…which means that excess money spent on it is going for something else, and likely improving *somebody*'s satisfaction, and probably illicitly. In light of the abstract nature of ENI services as modeled in this version of Athena, then, we will continue to assume that the effect of excess service is flat.

8.2.4.2 The Expectations Factor

The rate of change of civilian satisfaction should depend on the difference between the expected and actual levels of service. When expectations are not met, satisfaction should go down; when they are exceeded, satisfaction should go up. The expectation factor, $expectf_g$, ranges from $-G_E$ and $+G_E$, or nominally between -2 and 2, and is computed simply as follows:

$$expectf_g = G_E \times \left(\min(1, A_g) - X_g\right)$$

where

$expectf_g$	=	The "expectations" factor for group g.
A_g	=	The actual level of service received by g.
X_g	=	The expected level of service for g.

G_E	=	A model parameter,[32] the gain on $expectf_g$, nominally 2.0.

8.2.5 Vertical Relationship Effects

To compute the vertical relationship effects, we use the four service cases described in Section 8.1.3: **R−**, **E−**, **E**, and **E+**.

8.2.5.1 Categorize the Actual Level of Service

The first step is to determine which of the four cases applies:

- If $A_g < R_r$, then the actual is less than required; case **R−** applies.
- If $\left|A_g - X_g\right| \leq \delta \times X_g$, then the actual is about the same as expected; case **E** applies.
- If $A_g < X_g$, then the actual is less than expected; case **E−** applies.
- Otherwise, the actual is greater than expected, and case **E+** applies.

where

A_g	=	The actual level of service received by g.
X_g	=	The expected level of service for g.
R_r	=	The required level of service for residents of neighborhoods with urbanization r, where r is the urbanization of g's neighborhood, as defined above.
δ	=	A model parameter, the half-width of the band around the expected value that defines the notion "approximately equal to".[33]

8.2.5.2 Categorize Each Actor's Contribution

Next we must compute the credit, $credit_{ga}$, given to each actor a for the level of ENI services provided to group g.

If there is no funding, nobody gets any credit. This especially includes the actor in control of the neighborhood, if any:

- If $F_g = 0$, $credit_{ga} = 0$ for all a.

---

[32] Model parameter: `service.ENI.gainExpect`.

[33] Model parameter: `service.ENI.delta`, nominally 0.1.

Otherwise, we give credit to each actor according to the fraction of the total funding that he provides, but only up to the amount needed for saturation. Moreover, we give credit to the actor in control of the neighborhood *first*; if he provides a saturation level of service all by himself, the other actors get no credit. If he provides less than the saturation level of service, the other actors can only take credit for the increment of service provided up to the saturation level.

Let C be the actor in control of g's neighborhood. Then,

$$credit_{gC} = \min\left(\frac{F_{gC}}{\min(F_g,\ P_g)}, 1\right)$$

$$credit_{ga} = \frac{F_{ga}}{\sum_{b \neq C} F_{gb}} * \left(1 - credit_{gC}\right) \text{ for all } a \neq C$$

where

P_g	=	The funding required to saturate the supply of ENI services to group g, in \$/week.
F_g	=	Total funding of ENI services for group g, in \$/week.
F_{ga}	=	Actor a's funding for ENI services to group g.

We then categorize these contributions as shown in the following table:

Contribution	Categorization
0% ≤ $credit_{ga}$ ≤ 20%	**N**: Actor a is providing **negligible** funding.
20% < $credit_{ga}$ ≤ 50%	**S**: Actor a is providing **some** funding.
50% < $credit_{ga}$ ≤ 100%	**M**: Actor a is providing **most** of the funding.

8.2.5.3 Award Vertical Support by Case, Control, and Contribution

Finally, we compute ΔV_{eni}. Group g's vertical relationship with actor a depends on which of the four service cases applies, whether or not a is in control of g's neighborhood, and the relative size of a's contribution to the service g receives, as shown in the following table. The table is based on the following assumptions:

- The actor in control gains support so long as g is getting at least as much service as they expected, regardless of who is actually providing it. That is, so long as the civilians are getting the service they need, the actor in control is seen as doing his job.

- The actor in control loses support so long as g is getting less service than they expected. That is, the actor in control is held responsible for problems—but his level of effort is noticed.

- Other actors gain support for any non-negligible contributions they make, but are not held responsible for problems.

	Relative contribution (Actor in control)			Relative contribution (Actor not in control)		
Case	**Negligible**	**Some**	**Most**	**Negligible**	**Some**	**Most**
E+	XL+	XL+	XL+	0	XL+	XXL+
E	L+	L+	L+	0	L+	XL+
E−	XL−	L−	M−	0	M+	L+
R−	XXL−	XL−	L−	0	S+	M+

The value of ΔV_{eni} is applied as discussed in Section 4.3.2.

9. ATHENA ATTRITION MODEL (AAM)

The Athena Attrition Model (AAM, pronounced "aim") models attrition to force, organization, and civilian units. AAM provides a basic framework for attrition and attrition-related modeling that includes most areas, but models each area in a simple way that can be improved in later versions.

Unlike traditional, detailed attrition models, which model individual firefights between opposing, detected, uniformed forces at specific points of time at specific locations, AAM models attrition caused by both uniformed and non-uniformed forces over longer periods of time (typically a week) at unspecified—but nearby—locations.

By uniformed forces we are referring to military, paramilitary or other forces wearing uniforms or other outward indication that they belong to a particular force group. By non-uniformed forces we are referring to combatants (typically insurgents) who belong to a particular force group, but who are dressed to blend in with the local civilian population in order to avoid being recognized by enemy forces.

AAM does not explicitly consider the effects of equipment and weapons systems in use by force units (nor are these modeled in Athena). Rather, it is assumed that forces are appropriately equipped for their activities. The effects of equipment and weapons systems are implicit in the parameters of the various equations and algorithms.

In addition, AAM supports magic attrition of individual units, of any specific group in a neighborhood, and of all civilians in a neighborhood.

9.1 Overview

9.1.1 Attrition in the Real World

Attrition is the death of unit personnel due to inter-group violence, ranging from chance altercations to targeted attacks (assassinations and ambushes) to riots to open force-on-force combat.

In the real world, attrition can be caused by uniformed forces, non-uniformed forces, certain types of organizations (ORGs), hostile/militant civilians who do not belong to any force group, and complex crowds containing various mixes of people: non-uniformed forces, civilian supporters of force groups, "rent-a-crowds," and ordinary folks who are sympathetic to the cause, just looking for some excitement, or just happen to get caught up in things. Attrition can be suffered by all of the above, as well as by peaceful organizations and by innocent civilian bystanders.

At present Athena does not attempt to deal with all of these possibilities; many of them can be handled as special events using magic attrition.

9.1.2 Requirements for This Version

- Attrition shall take place in neighborhoods, based on the groups present in the neighborhood.
- All personnel are present in neighborhoods as unit personnel, i.e., as visible unit icons, staffed according to the actors' strategies.
- Force groups may be designated as *uniformed* or *non-uniformed*.
- Crowds of civilians (simple or complex) will not be represented.[34]
- For convenience,
    - A Uniformed Force (UF) is the collection of all units within a neighborhood that belong to a particular force group that is designated as uniformed.
    - A Non-uniformed Force (NF) is the collection of all units within a neighborhood that belong to a particular force group that is designated as non-uniformed.
    - The attitude changes due to attrition to a displaced civilian unit will be done as though the attrition was incurred in the unit's neighborhood of origin.[35]
- A neighborhood may contain any combination of UFs and NFs.
- Groups may be friendly, enemy, or neutral with each other, based on the value of their relationships.
- Attrition occurs during engagements between enemy forces.
- AAM shall model attrition caused by the following types of engagement:
    - UF hunting down enemy NF.
    - NF ambushing enemy UF.
    - Collateral damage to civilians from all engagements between forces.
- All casualties from attrition will be kills; there will be no computation of wounded.
- All attrition to organization group personnel will be handled by magic input.
- All mass killings of civilians by any group will be handled by magic input.
- Coercion of civilians by force groups is modeled as a force activity; it improves cooperation if the coercing force has sufficient security and coverage. Explicit killing of civilians by UF or NF for the purposes of coercion is not currently modeled.
- A UF can hunt down enemy NF in a neighborhood. In this case, the UF will benefit in the following ways if it can increase the cooperation of the population with itself and decrease the cooperation of the population with the NF:
    - Fewer attacks on UF by NF
    - Better loss exchange rates for UF when NF does attack
    - More attacks by UF against NF with less collateral damage to civilians on each attack.
- To find enemy NF, a UF must have troops present in the neighborhood. Increasing the coverage of the troops will produce more attacks on enemy NF but will also make the UF more vulnerable to attack by NF (up to a point).

---

[34] At Athena's timescale, most crowd-related phenomena are events rather than situations.

[35] This is not quite right; but the current model for displaced civilians is a stopgap until GRAM can support it properly.

- Force groups require funding/resources in order to conduct attacks. Therefore, reducing actor funding reduces the number of attacks the actor's forces can initiate.

9.1.3 Requirements for Later Versions

- To find enemy NF, a UF must have troops in the neighborhood with activities that expose them to the local population. Increasing the coverage of such activities will produce more attacks on enemy NF but will also make the UF more vulnerable to attack by NF (up to a point).
- NF can conduct IED attacks against UF.
- Force groups will be allowed to recruit new members from the militant pools of local civilian groups.
    - Under certain circumstances, such as a sustained lack of salary payments, some members may desert.
- Individual units might be designated as Uniformed or Non-uniformed, perhaps according to their activity, allowing a single force group to have both UF and NF.
- AAM will also model attrition caused by the following types of engagement.
    - Civilian vs. Civilian
    - UF vs. UF
    - NF vs. NF
    - Collateral damage to organizations from all engagements between forces.

9.1.4 Simplifying Assumptions

In order to keep AAM within the realm of the possible we made the following simplifying assumptions, which are still in force in this version:

- NFs will ambush UFs to inflict a few casualties and then run in order to limit their own casualties
- When a UF discovers an NF cell, the UF will attack with overwhelming force and suffer no casualties
- Neither UFs nor NFs will intentionally kill civilians in this model
- NFs will not explicitly use crowds to promote unrest or otherwise further their agenda.
- UFs will cause collateral damage (the killing of civilian bystanders) when attacking or defending against NFs. NFs will not kill civilians when they attack or are attacked by UF.
- There is no direct UF vs. UF or NF vs. NF conflict within the playbox.

9.2 Uniformed vs. Non-Uniformed Forces

Every force group will have a flag indicating whether it is uniformed or non-uniformed. We expect that regular military, paramilitary, and police groups will usually be uniformed (UF), and irregular military and criminal groups will usually be non-uniformed (NF).

9.3 Units and Unit Activities

Rather than representing a specific body of troops in some location, an Athena unit really represents an allocation of some number of personnel to an activity within the neighborhood, such as PATROL or COERCION. Troops assigned the activity NONE are presumed to be in reserve. At Athena's time scale, it makes no sense to model unit movement or location in any detail. Athena units have a precise location within each neighborhood, but only as an aid to visualization.

Some activities involve more exposure to the local population than others. Increased exposure has a number of effects: the unit is more likely to find enemy units; the unit is more likely to get intel from the local civilians; the unit is more likely to be attacked by enemy units. The following force activities are deemed to involve significant exposure to the local civilians:

CHECKPOINT
CMO_CONSTRUCTION
CMO_DEVELOPMENT
CMO_EDUCATION
CMO_EMPLOYMENT
CMO_HEALTHCARE
CMO_INDUSTRY
CMO_INFRASTRUCTURE
CMO_LAW_ENFORCEMENT
CMO_OTHER
COERCION
CRIMINAL_ACTIVITIES
CURFEW
PATROL
PSYOP

However, the results of such increased exposure are being deferred to a later version.

The following activities do not have significant exposure in the sense used here:

GUARD
NONE

9.4 Unit Number and Unit Size

Because units are used to allocate troops to particular activities, rather than to represent any military TOE, the breakdown of a force into units has no tactical significance. Consequently, the breakdown of a force into units will be ignored when computing the amount of attrition each group incurs.

9.5 Attrition and Mobilized Troops

All attrition comes out of the attrited units; but it also comes out of each force group's pool of mobilized troops, thus reducing the personnel available for deployment and assignment the following week. The following sections will speak only of attrition to units; the effect on the pool of mobilized troops is implicit.

9.6 Magic Attrition

The analyst can attrit units and groups magically. All attrition to civilian units will be assessed by the relevant DAM rule set. Attrition to civilians displaced from their neighborhood of origin will be assessed as though they were attrited in their neighborhood of origin.

9.6.1 Magic Attrition to Units

The analyst can attrit a specific unit, of any type.

9.6.2 Magic Attrition to Groups

The analyst can attrit a specific group in a specific neighborhood. If the group is a civilian group, only units present in the neighborhood will be attrited.

9.6.3 Magic Attrition to Neighborhoods

The analyst can also choose to apply attrition to all civilian units that happen to be a neighborhood; this is equivalent to collateral damage incurred during normal attrition.

9.7 Antagonists and ROEs

Neighborhood n can contain the kinds of forces listed in Section 9.2: NF and UF.[36] Attrition occurs when two forces a and b are antagonists as defined in this section.

9.7.1 Attacking ROEs: Maximum Number of Attacks

Attacking ROEs are specified by the actor that owns the attacking group. Each actor must manage his resources, and consequently each attacking ROE includes a "maximum number of attacks", N_{max}, as set by the actor in accordance with his strategy and the resources he has available. No group will attack more times in a week than is specified in its ROE.

9.7.2 Attacking ROEs: UF

Uniformed forces (UF) may attack non-uniformed forces (NF) in a neighborhood.

[36]We are ignoring civilian and organization group "forces" for the time being.

A UF will attack a particular NF only when directed to do so by its Rules of Engagement. In principle, each UF has an *attacking ROE* with respect to each NF in each neighborhood. This ROE may be set to one of the following values:

- ATTACK
- DO_NOT_ATTACK

In practice, Athena will only track ROEs set to ATTACK; if no ROE is set, no attacks will take place.

For example, consider the following Attacking ROE table, in which BLUE is a uniformed force and ALQ is a non-uniformed force.

Nbhood	***f***	***g***	**Max Attacks**
N1	BLUE	ALQ	5
N2	BLUE	ALQ	10

This table indicates that Blue has been ordered to attack Al Qaeda in neighborhood N1 up to 5 times during one week, and in neighborhood N2 up to 10 times. However, Blue will not attack Al Qaeda in any other neighborhood. In either case, Blue will attack Al Qaeda as many times as it can up to the maximum limit.

The maximum number of attacks is set by the force group's owning actor, and is constrained in the strategy model by the actor's cash-on-hand.

9.7.3 Attacking ROEs: NF

Non-uniformed forces (NF) may attack uniformed forces (UF) in a neighborhood.

Just as with UFs, an NF will attack a particular UF only when ordered to do so by its Rules of Engagement. In principle, each NF has an *attacking ROE* with respect to each UF. This ROE may be set to

- HIT_AND_RUN: The NF will attempt to kill UF troops through ambushes, IED attacks, and so forth, while limiting their own losses by running as soon as possible.

- STAND_AND_FIGHT: The NF will attempt to kill UF troops by sucking them into ambushes and pinning them down, so as to kill as many UF troops as possible regardless of their own losses.

- DO_NOT_ATTACK

In practice, Athena will not track ROEs set to DO_NOT_ATTACK; if neither HIT_AND_RUN nor STAND_AND_FIGHT is set, no attacks will take place.

For example, consider the following Attacking ROE table, in which BLUE is a uniformed force and ALQ is a non-uniformed force.

Nbhood	***f***	***g***	**ROE**	**Max Attacks**
N1	ALQ	BLUE	STAND_AND_FIGHT	7
N2	ALQ	BLUE	HIT_AND_RUN	12

Al Qaeda will attack Blue in both neighborhoods. In N1 it will try to attack up to seven times during the week, and will stand and fight, taking greater casualties. In N2, however, it will try to attack twelve times during the week, but will try to conserve its strength by using hit and run tactics. In either case the actual number of attacks is limited by the number of BLUE troops and the cooperation of the neighborhood with both groups.

The maximum number of attacks is set by the force group's owning actor, and is constrained in the strategy model by the actor's cash-on-hand.

9.7.4 Defending ROEs

In addition to its Attacking ROEs, each uniformed force has a Defending ROE in each neighborhood, which can have one of three values:

- FIRE_BACK_IMMEDIATELY: Fire back immediately if fired upon. This will cause collateral damage to civilians whenever the UF defends itself.

- FIRE_BACK_IF_PRESSED: Do not fire back unless the enemy continues to engage. This will cause collateral damage to civilians only when the UF defends itself against an NF with an attacking ROE of STAND_AND_FIGHT.

- HOLD_FIRE: Do not fire back. The UF will never cause collateral damage when defending.

The default is to FIRE_BACK_IF_PRESSED.

9.8 The Attrition Cycle

Attrition will be computed at regular intervals; the default interval is one week.[37] Note that magic attrition that occurs during the interval is accumulated, and assessed for attitude implications along with the normal attrition computed here.

[37] Model parameter: `aam.ticksPerTock`

At the end of the attrition interval, the following algorithm will compute the attrition for the interval:

For each neighborhood *n*:
  Determine each pair *a,b* of antagonists in *n*.
  For each pair *a,b*:
    Compute attrition for each pair, according to the kind of antagonists.
      UF vs. NF
      NF vs. UF
    Accumulate the attrition to each force.
  Apply all attrition at the end.

Note that the outcome is independent of the order in which the pairs are processed, as the attrition is applied to each force after all attrition has been computed. This is standard for Lanchester attrition models: all the bullets are fired, and then they all hit at once.

9.9 Computing Attrition

9.9.1 Uniformed vs. Non-uniformed

Non-uniformed forces operate in small cells and hide among the civilian population. ROE and resources[38] permitting, a UF will attack an NF cell every time it gets a chance, and will do so with overwhelming force. Every cell found will be destroyed, and collateral damage to civilians is likely. There will be no UF casualties in this version as a result of UF attacks on NF cells.

The UF must find the NF cells in order to attack them. The number of cells found increases with:

- The number of troops in the UF: the more troops, the more chance of contact.
- Increased cooperation of the civilians with the UF, because the UF will get more intel.
- Decreased cooperation of the civilians with the NF, because the civilians are less likely to warn the NF of an impending attack.
- The number of troops in the NF: the more troops, the easier they are to find.
- In a later version: The exposure of the UF troops to the population (i.e., the UF units are assigned activities that imply contact with the locals); you can't find the bad guys while guarding the base.

Let

[38] Each force group has a cost per attack, set as part of the scenario by the analyst. The force group cannot attack without the required funds.

ΔT	=	The duration of the attrition interval in days
Ω_{NF}	=	The composite cooperation of the civilians in *n* with the NF.
Ω_{UF}	=	The composite cooperation of the civilians in *n* with the UF.
N_p	=	The possible number of attacks by the UF on the NF during the interval.
N_a	=	The actual number of attacks by the UF on the NF during the interval
cov_{UF}	=	The actual coverage fraction of the UF.[39]
$nomcov_{UF}$	=	The nominal coverage fraction of the UF for this algorithm.[40]
cov_{NF}	=	The coverage fraction of the NF.[41]
$nomcov_{NF}$	=	The nominal coverage fraction of the NF for this algorithm.[42]
TF	=	The average time to find an NF cell, given equal cooperation and the nominal coverage fractions.[43] This is the parameter that drives this part of the model.

Then, we compute the possible number of attacks as follows:

$$N_p = \text{round}\left(\frac{\Omega_{UF}}{\max(\Omega_{NF}, 10.0)} \times \frac{cov_{UF}}{nomcov_{UF}} \times \frac{cov_{NF}}{nomcov_{NF}} \times \frac{\Delta T}{TF}\right)$$

Thus, when the cooperation levels are balanced ($\Omega_{NF} = \Omega_{UF}$), and when the coverage fractions are at their nominal levels, we get precisely the average number of attacks during the interval:

$$N_p = \text{round}\left(\frac{\Delta T}{TF}\right)$$

The possible number of attacks is increased by increased cooperation of the population with the UF, and by increased coverage for either group, and is decreased by increased cooperation of the population with the NF, as desired.

Next, we must determine the actual number of attacks, and the attrition resulting from them. Let

cellsize	=	The average number of troops per NF cell.[44]

[39] Computed in-line using a coverage function; model parameter: `aam.UFvNF.UF.coverageFunction`

[40] Model parameter: `aam.UFvNF.UF.nominalCoverage` e.g., 0.3 (depends on what would be expected as average in this scenario)

[41] Computed in-line using a coverage function; model parameter: `aam.UFvNF.NF.coverageFunction`. This is identical to the default COERCION coverage function.

[42] Model parameter: `aam.UFvNF.NF.nominalCoverage`, e.g., 0.4

[43] Model parameter: `aam.UFvNF.UF.timeToFind`, e.g., 5 days

[44] For now, this is a model parameter, `aam.UFvNF.NF.cellSize`, e.g., 7 NF/cell in every neighborhood. Later, it

$personnel_{NF}$ = The number of NF troops in the neighborhood.

The number of cells available to attack is then

$$N_{cells} = \text{ceiling}\left(\frac{personnel_{NF}}{cellsize}\right)$$

Since each attack kills an entire cell, the actual number of attacks cannot exceed the number of cells; and in addition, the actual number of attacks cannot exceed the specified maximum:

$$N_a = \min(N_{max}, N_p, N_{cells})$$

The number of NF troops killed is then

$$N_{killed} = \min(N_a \times cellsize, personnel_{NF})$$

Next, we must compute civilian casualties. Let

$\hat{\Omega}_{UF}$ = The nominal composite cooperation of the neighborhood with the UF for this algorithm.[45]

$ECDA$ = The Expected Collateral Damage per Attack, i.e., the number of civilians killed for each attack on an NF cell, assuming nominal cooperation. This value will depend on the urbanization level of the neighborhood (rural, suburban, urban). For example, if *ECDA* is 2.0, then we expect two civilian casualties for each cell killed.[46]

Then

$$N_{civcas} = \text{floor}\left(N_a \times ECDA \times \frac{\hat{\Omega}_{UF}}{\max(\Omega_{UF}, 10.0)}\right)$$

If the actual cooperation of the neighborhood with the UF is exactly the nominal (that is, if $\Omega_{UF} = \hat{\Omega}_{UF}$) then the civilian casualties will be just as expected. If the nominal cooperation is 50%, then better than nominal cooperation can cut casualties in half—but minimal cooperation (10% or less) can increase casualties by a factor of five.

---

might be allowed to vary by group and by neighborhood.

[45] Model parameter: `aam.UFvsNF.UF.nominalCooperation`, e.g., 35 %

[46] Model parameter: `aam.UFvsNF.ECDA.`*`urbanization`*, e.g., 1 in rural, 3 in suburban, and 5 in urban.

9.9.2 Non-uniformed vs. Uniformed

An NF will attack a UF every time it can (given the ROE to attack), within the specified maximum number of attacks. Obviously, the NF can only attack if both the NF and the UF have personnel in the neighborhood. However, the number of such attacks is limited by the NF's resources[47], the availability of UF target opportunities, the cooperation of the civilian population with the NF, and the desire of the NF to limit their casualties as indicated by their ROE (HIT_AND_RUN or STAND_AND_FIGHT). In particular, the number of potential attacks should:

- Vary inversely with UF security
- Vary directly with the cooperation of the neighborhood with the NF.
- Vary directly with the coverage of UF units in the neighborhood.

First we compute N_p, the number of potential attacks:

$$N_p = \text{round}\left(\frac{100 - security_{UF}}{100} \times \frac{\Omega_{NF}}{\hat{\Omega}_{NF}} \times \frac{cov_{UF}}{nomcov_{UF}} \times rate \times \Delta T\right)$$

where

$rate$	=	The nominal attack rate per day.[48]
$security_{UF}$	=	The UF's security in the neighborhood. Security ranges from -100 to +100.
Ω_{NF}	=	The composite cooperation of the neighborhood with the NF.
$\hat{\Omega}_{NF}$	=	The nominal cooperation of the neighborhood with the NF for this algorithm.[49] This parameter depends on the ROE: HIT_AND_RUN or STAND_AND_FIGHT.
cov_{UF}	=	The actual coverage fraction of the UF, based on total personnel in the neighborhood.[50]
$nomcov_{UF}$	=	The nominal coverage fraction of the UF for this algorithm.[51] This parameter depends on the ROE: HIT_AND_RUN or STAND_AND_FIGHT.

[47] Each force group has a cost per attack, set as part of the scenario by the analyst. The force group cannot attack without the required funds.

[48] Ideally, the number of attacks should depend on the NF's resources, so that attacking the resources will reduce the number of attacks. Our notion is that the nominal number of attacks is determined by a Z-curve whose X-axis is a measure of the resources available to the NF. The currently envisioned model is that this measure is the weighted sum of the economic clout of the actors that have influence over the NF, weighted by the relationships between the NF and the actors. All this must wait until a later spiral, however. For now, the attack rate per day is part of the NF's attacking ROE.

[49] Model parameter: `aam.NFvsUF.`*`roe`*`.nominalCooperation`, e.g., 50%.

[50] Model parameter: `aam.NFvsUF.UF.coverageFunction`, e.g., {25 1000}.

ΔT = The duration of the attrition interval in days.

Now, the maximum number of attacks, N_{max}, can be thought of as the desired rate of attacks during one attrition interval. Thus, given

$$rate = \frac{N_{max}}{\Delta T}$$

the equation for N_p becomes

$$N_p = \text{round}\left(\frac{100 - security_{UF}}{100} \times \frac{\Omega_{NF}}{\hat{\Omega}_{NF}} \times \frac{cov_{UF}}{nomcov_{UF}} \times N_{max}\right)$$

If N_p is 0, then of course the NF cannot attack. Otherwise, whether the NF will actually attack or not depends on the expected NF casualties, as controlled by the ROE of HIT_AND_RUN or STAND_AND_FIGHT.

9.9.2.1 Loss Exchange Ratio

The loss exchange ratio (LER) for an attack is the number of NF casualties for each UF trooper killed. If the NF killed four UF personnel at a loss of one NF personnel, that would be an LER of ¼. AAM determines the loss exchange ratio (LER) as follows:

$ELER_{ROE}$	=	The Expected Loss Exchange Ratio: the expected number of NF casualties per UF casualty, when the UF fires back. This number depends on the NF's attacking ROE.[52]
Ω_{NF}	=	The composite cooperation of the neighborhood with the NF.
Ω_{UF}=	=	The composite cooperation of the neighborhood with the UF.
$MAXLER_{ROE}$	=	The maximum loss exchange ratio the NF is willing to accept when attacking with the specified ROE.[53]

Then

$$LER_{ROE} = ELER_{ROE} \times \frac{\Omega_{UF}}{\max(\Omega_{NF}, 10)}$$

In other words, the loss exchange ratio depends on the intelligence available to the NF and to the UF, as indicated by the cooperation of the neighborhood with each. As UF's intel improves,

[51] Model parameter: `aam.NFvsUF.UF.nominalCoverage`, e.g., 0.2

[52] Model parameter, `aam.NFvsUF.`*`roe`*`.ELER`, e.g., 0.33 for HIT_AND_RUN, 3.0 for STAND_AND_FIGHT.

[53] Model parameter, `aam.NFvsUF.`*`roe`*`.MAXLER`, e.g., 0.25 for HIT_AND_RUN, 4.0 for STAND_AND_FIGHT.

the LER gets larger; as NF's intel improves, the LER gets smaller. If the neighborhood cooperates equally with both, it's a wash and the LER is simply the expected LER.

The NF will only attack if the LER is their favor. That is,

If $LER_{ROE} \leq MAXLER_{ROE}$
Then attack,
Otherwise do not attack.

9.9.2.2 NF and UF Casualties

Hit-and-Run: When the ROE is HIT_AND_RUN, the NF will husband their forces, trying to do damage to the UF without losing too many people. We assume that the NF wants to inflict $UFCAS_{ATTACK}$ casualties on the UF during each attack.[54] Since they would not be attacking unless the Loss Exchange Ratio were in their favor, we know that they are prepared to take $NFCAS_{ATTACK}$ casualties in each attack, where

$$NFCAS_{ATTACK} = UFCAS_{ATTACK} \times LER_{ROE}$$

Stand-and-Fight: When the ROE is STAND_AND_FIGHT, on the other hand, the NF is prepared to suffer significant casualties in order kill UF personnel. We assume that the NF is willing to expend $NFCAS_{ATTACK}$ personnel to kill as many UF personnel as they can.[55] Given the LER, they can then kill

$$UFCAS_{ATTACK} = \frac{NFCAS_{ATTACK}}{\max(LER_{ROE}, 0.01)}$$

In either case, the number of NF casualties actually incurred depends on the Defending ROE of the UF: unless they fire back, no NF personnel will be killed. However, the NF must make their plans presuming that the UF will fire back.

Now, the NF can potentially make N_p attacks, given their access to the UF forces. The actual number of attacks is limited by the NF and UF personnel available. The NF cannot kill more UF personnel than are there, and will do so with the fewest casualties to themselves; and since they must presume that the UF will fire back they cannot schedule more attacks than they have personnel to lose. And, of course, they can make no more than the N_{max} attacks specified in their ROE. Thus, the actual number of attacks N_a is computed as follows.

[54] Model parameter, `aam.NFvsUF.HIT_AND_RUN.ufCasualties`, e.g., 4.
[55] Model parameter, `aam.NFvsUF.STAND_AND_FIGHT.nfCasualties`, e.g., 20

$$N_a = \text{floor}\left[\min\left(N_{max}, N_p, \frac{personnel_{NF}}{NFCAS_{ATTACK}}, \frac{personnel_{UF}}{UFCAS_{ATTACK}}\right)\right]$$

Given that an attack is possible, and the LER is in the NF's favor, the NF will **always** attack at least once.

The total number of UF casualties is then

$$UFCAS_{TOTAL} = \text{floor}[\min(N_a \times UFCAS_{ATTACK}, personnel_{UF})]$$

NF and civilian casualties depend on whether or not the UF fires back, as shown in the following table:

NF Attacks	UF Defends	UF Fires Back
HIT_AND_RUN	FIRE_BACK_IMMEDIATELY	Yes
HIT_AND_RUN	FIRE_BACK_IF_PRESSED	No
HIT_AND_RUN	HOLD_FIRE	No
STAND_AND_FIGHT	FIRE_BACK_IMMEDIATELY	Yes
STAND_AND_FIGHT	FIRE_BACK_IF_PRESSED	Yes
STAND_AND_FIGHT	HOLD_FIRE	No

If the UF fires back, then the number of NF casualties is

$$NFCAS_{TOTAL} = \text{floor}[\min(UFCAS_{TOTAL} \times LER_{ROE}, personnel_{NF})]$$

Otherwise, no NF casualties are incurred.

9.9.2.3 Civilian Collateral Damage

In this version we assume no collateral damage from the NF attack itself—NF does not want to kill civilians in these attacks, but would be glad if UF fired back and did kill some. Thus, civilian casualties occur only if the UF fires back. In this case, the total number of civilian casualties is

$$CIVCAS_{total} = ECDC \times NFCAS_{TOTAL}$$

where

$ECDC$	=	the Expected Collateral Damage per NF Casualty. This is a model parameter which depends on the urbanization level

(urban, suburban, or rural) of the neighborhood, and on the NF's ROE.[56]

9.10 Applying Attrition

All attrition is computed before any attrition is applied to the neighborhood. Attrition to a group is applied to each of a group's units in the neighborhood in proportion to its size. The attrition applied to each unit is also applied to the unit's group's total personnel. Civilian attrition is saved for later assessment by the DAM rule sets.
First, we build a list of the units in the neighborhood that belong to the group being attrited. (Note that civilian units from other neighborhoods are treated as belonging to distinct groups.)

Next, we compute the total number of personnel in the list of units.

Next, we compute the fraction of the total represented by each of the units.

Next, we sort the list of units in decreasing order of size.

Next, we apply attrition to each unit in turn, attriting it by its proportional share of the casualties. Let

casualties	=	The total number of casualties to inflict
i	=	The index of the i^{th} unit to receive casualties
f_i	=	The fraction of casualties to be taken by i.

Then,

Let r = *casualties*
For each unit i to be attrited,
    Let $k = \min[r, \text{ceiling}(f_i \times casualties)]$
    Apply k casualties to r.
    Let $r = r - k$
    If r is 0, then stop.

This algorithm rounds fractional casualties in favor of the smaller units; the smallest unit will tend to get less than its "fair" share of casualties.

[56] Model parameter: `aam.NFvsUF.ECDC.`*`urbanization`*, e.g., 0.1 for RURAL, 0.15 for SUBURBAN, 0.2 for URBAN

9.11 Assessing the Attitude Implications

Once attrition has been computed and applied, it is necessary to assess the implications for civilian attitudes via the CIVCAS rule set in the Driver Assessment Model (DAM).

9.11.1 Contrasted with JNEM

In theory, JNEM assesses civilian casualties incident by incident—in theory, because it is the ground model's responsibility to decide what constitutes an incident, and some ground models do a better job than others. In consequence, JNEM accumulates all attrition to a group occurring within a short window, and calls that an "incident". From this attrition, JNEM computes the effective number of kills, and passes this through a Z-curve to get a multiplier which is used to scale the magnitudes in the CIVCAS rules.

In general, then, attrition happens when it happens, and the attitude effects of different incidents can interfere with each other (based on sharing the same cause) as they play out over time.
In Athena, we assess attrition periodically, nominally once a week, reflecting the incidents that have implicitly occurred over the previous week. So long as we approach attrition in this aggregate way, there's no way to assess it incident by incident. It is clearly wrong to say, "There were 17 fire fights this week in which Punjabis were killed in this neighborhood, so we'll have 17 inputs to GRAM all time-stamped today and all with the same cause." Consequently, we will assess the attitude implications once per week as well, based on the aggregate attrition over the week.

Magic attrition to civilian groups will be applied to those groups as it occurs, but the total attrition will be saved and assessed with the normal attrition at the end of the week.

9.11.2 Satisfaction Effects of Attrition

We will assess the satisfaction effects of attrition on a civilian group as follows. First, all attrition occurring during the week, both magic and normal, to group *f* in neighborhood *n* will be accumulated in the **attrit_nf** table. Let

n	=	The neighborhood. For civilian units, this is the neighborhood of origin, rather than the neighborhood in which the attrition occurred. (They will often be the same, of course.)
f	=	The attrited civilian group.
casualties	=	The total number of casualties to *f* in *n* during the week.
ZSAT()	=	A Z-curve which converts a total number of casualties into a casualty multiplier used in the CIVCAS satisfaction rules.[57]

[57]Model parameters: `dam.CIVCAS.Zsat`

M	=	The casualty multiplier.

We compute the casualty multiplier, M, as follows:

$$M = ZSAT(casualties)$$

When the CIVCAS rule set is triggered, it has access to n, f, *casualties*, and *M.*

9.11.3 Cooperation Effects of Attrition

Athena only tracks the cooperation of neighborhood groups with force groups; consequently, we assess cooperation effects only for attrition to civilian groups. The CIVCAS rule set attends to this.

All normal civilian attrition is (at present) due to collateral damage resulting from fighting between two force groups. Magic attrition can optionally be attributed to one or two force groups. As attrition occurs, Athena accumulates the total casualties to group f in neighborhood n in which force group g was in some way involved in the **attrit_nfg** table. If two force groups are involved in an altercation, as is usually the case, the total civilian casualties are attributed equally to both.

Then, let

n	=	The neighborhood of origin, rather than the neighborhood in which the attrition occurred. (They will often be the same, of course.)
f	=	The attrited civilian group, resident in n.
g	=	A force group.
casualties	=	The total number of casualties to f in n during the past week, in which g was involved.
R_{fg}	=	The relationship between civilian group f and force group g.
$ZCOOP()$	=	A Z-curve which converts a total number of casualties into a casualty multiplier used in the CIVCAS cooperation rules.[58]
M	=	The resulting casualty multiplier.

We compute the casualty multiplier, M, as follows:

$$M = ZCOOP(casualties)$$

When the CIVCAS rule set is triggered it has access to n, f, g, *casualties*, R_{fg}, and *M.* The actual magnitude of the rule firing will be

[58]Model parameter: `dam.CIVCAS.Zcoop`

$$magnitude = M \times \text{enmore}\left(R_{fg}\right) \times \text{M-}$$

where M- connotes a medium-sized negative (i.e., -5.0 point) effect and enmore(R) is the "enemies more" relationship multiplier function. See the *Mars Analyst’s Guide* for more information on relationship multiplier functions, and the *Athena Rules* document for details on how rule magnitudes are specified.

10. DEMOGRAPHICS

The Athena Demographics model models the number of people in the civilian population broken down in a variety of ways. In the long run, this model will handle births, deaths due to old age and other causes, and aging of the population; in this version, it tracks the following:

- The base population, by civilian group broken down into two groups: those who support themselves by means of subsistence agriculture and barter (the subsistence population), and those who participate in the regional economy (also known as consumers).
- The number of deaths due to attrition for each neighborhood group.
- Resident population: people in their neighborhood of origin.
- Displaced population: people assigned activities outside of their neighborhood of origin.[59]

And, taking all of these into account,

- The total number of consumers in each neighborhood.
- The total labor force present in a neighborhood, including both resident and displaced population present in the neighborhood. In computing the labor force, we assume that the subsistence population is not included in the job market.

10.1 Requirements for This Version

The demographics model is intended to be as simple as possible while meeting the needs of the other Athena models. In particular:

- GRAM requires the current population of each civilian group.
- The Ground model requires that the civilian population of a neighborhood can be assigned activities in other neighborhoods.
- The Athena Attrition Model (AAM) requires that the civilian population can take collateral damage as the result of combat between forces.

[59]This term is somewhat ambiguous. Civilians can be assigned the DISPLACED activity while remaining in their own neighborhood; in this case, they are considered "resident" rather than "displaced".

- The Economics Model requires the number of people who participate in the regional economy (the consumers) and the number of people in the potential labor force (the workers).

Consequently,

- The playbox population is input to the scenario as the initial population of each civilian group.

- Each civilian group has a percentage of its population (possibly zero) that supports itself by subsistence agriculture and therefore does not participate in the regional economy. This is the subsistence population. The consumers are the remainder of the population.[60]

- Resident population can be displaced; that is, it can be assigned activities outside of its neighborhood of origin.

    - The reason for the displacement can be indicated by the choice of activity, e.g., DISPLACED or IN_CAMPS; however, this is outside the scope of DEMOG.

- The labor force is a fraction of the consumers in the neighborhood, disregarding those displaced personnel who are not in a position to work. (I.e., those in camps, or new in the neighborhood).

10.2 Simplifying Assumptions

We make the following simplifying assumptions:

- We do not track births, or deaths from causes other than attrition.

- The population does not age.

- The subsistence population is a simple percentage of the non-displaced population of each neighborhood group.

- The subsistence population is outside the regional cash economy.

[60] In reality, of course, the dividing line between subsistence and consumption will not be so stark; subsistence farmers will often buy or trade for certain goods, and some consumers may also grow crops or animals for their own use. As a first approximation, however, this suffices.

- Displaced personnel are displaced from their land, and clearly cannot be doing subsistence agriculture.[61]

- The labor force is a simple fraction of the total consumers, taking civilian activities (e.g., refugee status) into account.

10.3 Population and Units

Athena 1 had the distinction between explicit population (represented in units) and implicit population (the default). In Athena 3, units are an output rather than an input: the ground model "staffs" units based on resident populations and deployed forces, both for the purpose of visualization and to ease subsequent computations. As a result, every civilian is represented both implicitly and explicitly, and the distinction is no longer of interest.

10.4 Civilian Group Population

The resident population of civilian group g at the current simulation time is

$$population_g = BP_g - DP_g - attrition_g$$

where

BP_g	=	The base population of group g at time zero, as defined in the scenario.
DP_g	=	The displaced population: the total number of personnel belonging to group g and assigned activities in some other neighborhood.
$attrition_g$	=	The total number of casualties suffered by group g since time zero.

10.4.1 Civilian Attrition

In Athena 3, all attrition occurs to personnel in units created by the ground model's staffing algorithm, but the actual attrition is applied to the pool of people from which the units are staffed. For civilians, consequently, attrition to a civilian group's population is handled by adding the casualties to $attrition_g$:

$$attrition_g = attrition_g + casualties$$

[61] Nomadic herdsmen are clearly a separate case, and one that must wait until we implement migration fully. Consequently, any subsistence herdsman and assumed to stay within their neighborhoods of origin (for now).

10.4.2 Subsistence Population

The subsistence population of a neighborhood group, SP_g, is a fraction of the total resident population:

$$SP_g = \frac{SAP_g}{100.0} * population_g$$

where

SAP_g	=	The Subsistence Agriculture Percentage for neighborhood group g.

Note that displaced personnel are *never* part of the subsistence population. This opens a slight hole in the model: if the displaced personnel are brought back to their neighborhood of origin, some of its personnel might once again be counted as part of the subsistence population. This is unrealistic—people who leave subsistence agriculture are rarely able to go back, as they generally have lost their land and livestock.

10.4.3 Consumer Population

The consumer population of a civilian group is simply that part of the resident population that participates in the regional economy, i.e., that are not members of the subsistence population. Thus, we define civilian group g's consumer population CP_g as follows:

$$CP_g = population_g - SP_g$$

We will account for consumers in the group's displaced population at the neighborhood level, rather than as part of a neighborhood group.

10.4.4 Labor Force

Civilians contribute to the labor force to different degrees, depending on their assigned activities, and in addition, we must exclude the subsistence population. Since the entire civilian population is now represented in units, we define the contribution to the labor force[62] of group g as follows:

$$LF_{ng} = \frac{100.0 - SAP_g}{100.0} \times \sum_{\substack{u \in g \\ u \in n}} [LFF(a_u) \cdot personnel_u]$$

[62]By "labor force" we mean that portion of the population that seeks to be employed, whether they are in fact employed or not.

where

SAP_g	=	The subsistence agriculture percentage for group g.
n	=	The neighborhood in which group g resides.
u	=	A non-displaced unit belonging to group g, and consequently located in n.
a_u	=	The activity assigned to unit u.
$personnel_u$	=	The number of personnel in unit u.
$LFF(a_u)$	=	The labor force fraction for activity a.

Note that we will account for displaced workers at the neighborhood level, rather than as part of a neighborhood group.

The Labor Force Fractions are model parameters[63], changeable at run-time; the default values are as follows:

Unit Activity	**Labor Force Fraction**
NONE	0.6
DISPLACED	0.4
IN_CAMP	0.0

The assumption is that DISPLACED units, which are mingled with the host population, look for jobs at a decreased rate due to their unsettled condition. As DISPLACED units assimilate, they can be given an activity of NONE. Further, we assume that personnel in refugee camps have no opportunity to look for work. Again, if camps turn into permanent settlements the unit activity can be changed.

10.5 Neighborhood Population

Neighborhood figures are the total of the civilian group figures for the groups resident in the neighborhood, plus the total of the figures for units displaced from their neighborhoods of origin. Thus, we must first determine the figures for displaced units.

10.5.1 Displaced Personnel

The total personnel in units displaced from their neighborhood of origin is simply

$$DP_n = \sum_{\substack{u \in n \\ u \in \text{displaced}}} personnel_u$$

[63] `demog.laborForceFraction.`*`activity`*

where

u	=	A unit in neighborhood n, but displaced from its neighborhood of origin.
$personnel_u$	=	The number of personnel in unit u.

10.5.2 Displaced Consumers

To be displaced is to be displaced from one's land, crops, and livestock; hence, all displaced personnel must willy-nilly participate in the regional economy. Thus, all displaced personnel are consumers.

10.5.3 Displaced Labor Force

The displaced labor force in neighborhood n, DLF_n, is computed much like the labor force for a neighborhood group:

$$DLF_n = \sum_{\substack{u \in n \\ u \in \text{displaced}}} LFF(a_u) \times personnel_u$$

10.5.4 Neighborhood Totals

The total population of neighborhood n is simply the population of all resident civilian groups, plus all displaced personnel:

$$population_n = DP_n + \sum_{g \in n} population_g$$

The consumer population and labor force are computed similarly:

$$CP_n = DP_n + \sum_{g \in n} CP_g$$

$$LF_n = DLF_n + \sum_{g \in n} LF_g$$

10.6 Regional Population

The regional population, consumers, and labor force are simply summed up across the "local" neighborhoods in the obvious way.

10.7 Unemployment

Unemployment can drive attitude change. The Economics Model computes the unemployment rate, UR, for the region of interest.

10.7.1 Disaggregation to Neighborhoods

Lacking any better way to disaggregate unemployment, we will assume that UR affects each neighborhood in proportion to its labor force. That is, if UR is 5%, then 5% of each neighborhood's labor force will be unemployed. The unemployed population for the neighborhood is then UP_n:

$$UP_n = LF_n \times \frac{UR}{100.0}$$

where

LF_n = The number of people in the labor force in neighborhood n.

However, the size of the labor force relative to the neighborhood population as a whole depends on the subsistence population of each neighborhood group. A high unemployment rate may be of little concern in a neighborhood with 90% subsistence. For attitude effects, we are primarily concerned with the ratio of unemployed workers with the total population: the unemployed workers per capita, or UPC_n:

$$UPC_n = 100.0 \times \frac{UP_n}{population_n}$$

10.7.2 Disaggregation to Civilian Groups

Precisely the same logic applies to each group in each neighborhood. The unemployed population for civilian group g is UP_g:

$$UP_g = LF_g \times \frac{UR}{100.0}$$

where

LF_g = The number of people from group g in the labor force.

And then, the unemployed per capita for each group g is:

$$UPC_g = 100.0 \times \frac{UP_g}{population_g}$$

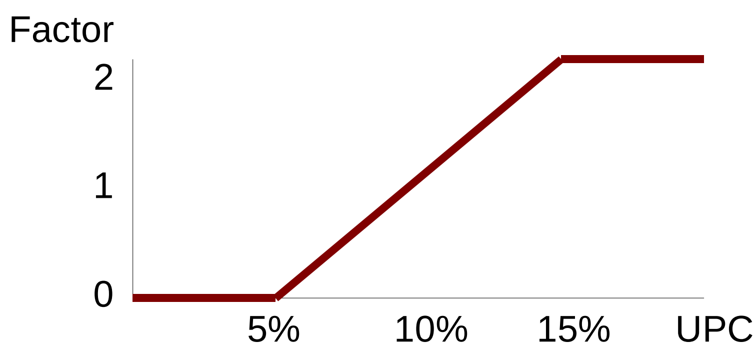
10.7.3 Unemployment Situations

How does unemployment affect the civilians?

- The unemployed civilians take a hit to their satisfaction levels, with the usual indirect effects on other groups. This will particularly affect Quality of Life (QOL), and should depend on each neighborhood group's effective unemployment rate.

- If there are many unemployed civilians, the fact of their presence will become a problem for their neighbors due to increases in crime and other social conflict. This effect will be negligible for small unemployment rates, and dominant for large ones (for some value of "small" and "large"). This will particularly affect Safety (SFT), and possibly Autonomy (AUT), and should depend on the effective unemployment rate of the neighborhood as a whole.

If unemployment in a neighborhood is sufficiently dire, it will engender an UNEMP (unemployment) demographic situation. Following the pattern used for activity situations, we compute a factor, analogous to a coverage fraction, that will be used to scale the magnitudes in the situation's rule set. We will use a Z-curve to convert the unemployment per capita into a multiplicative factor, ranging from 0.0 to 2.0:



The above curve says that the unemployment rules will not begin to affect general attitudes until about 5% of the general population are unemployed; the rules reach their nominal magnitude when 10% of the population are unemployed; and the rules have their maximum effect when 15% are unemployed. This is not to say that problems do not continue to get worse when more than 15% of the population is unemployed; but very high unemployment brings other problems that can't simply be addressed by an attitude rule set.

Using the above curve, we will compute an unemployment attitude factor (UAF) for the neighborhood as a whole, and for each individual group within it:

$$UAF_n = Z_{UAF}(UPC_n)$$

And

$$UAF_g = Z_{UAF}(UPC_g)$$

Then, an unemployment situation will exist in neighborhood n if $UAF_n > 0.0$, or if $UAF_g > 0.0$ for any g. The situation is assessed by the UNEMP rule set; see the *Athena Rules* document for details.

We assume that a civilian group's QOL is affected primarily by unemployment within the group, but that SFT and AUT are primarily affected by the number of unemployed workers in the neighborhood as a whole. For example, one would expect the crime rate to rise with the number of unemployed workers.

11. ECONOMICS

The Athena Economics area models the economy of the region of interest, which can be an entire country, a portion of a country, or several small countries taken together. We will refer to this as the *local economy*. The core of the Economics area is a 3-sector Computable General Equilibrium (CGE) model solved as a system of non-linear equations using the Gauss-Seidel algorithm. This document will give a brief overview of the Economics area, and then explain how the CGE is embedded in it and how it relates to the rest of Athena. The CGE itself is implemented as a "cell model"; the cell model code is included in Section 12.

11.1 Sectors

The CGE partitions the local economy into three sectors: **goods**, **pop**, and **else**.

The "goods" Sector: The **goods** sector includes all production of goods and services in the local region. The unit of production is the *goods basket* (GBasket), a notional basket of goods and services nominally costing about $1.

The "pop" Sector: The **pop** sector includes all labor by the workers in the local region, and all consumption by the population of the local region. The unit of production is the work-year of an *average worker*. Just as the goods basket represents a notional bundle of goods and services, the average worker represents a notional bundle of skills and kinds of work.

The "else" Sector: The **else** sector represents everything in the local economy that isn't covered by the other two sectors, as well as the entire rest of the world. In practice, the "rest of the world" means imports and exports. The unit of production is the *else basket* (EBasket), which is similar to the goods basket and also nominally costs about $1.

11.2 The Economic Tableau

ECON displays the current state of the economy in a spreadsheet-like tableau:

	goods	**pop**	**else**	**Revenue**	**Price**	**Quantity**	**Units**
goods	X_{gg}	X_{gp}	X_{ge}	REV_g	P_g	QD_g	GBasket/yr
pop	X_{pg}	X_{pp}	X_{pe}	REV_p	P_p	QD_p	work-year/yr
else	X_{eg}	X_{ep}	X_{ee}	REV_e	P_e	QD_e	EBasket/yr
Expense	EXP_g	EXP_p	EXP_e				

where

X_{ij} = The payment in $/year from sector *j* to sector *i*.
EXP_j = The total expenditure, in $/year, of sector *j*.

REV_i	=	The total revenue, in $/year, of sector *j*.
P_i	=	The price, in $, of one unit of production of sector *i*.
QD_i	=	The quantity of *i*'s product demanded per year.

In short, the CGE determines how much each sector produces, what each unit of production costs, the revenue and expenditure for each sector, and (from these) the Consumer Price Index (CPI), the Gross Domestic Product (GDP), and the Deflated Gross Domestic Product (DGDP).

The following equations are true, either by definition or whenever the economy is in equilibrium:

$$EXP_j = \sum_i X_{ij}$$

$$REV_i = \sum_j X_{ij}$$

$$REV_i = P_i \times QD_i$$

$$REV_i = EXP_i$$

11.3 Text Notation

The CGE itself is implemented as a *cell model*, a pseudo-spreadsheet in which every cell has a name rather than a row and column. The cell model is contained in a text file, and so we also have a plain text notation for each of the mathematical symbols given above:

X_{ij}	=	`X.i.j`
EXP_j	=	`EXP.j`
REV_i	=	`REV.i`
P_i	=	`P.i`
QD_i	=	`QD.i`

When we are referring to a particular sector or sectors, the sector names are spelled out in full in the plain text form:

X_{gp}	=	`X.goods.pop`

11.4 Shape vs. Size

We distinguish between the *size* of the economy, which can be roughly thought of as total revenues, and the *shape* of the economy, or the proportion of revenues across the sectors. The size of the economy is driven by consumption: increase the number of consumers, or the amount consumed by each, and the economy must increase in size. Decrease the amount of consumption, and the economy must shrink. But as the economy increases and decreases in size, its basic shape remains the same, because the basic industries and technologies in use remain the same.

The size is driven by the number of consumers; and the number of consumers is given to the Economic model by the Demographic model. The shape of the economy is determined when the economy is calibrated; see Section 11.6.

11.5 Production Functions

The shape of the economy is largely determined by the technologies used by the sectors to produce their products; and these technologies are described by *production functions*.

A production function determines how much product a sector consumes (from its own and other sectors) to produce one unit of its own product. All three sectors are modeled using the Cobb-Douglas production function, which implies that, when ingredient quantities are chosen to minimize costs, the sector will spend a fixed proportion of its money on each of the three sectors, i.e., 0.5 on **goods**, 0.4 on **pop**, and 0.1 on **else**.[64] As prices change, the Cobb-Douglas production function allows the sectors to trade off their requirements for the product of one sector for the product of another.

These fixed proportions are called the Cobb-Douglas parameters, and are denoted f_{ij}.

11.6 Calibrating the CGE

The CGE is calibrated by setting a number of model parameters:

- `econ.f.i.j`, the Cobb-Douglas parameters for the three sectors.
- `econ.BaseWage`, the average wage for one work-year, in dollars.
- `econ.GBasketPerCapita`, the average consumption of **goods** by each consumer per year, in goods baskets.

In theory we should also set the base prices for the **goods** and **else** sectors; but as the unit of production for these sectors is an arbitrarily-sized basket worth nominally $1, the base price for these sectors is naturally $1.

[64] Another choice is the Leontief production function, which says that each unit produced requires a fixed amount of the products of the other sectors, e.g., 3 cups of flour, 2 eggs, and so forth. The Leontief production function provides a better model of production requirements for some sectors in real economies, and it is likely that we will use it in the future.

Athena has preset values for these parameters. If desired, the analyst can choose other values based on the economy of the region of interest, as described in the following subsections. These parameter values should be set during Scenario Preparation, before the scenario is locked and enters time 0.

11.6.1 Fill in the Social Accounting Matrix

The Social Accounting Matrix, or SAM, is the upper left portion of the economic tableau:

	goods	**pop**	**else**	**Revenue**
goods	BX_{gg}	BX_{gp}	BX_{ge}	$BREV_g$
pop	BX_{pg}	BX_{pp}	BX_{pe}	$BREV_p$
else	BX_{eg}	BX_{ep}	BX_{ee}	$BREV_e$
Expense	$BEXP_g$	$BEXP_p$	$BEXP_e$	

The analyst must determine, from whatever sources, the base flow of money to each sector from each sector (the BX_{ij}'s), in dollars,[65] and compute the base revenues and expenses: the row and column sums, $BREV_i$and $BEXP_j$. In a typical SAM, the revenues for each sector will equal the expenses.

11.6.2 Compute the Cobb-Douglas Parameters

Given the base SAM and the Cobb-Douglas production assumptions, the analyst can compute the Cobb-Douglas parameters. Let

$$f_{ij} = \frac{BX_{ij}}{BEXP_j}$$

Note that the f_{ij}'s sum to 1.0 down each column; hence, of the nine values we only need to enter six model parameters:

```
econ.f.goods.goods
econ.f.goods.pop
econ.f.goods.else
econ.f.pop.goods
econ.f.pop.pop
econ.f.pop.else
```

[65] Athena's concept of dollars is somewhat notional, and certainly isn't tied to the buying power of a real American dollar. The analyst can actually use any monetary units he chooses, provided he is consistent.

11.6.3 Set the Base Wage and Consumption

Next we need to provide the link between shape of the economy and its size. We do this by setting the base wage and the consumption rate.

The base wage is the average wage in dollars for one work-year of work, where work-year is defined as in Section 11.1. The model parameter is `econ.BaseWage`. This determines each worker's purchasing power.

Then, we set the consumption of goods baskets (GBasket) per capita per year: how much **goods** each consumer consumes on average. This figure will usually be rather less than the average wage, because each basket initially costs one dollar, consumers consume **pop** and **else** as well as **goods**, and not all consumers are workers. The model parameter is `econ.GBasketPerCapita`.

11.7 Scenario Inputs

The following input values are plugged into the CGE by the ECON model as part of the base scenario:

Cell	Source	Description
`BaseConsumers`	DEMOG	The total number of consumers in the local region at time 0. This number is used to size the economy, and calibrates a number of constants used subsequently.

11.8 Run-time Inputs

The following input values are plugged into the CGE by the ECON model at each "tock", that is, at each update of the CGE as time passes:

Cell	**Source**	**Description**
`In::Consumers`	DEMOG	The number of consumers in the local region at the current time. This number drives consumption, which determines the size of the unconstrained economy.
`In::WF`	DEMOG	The number of workers in the local economy at the current time. This number determines the production constraint for the **pop** sector, and also drives the computation of unemployment.

Cell	Source	Description
`In::LSF`	ECON, Ground	The Labor Security Factor, a number from 0.0 to 1.0. The LSF decreases with neighborhood security; (1-LSF) is the fraction of the work force that stays home from work out of fear for their lives. Thus, this number also affects the production constraint for the **pop** sector.
`In::CAP.goods`	Ground	Each neighborhood can produce a certain quantity of goods; this quantity is calibrated at time 0 based on the size of the labor force in that neighborhood. The maximum possible production is the capacity of the economy to produce goods, which constrains the size of the economy. The contribution of each neighborhood to `CAP.goods` can be increased or decreased using the neighborhood's Production Capacity Factor (PCF).

11.9 Outputs

The CGE produces the following output values.

Cell	Used By	Description
`Out::P.`*`i`*	Display	The price of one unit of sector *i*, in dollars.
`Out::QS.`*`i`*	Display	The quantity supplied for sector i, i.e., the number of units produced.
`Out::REV.`*`i`*	Display	The revenue of sector *i*, that is, `Out::P`*`i`*`*Out::QS.`*`i`* in dollars.
`Out::EXP.`*`i`*	Display	The expense of sector *i*, that is, the dollars spent on the ingredients for the product of sector *i*.
`Out::QD.`*`i.j`*	Display	The quantity of sector *j*'s output purchased by sector *i*.
`Out::X.`*`i.j`*	Display	The dollars spent by sector *i* on sector *j*'s output.
`Out::LATENTDEMAND.`*`i`*	Display	The additional quantity of sector *i*'s product that the economy would purchase if it could be produced. (**goods** and **pop** only)
`Out::IDLECAP.`*`i`*	Display	The additional quantity of sector *i*'s product that the sector could produce if only there were demand for it. (**goods** and **pop** only)

Cell	Used By	Description
`Out::Unemployment`	Display	The number of workers who are currently unemployed, including normal turbulence.
`Out::UR`	DEMOG	The unemployment rate, as a percentage.
`Out::GDP`	Display	The Gross Domestic Product, in dollars: the total revenue of the economy, excluding the **else** sector.
`Out::CPI`	Display	The Consumer Price Index, which measures changes in buying power since the start of the simulation.
`Out::DGDP`	Display	The Deflated Gross Domestic Product, i.e., the GDP divided by the CPI. This is the current size of the economy, in "time 0" dollars.

11.10 Ways to Affect the Economy

The Economy is affected at each economic tock by the inputs listed in Section 11.8. Consequently, the following things taking place in Athena as a whole will affect the economy:

- Civilian casualties can decrease the number of consumers and workers.
- Subsistence population, when displaced from their land, willy-nilly become consumers; they might not be able to contribute to the work force, depending on their assigned activity.
- When a civilian group's security in a neighborhood decreases, workers stay home out of fear, thus reducing the effective size of the work force. This is measured by the Labor Security Factor (LSF).
- Each neighborhood's Production Capacity Factor can be increased or decreased, reflecting building of new plant or destruction of existing plant with the consequent effect on `CAP.goods`.

11.11 Ways the Economy Affects Athena

There are many ways in which the economy *should* affect Athena; at present, the only implemented effect is that of unemployment on the civilian population. This is done in the Demographic model. See Section 10.

11.12 CGE Architecture

The CGE equations are implemented as a *cell model*[66]; the cell model is given in Section 12. A cell model is like a non-GUI spreadsheet model, in which each each cell has a name rather than row and column indices. Cells can contain constants or formulas; the application can plug in input values by setting the values of constant cells, and can read outputs by retrieving the value of formula cells. The model is broken up into a number of pages, each with a well-defined purpose:

- The **null** page contains calibration constants, and values computed from them, that are used in later pages. The values on the **null** page never change after the CGE is calibrated.

- The **Cal** page contains a system of non-linear equations which are iterated to a solution using the Gauss-Seidel algorithm when the CGE is calibrated. During this step a number of values are computed which are used as the basis for the evolution of the CGE over time.

- The **In** page contains all inputs to the CGE that vary as the time progresses. Athena updates these values prior to every economic tock.

- The **U** page defines the unconstrained model of the economy over time. It inherits most of its equations from **Cal**, with the necessary changes to use **Cal**'s outputs and **In**'s inputs. It computes the economy as it would be if there were no labor or goods capacity constraints.

- The **C** page inherits the equations from **U**, and modifies them to apply labor and goods capacity constraints. Starting with the outputs from **U**, and based on the Cobb-Douglas assumptions, it adjusts the prices and quantities for **goods** and **else** such that the constraints are not exceeded.

- The **Out** page contains all outputs used or displayed by Athena. In addition to copying many of **C**'s outputs directly, it also computes the unemployment rate, `Out::UR`, the CPI, `Out::CPI`, and the Deflated Gross Domestic Product, `Out::DGDP`.

The CGE itself is documented in Section 12.

[66] See the cellmodel(5) man page for a complete description of cell model syntax and semantics.

Appendices

12. THE ECONOMICS CELL MODEL

This section defines the Economics Model's Computable General Equilibrium (CGE) model in cellmodel(5) format. The cellmodel(5) format and tools are defined by the Mars Simulation Infrastructure Library.

```
# -*-Tcl-*-
#-----------------------------------------------------------------------
# TITLE:
#    eco3x3.cm, version r
#
# AUTHOR:
#    Bob Chamberlain
#    Will Duquette
#
# DESCRIPTION:
#    Prototype CGE for the Athena Economics Model.  This is a
#    3x3 Cobb-Douglas model, based on Ian Sue Wing's MIT paper.
#
#    Criteria for success:
#
#    * The model converges.
#    * All REV.i = EXP.i
#    * Prices and quantities are reasonable
#      * Prices should recover the values used to calibrate the
#        constants A.i.
#    * Quantities produced should equal the sum of the demands.
#      * I.e., deltaQ.i should = 0 for all i.
#    * Quantities should not all be zero.
#
# PAGES:
#    The model contains the following pages.  We expect the whole
#    model to be computed at time 0, as part of calibration; as time
#    advances, the model will be recomputed periodically starting at
#    page In.
#
#    "null" Basic inputs, including the Base Case and possibly SAM
#           data, and non-iterative calibration.
#    Cal    Iterative calibration based on data from the null page.
#    In     Application-settable inputs for the U and C pages.
#    U      Unconstrained model; the size of the economy is driven by
#           consumer demand.
#    C      Constrained model; the size of the economy is constrained
#           by the production capacity and the labor supply, both of
#           which are supplied by the rest of Athena.
#    Out    Output page: computes overages, shortages, and idle capacity
#           (e.g., unemployment) by comparing the constrained and
#           unconstrained results, as well as other outputs to the
#           rest of Athena.
#
#    NOTE: The Athena application is aware only of the "null", In, and Out
#    pages.
#
```

```
# HISTORY:
#    Version r: Revised the formulas for C::QS.goods and C::QS.pop so that
#               in the absence of binding constraints they will be at least
#               as big as on the U page.
#
#               Added a number of diagnostic cells to the Out page, to
#               support the application's sanity check of the results.
#
#    Version q: Added CSF and LSF, so that low security can decrease
#               both consumption and the labor supply.
#               Added ediff() to the deltaQ.i and deltaREV.i formulas,
#               so that it's easier to see whether they are met or not.
#               Copied the values of deltaQ.i and deltaREV.i to the
#               Out page, in case we want the application to keep
#               an eye on them.  Note that CSF has no affect at this time.
#
#               Added a new command to initialize the C page from the
#               U page each time C is solved.  The C page does not
#               stand alone; it is explicitly an adjustment of the
#               unconstrained result.
#
#    Version p: Corrected computation of CPI on page C to be based on
#               consumer purchases rather than all purchases.  Removed
#               CAP.else once again, as it is no longer needed.
#               Renamed the names BasePopulation and In::population
#               with the correct names BaseConsumers and
#               In::Consumers.  Added computation of GDP.
#
#    Version o: Added capacity limits for the else sector to facilitate
#               further testing.  Deleted the "deltaP" parameters, as
#               they aren't needed.  Other cosmetic changes.
#
#    Version n: Corrected the price adjustment equations so they
#               correctly use the optimal Cobb-Douglas demands to
#               adjust prices so that demand equals supply (i.e.,
#               markets clear).  This is justified by the assumption
#               that there are no economies of scale in any sector.
#               Changed numeraire for the price equations on the C
#               page to U::P.pop, the average wage in the
#               unconstrained case.  Revised QD.goods.pop to maintain
#               f.goods.pop under reduced production.
#
#    Version m: The price adjustments in version l affect the CPI, so
#               the P.goods implied by the base case CPI, U::P.goods,
#               is used as the numeraire instead of the CPI.
#
#    Version l: Adjusts prices when production constraints are encountered
#               on page C until demand reduces enough to achieve market
#               clearance.  Based on version j, not on version k.
#
#    Version k: Attempted to achieve market clearance when production is
#               constrained by simply limiting QD.i.j.  Didn't work;
#               some QD.i.j went negative during iteration.
#               Constraining QD.i.j >= 0.0 still didn't work.  Attempt
#               abandoned for further analysis.
#
```

```
#    Version j: Replaced [Cal::SUM] in U::P.goods with [SUM]; [SUM]
#               is now computed as part of pages U and C.  In addition,
#               SHORTAGE.i is really the latent demand for the product
#               of section i, so it is renamed LATENTDEMAND.i.  OVERAGE.i
#               turns out not to be useful, so it has been removed.
#
#    Version i: Added computation of labor market turbulence as a
#               percentage of the labor force.  Added explicit
#               computation of the non-turbulent, "real",
#               unemployment.  Made minor cosmetic improvements.
#
#    Version h: Re-integration of version g into Athena.
#               * Added Out::BQS.goods = Cal::QS.goods, to support the
#                 CAP.goods calibration.
#
#    Version g: Re-ordered the equations per RGC.
#
#    Version f: Restored the notion that the work force should be
#               endogenous by calibrating the per capita demands on
#               the null page, then assuming jobs will be driven by the
#               demand for labor, but limited by the possibly changing
#               workforce statistics (population, available work force).
#
#    Version e: Made the CPI the numeraire.  Assumed the demographics
#               model will compute both CAP.pop and QS.pop, i.e., the
#               demographics model owns the unemployment rate.
#
#               WHD: It's reasonable that Demographics should compute
#               the size of the workforce given wages and other
#               opportunities.  But the CGE must compute the number
#               of people who actually *can* work given production
#               constraints.  (This assumes that idle goods production
#               capacity, if any, can be put to work as demand rises
#               in the equilibrium economy.)
#
#               WHD: In this version, the cal page and the U page could
#               be merged; however, I'm going to leave them be.
#
#    Version d: Revised P.pop and A.pop: per RGC, they are defined
#               just like the other P.i's and A.i's.  Also, completed
#               distinction between Quantity Supplied (QS.i) and
#               Quantity Demanded (QD.i, QD.i.j).
#
#    Version c: Added QS.i, Quantity Supplied, with REV.i = P.i*QS.i.
#
#    Version b: Set In::CAP.goods and In::CAP.pop to 1e15 initially;
#               they are set by econ(sim) and should be effectively
#               infinite until then.
#
#               Copied X.i.j, Q.i.j, P.i, Q.i, REV.i, EXP.i from C
#               to out, to make them visible to econ(sim).
#
#    Version a: Based on prototype cd3x3r.cm.
#               Added In:: page, distinguished between base case inputs
#               and dynamic inputs (e.g., BasePopulation and
#               In::population).
```

```
#
#-----------------------------------------------------------------------

#-----------------------------------------------------------------------
# Indices

index i     {goods pop else}
index ing   {pop else}
index j     {goods pop else}
index imost {goods pop}

#=======================================================================
# Null Page
#
# The "null" page contains cells that are global to the rest of the
# model, and that do not change as simulation time advances.  Some are
# inputs to the model; others are computed from the inputs and should
# not be changed.  In principle, many of these parameters are "calibrated"
# from a Social Accounting Matrix (SAM).

# Normal turbulence in employment

let TurFrac = 0.04 ;# Average fraction of workers "temporarily" unemployed

# SAM-based Parameters
#
# A Social-Accounting Matrix can in principle be used to calibrate the
# model, as described here.  HOWEVER, from Athena's point of view the
# SAM data should be used only to determine the "shape" of the economy,
# i.e., the f.i.j's; the "size" of the economy must be driven
# by the Ground and Demographic models.
#
# If there is SAM data, it is:
#
#   BX.i.j     The payment in $/year from sector j to sector i
#   BP.j       The price of one unit of the product of sector j
#
# Then compute:
#
#   BREV.i      = sum.j BX.i.j
#   BEXP.j      = sum.i BX.i.j
#   BQD.i.j     = BX.i.j/BP.i
#   f.i.j       = BX.i.j/BREV.j
#   A.goods.pop = BQD.goods.pop/BaseConsumers
#
# If there is no SAM, we need to input BaseConsumers, A.goods.pop,
# f.i.j, and BP.j.
#
# Since the SAM specifies both "size" and "shape", whereas the
# f.i.j's specify only "shape", and since we must determine the "size"
# from the Ground and Demographic models, we prefer to input the
# f.i.j's rather than the BX.i.j's.  The f.i.j's can, of course,
# be computed from a SAM ahead of time.

#-----------------------------------------------------------------------
# Scenario Inputs
```

```
let BaseConsumers = 1e6     ;# Number of consumers in the inital population.

#  f.i.j is the fraction of j's revenue that is spent in sector i.

let f.goods.goods = 0.2
let f.pop.goods   = 0.4
let f.else.goods  = {1 - [f.goods.goods] - [f.pop.goods]}

let f.goods.pop   = 0.75
let f.pop.pop     = 0.1
let f.else.pop    = {1 - [f.goods.pop] - [f.pop.pop]}

let f.goods.else  = 0.3
let f.pop.else    = 0.05
let f.else.else   = {1 - [f.goods.else] - [f.pop.else]}

# Base prices: These are the sector prices used to calibrate the
# Cobb-Douglas coefficients.

let BP.goods = 1       ;# $/goodsBKT
let BP.pop   = 400     ;# $/work-year
let BP.else  = 1       ;# $/elseBKT

# Base quantities demanded by pop.  The population spends its
# income, BREV.pop, on the sectors according to the f.i.pop's.
# We begin with the per-capita consumption of goods, A.goods.pop,
# and the base number of consumers.  From this, we compute BQD.goods.pop.
# Given the price of goods and the fraction that the population
# spends on goods, we get BREV.pop.  We then use the f.i.pop's and
# P.j's to compute the other BQD.i.pop's.
#
# The BQD.i.pop's are used in the equation for P.goods.  Ultimately,
# the "size" of the economy depends on A.goods.pop and the population.

let A.goods.pop    =  114    ;# Direct consumption of goods, in
                              # goodsBKT/year per capita

let BQD.goods.pop  =  {
    [BaseConsumers]*[A.goods.pop]
} -value 1.14e8    ;# goodsBKT/year

let BREV.pop       =  {
    [BQD.goods.pop]*[BP.goods]/[f.goods.pop]
} -value 1.52e8

let BQD.pop.pop    =  {
    [f.pop.pop]*[BREV.pop]/[BP.pop]
} -value 38000     ;# work-year/year

let BQD.else.pop   =  {
    [f.else.pop]*[BREV.pop]/[BP.else]
} -value 2.28e7    ;# elseBKT/year

#----------------------------------------------------------------------
# Calibration Constants
```

```
#
# These values are computed from the inputs above, and should not
# be modified.

# Cobb-Douglas production function coefficients.  The following
# formulas compute the calibrated Cobb-Douglas coefficients A.i that
# should yield the P.i = BP.i when the model is solved.
#
# The A.j's describe the technology via the production function
# (and the utility via the utility function).  Athena might want to
# change these assumptions eventually, especially for different kinds
# of sectors.

let A.pop = {
    <:prod i {([BP.$i]/[f.$i.pop])**[f.$i.pop]}:> / [BP.pop]
} -value 0.0094501

let A.else = {
    <:prod i {([BP.$i]/[f.$i.else])**[f.$i.else]}:> / [BP.else]
} -value 2.975941843

# Base CPI: defines the CPI for the base case; indexes the CPI to the
# start of the simulation.
let BCPI = 1.0

#========================================================================
# Calibration
#
# The following page is used to calibrate the CGE during scenario
# preparation, based on data from the null page.  It is not recomputed
# as time advances.

#------------------------------------------------------------------------
# Calibration Page
#
# This page defines the basic CGE equations, and solves by using data
# from the null page, sizing the economy by population's demand for
# goods in the base case.

page Cal

# REV.i is the income of sector i: the product of P.i * QS.i, where
# QS.i is the quantity supplied of i's product.
#
# EXP.j is the expenditures by sector j on the various
# sectors i: sum of the X.i.j's down the column.  At present, it is
# used only for output, to verify that EXP.j=REV.j.

# REV.i = P.i * QS.i
# EXP.j = sum.i X.i.j

define REV.i {i} {[P.$i]*[QS.$i]}
define EXP.j {j} {<:sum i {[X.$i.$j]}:>}

let REV.goods = [REV.i goods]                      -value 2.92125e8
let EXP.goods = [EXP.j goods]                      -value 2.92125e8
```

```
let REV.pop   = [REV.i pop]                        -value 1.52e8
let EXP.pop   = [EXP.j pop]                        -value 1.52e8

let REV.else  = [REV.i else]                       -value 3.99e8
let EXP.else  = [EXP.j else]                       -value 3.99e8

# X.i.j is the revenue sector i receives from sector j; it's computed
# as i's share of j's total revenue.
#
# X.i.j = f.i.j * REV.j

define X.i.j  {i j} { [f.$i.$j] * [REV.$j] }

let X.goods.goods  = {<:X.i.j goods goods:>}  -value 5.8425e7
let X.pop.goods    = {<:X.i.j pop goods:>}    -value 1.1685e8
let X.else.goods   = {<:X.i.j else goods:>}   -value 1.1685e8
let X.goods.pop    = {<:X.i.j goods pop:>}    -value 1.14e8
let X.pop.pop      = {<:X.i.j pop pop:>}      -value 1.52e7
let X.else.pop     = {<:X.i.j else pop:>}     -value 2.28e7
let X.goods.else   = {<:X.i.j goods else:>}   -value 1.197e8
let X.pop.else     = {<:X.i.j pop else:>}     -value 1.995e7
let X.else.else    = {<:X.i.j else else:>}    -value 2.5935e8


# QD.i.j is number of i's units "purchased" by j at price P.i;
# it's simply the dollar amount divided by the price.
#
# QD.i.j = X.i.j / P.i
#
# Note that QD.goods.pop is special, as it drives the size of the
# economy.

define QD.i.j {i j} { [X.$i.$j] / [P.$i] }

let QD.goods.goods = {<:QD.i.j goods goods:>} -value 5.8425e7
let QD.pop.goods   = {<:QD.i.j pop goods:>}   -value 292125
let QD.else.goods  = {<:QD.i.j else goods:>}  -value 1.1685e8
let QD.pop.pop     = {<:QD.i.j pop pop:>}     -value 38000
let QD.else.pop    = {<:QD.i.j else pop:>}    -value 2.28e7
let QD.goods.else  = {<:QD.i.j goods else:>}  -value 1.197e8
let QD.pop.else    = {<:QD.i.j pop else:>}    -value 49875
let QD.else.else   = {<:QD.i.j else else:>}   -value 2.5935e8

# Some sort of exogenous demand is required to size the
# economy, so we have chosen to size the economy (in the unconstrained
# case) based on the per capita consumer demand for goods.

# NOTE: We'll redefine this in U as A.goods.pop*In::consumers
let QD.goods.pop = {[BQD.goods.pop]} -value 1.14e8

# QD.i is the demand for the product of sector i in the sector's units.
# It is computed as the sum of the sector-by-sector demands for sector
# i's product.
#
# QD.i = SUM.j(QD.i.j)
```

```
define QD.i {i} {
    <:sum j {[QD.$i.$j]}:>
}

let QD.goods = {<:QD.i goods:>} -value 2.92125e8
let QD.pop   = {<:QD.i pop:>}   -value 380000
let QD.else  = {<:QD.i else:>}  -value 3.99e8

# In the unconstrained case, the quantities supplied, QS.i, are made
# equal to the quantity demanded, QD.i.  This is Walras' Law that
# supply = demand at equilibrium.

let QS.goods = {[QD.goods]}     -value 2.92125e8
let QS.pop   = {[QD.pop]}       -value 380000
let QS.else  = {[QD.else]}      -value 3.99e8

# When j is a production sector, the price of one unit of its product,
# P.j, is obtained by inserting the expressions for the optimal values
# of the ingredients,
#
#   QD.i.j = f.i.j*P.j*QS.j/P.i,
#
# into the condition that REV.j=EXP.j, where REV.j is the product of
# the price, P.j, and the quantity supplied, QS.j, and EXP.j is computed
# by summing the expenses.  We use the Cobb-Douglas production function,
#
#   QS.j = A.j * PROD.i (QD.i.j ** f.i.j)
#
# to express the supplied quantity in terms of the quantities of
# ingredients, and insert the expressions for those quantities.  The
# QS.j drop out of the equation and we solve for P.j.  The solution is
#
#   P.j = (PROD.i (P.i/f.i.j)**f.i.j)/A.j.
#
# Goods and else are production sectors; pop may be treated like one,
# with consumption "producing" the labor.

define P.j {j} {
    <:prod i {([P.$i]/[f.$i.$j])**[f.$i.$j]}:> / [A.$j]
}

# let P.goods = [P.j goods] -value 1.0
let P.pop   = [P.j pop]   -value 400.0
let P.else  = [P.j else]  -value 1.0

# However, the price equations are homogeneous, so one of them is
# useless, and the above equation for P.goods is not used. Instead,
# we use the BCPI (base case consumer price index) as the numeraire that
# defines the value of the $ in terms of a weighted sum of prices,
# where the weights are consumption by consumers in a base case.
# Solving BCPI=1 for P.goods gives:

let C       = {<:sum i {[BP.$i] * [BQD.$i.pop]}:>}      -value 1.52e8
let SUM     = {<:sum ing {[P.$ing] * [BQD.$ing.pop]}:>} -value 3.8e7
let P.goods = {([C]*[BCPI] - [SUM])/[BQD.goods.pop]}    -value 1
```

```
# P.goods was chosen because QD.goods.pop, which becomes a
# divisor, is never zero.

#-----------------------------------------------------------------------
# Diagnostics

# deltaQD.i.pop verifies that QD.i.pop = BQD.i.pop at the end of
# calibration.

define deltaQD.i.pop {i} {
    ([QD.$i.pop] - [BQD.$i.pop])/
    max(1.0, [QD.$i.pop], [BQD.$i.pop])
}

let deltaQD.goods.pop = {<:deltaQD.i.pop goods:>}
let deltaQD.pop.pop   = {<:deltaQD.i.pop pop:>}
let deltaQD.else.pop  = {<:deltaQD.i.pop else:>}

# deltaQ.i: Verifies that QS.i = SUM.j QD.i.j.  The value of deltaQ.i
# should be within an epsilon of 0.0.

define deltaQ.i {i} {
    ediff(0.0, 1.0 - <:sum j {[QD.$i.$j]}:>/[QS.$i])
}

let deltaQ.goods = {<:deltaQ.i goods:>}
let deltaQ.pop   = {<:deltaQ.i pop:>}
let deltaQ.else  = {<:deltaQ.i else:>}

# deltaREV.i: Verifies that REV.i = SUM.j X.i.j.  The value of deltaREV.i
# would be within an epsilon of 0.0.

define deltaREV.i {i} {
    ediff(0.0, ([REV.$i] - <:sum j {[X.$i.$j]}:>)/[REV.$i])
}

let deltaREV.goods = {<:deltaREV.i goods:>}
let deltaREV.pop   = {<:deltaREV.i pop:>}
let deltaREV.else  = {<:deltaREV.i else:>}

#=======================================================================
# Dynamic Pages
#
# The following pages are recomputed at each "tock".  Inputs from the
# rest of Athena and from the user that can change as time passes
# appear on the In page.  Outputs to Athena appear on the Out
# page.

#-----------------------------------------------------------------------
# Inputs page
#
# The values given on this page are notional; the real values will
# come from outside the CGE.

page In
```

```
# Current Consumer Price Inflator
let Inflator   = 1.0

# Consumption Security Factor: decreases consumption due to low
# neighborhood group security.
let CSF        = 1.0

# Labor Security Factor: decreases labor due to low neighborhood group
# security.
let LSF        = 1.0

# Number of consumers currently in the population.
let Consumers = 1e6          ;# people.

# Max capacity for each sector.  These are set by Athena; the initial
# values are intended to be effectively infinite.

# Max production rate for goods
let CAP.goods  = 1e15        ;# goodsBKT/year

# Work Force: Number of people who want to be employed
let WF         = 400000      ;# work-years/year

# Due to the normal turbulence, some of those in the Work Force
# are temporarily unemployed and hence not available to work. Thus,
# CAP.pop is the WF less this turbulence.  In addition, low security
# can reduce the effective workforce because people are afraid to
# go to work.
let CAP.pop = {[WF]*[LSF]*(1 - [TurFrac])}

# NOTE: Because else includes the rest of the world, it is presumed to
# have unlimited capacity even in the constrained case.  Hence, there
# is no CAP.else.

# let CAP.else = 1e15


#-----------------------------------------------------------------------
# Unconstrained Page
#
# This pages runs the CGE for the unconstrained solution, using
# equations copied from the Cal page, sizing the economy based on
# demand for goods by the current population.

page U

copypage Cal -except {
    C
    deltaQD.goods.pop
    deltaQD.pop.pop
    deltaQD.else.pop
}

# QD.goods.pop is now computed from the current population, not
# from the base population.
```

```
let QD.goods.pop = {[A.goods.pop]*[In::Consumers]} -value 1.14e8

# P.goods is now based on the current Consumer Price Index, rather
# than the Base CPI.
let P.goods = {([Cal::C]*[In::Inflator] - [SUM])/[BQD.goods.pop]} -value 1

let CPI = {
    <:sum i {[P.$i]  * [BQD.$i.pop]}:> / <:sum i {[BP.$i] * [BQD.$i.pop]}:>
} -value 1.0

let deltaCPI = {
    ([CPI] - [In::Inflator])/[In::Inflator]
}


#-----------------------------------------------------------------------
# Constrained Page
#
# This page runs the CGE for the constrained solution, i.e., it takes
# the current labor force and production capacity into account and may
# therefore impose limits on the Quantity Supplied (QS.i) by the
# goods and pop sectors, as compared with the unconstrained solution.
# When demand exceeds capacity, prices increase to force market clearance.
#
# Otherwise, the model is the same as on the U page.

page C
copypage U
initfrom U

let QD.goods.pop = {
    [f.goods.pop]*[REV.pop]/[P.goods]
} -value 1.14e8

# The quantity supplied is constrained by the production capacity; if
# the constraint is not binding, it should be at least as big as on
# the U page.
let QS.goods = {
    min([In::CAP.goods], max([QD.goods], [U::QD.goods]))
} -value 2.92125e8

let QS.pop = {
    min([In::CAP.pop], max([QD.pop], [U::QD.pop]))
} -value 380000

# The capacity of else is unconstrained.  This entry doesn't really
# change anything, but it makes the assumption explicit.
let QS.else = {[QD.else]}   -value 3.99e8

let P.goods = {
    ([f.goods.pop]*[REV.pop] + [f.goods.else]*[REV.else]) /
    ((1.0 - [f.goods.goods])*[QS.goods])
} -value 1

# The value of P.pop implied by CPI=1 in the unconstrained case is the
# numeraire for the constrained case.  Note that P.pop is already
```

```
# inflated by the In::Inflator.

let P.pop = {[U::P.pop]} -value 400

let P.else = {
    ([f.else.goods]*[REV.goods] + [f.else.pop]*[REV.pop]) /
    ((1.0 - [f.else.else])*[QS.else])
} -value 1


#========================================================================
# Page Out: The outputs.

page Out

# Copy base case outputs from Cal
let BQS.goods = {[Cal::QS.goods]}

# Copy outputs from C
foreach i {goods pop else} {
    let P.$i        = {[C::P.$i]}
    let QS.$i       = {[C::QS.$i]}
    let REV.$i      = {[C::REV.$i]}
    let EXP.$i      = {[C::EXP.$i]}
    let deltaQ.$i   = {[C::deltaQ.$i]}
    let deltaREV.$i = {[C::deltaREV.$i]}

    foreach j {goods pop else} {
        let QD.$i.$j = {[C::QD.$i.$j]}
        let X.$i.$j  = {[C::X.$i.$j]}
    }
}

let CPI = {[C::CPI]}

# goods shortages and overages: goodsBKT/year
let LATENTDEMAND.goods = {max(0.0, ediff([U::QD.goods],   [C::QS.goods]))}
let IDLECAP.goods      = {max(0,0, ediff([In::CAP.goods], [C::QS.goods]))}

# pop shortages and overages: work-years/year
let LATENTDEMAND.pop   = {max(0.0, ediff([U::QD.pop],   [C::QS.pop]))}
let IDLECAP.pop        = {max(0,0, ediff([In::CAP.pop], [C::QS.pop]))}

# Unemployment Statistics
let RealUnemployment = {max(0.0, ediff([In::CAP.pop],[C::QS.pop]))}  ;# People
let Turbulence       = {[TurFrac]*[In::WF]}                          ;# People
let Unemployment     = {[RealUnemployment] + [Turbulence]}           ;# People

# Unemployment Rates, real and reported
let RealUR = {100.0 * [RealUnemployment]/[In::WF]} ;# Percentage
let UR     = {100.0 * [Unemployment]/[In::WF]}     ;# Percentage

# Gross Domestic Product, GDP

let GDP  = {
    [X.goods.pop] + [X.pop.pop] + [X.goods.else] + [X.pop.else]
```

```
}

# GDP deflated by CPI, which is used as a proxy for the GDP deflator.
let DGDP = {[GDP]/[CPI]}

# Sanity Check Values
let SUM.QS = {
    ediff([QS.goods] + [QS.pop] + [QS.else], 0.0)
}

let FLAG.QS.NONNEG = {
    [QS.goods] >= 0.0 || [QS.pop] >= 0.0 || [QS.else] >= 0.0
}

let FLAG.P.POS = {
    [P.goods] > 0.0 || [P.pop] > 0.0 || [P.else] > 0.0
}

let FLAG.DELTAQ.ZERO = {
    [deltaQ.goods] == 0.0 && [deltaQ.pop] == 0.0 && [deltaQ.else] == 0.0
}
```

13. ACRONYMS

AAM	Athena Attrition Model
AUT	Autonomy (concern)
CIV	Civilian
CMO	Civil/Military Operations
CTR	Contractor
CUL	Culture (concern)
DAM	Driver Assessment Model
ECDA	Expected Collateral Damage per Attack
ECDC	Expected Collateral Damager per NF Casualty
ELER	Expected Loss Exchange Ratio
FRC	Force
GRAM	Generalized Regional Attitude Model
IED	Improvised Explosive Device
IGO	International or Inter-Governmental Organization
JNEM	Joint Non-lethal Effects Model
LER	Loss Exchange Ratio
LFF	Labor Force Fraction
MAD	Magic Attitude Driver
MAG	Mars Analyst's Guide
NF	Non-uniformed Force
NGO	Non-Governmental Organization
ORG	Organization
QOL	Quality of Life (concern)
ROE	Rules Of Engagement
S&RO	Stability & Recovery Operations
SFT	Safety (concern)
UF	Uniformed Force
UNESCO	United Nations Educational, Scientific, and Cultural Organization