The General Theory of RTS Models

Papers 2 & 3 Combined

RTS and Classical Military Strategy

Table of Contents

[1. Introduction 3](#_Toc333473366)

[3. Overview of RTS Game Design 1996 to Present 4](#_Toc333473367)

[2. Why Develop a Strategic Game? 5](#_Toc333473368)

[2.1 Definitions of Strategy and Tactics 5](#_Toc333473369)

[2.2 Effects of Game Modelling on Military Strategy and Tactics 5](#_Toc333473370)

[3. A Statement of Design Intent 8](#_Toc333473371)

[4. Physical Scales 9](#_Toc333473372)

[4.1 Small Arms and Artillery 10](#_Toc333473373)

[5. Commencing the Design Process 11](#_Toc333473374)

[5.1 Bypassing Systems Modelling in Initial Trial Cuts 11](#_Toc333473375)

[6. Making the Game Playable in the Player’s Real Time 13](#_Toc333473376)

[6.1 Essential Elements of the Time and Space Problem 13](#_Toc333473377)

[6.2 An Innovative Solution to Time Scale Problems 15](#_Toc333473378)

[7. The Trial Approach 17](#_Toc333473379)

[8. Towards Fully Integrated Design – User Control 18](#_Toc333473380)

[9. How Humans Play RTS – Lessons for AI design 20](#_Toc333473381)

[9.1 Compact Bases and Nodal Spread 20](#_Toc333473382)

[10. A Proposed Advance in RTS AI 22](#_Toc333473383)

[10.1 Notes toward a Strategic AI (Resourcing) 23](#_Toc333473384)

[10.2 Notes Toward a Strategic AI – Military Strategy 25](#_Toc333473385)

[Still Not Convinced? 26](#_Toc333473386)

[A Simplified “Clausewitzian” AI - Proof of Concept Example 29](#_Toc333473387)

[Appendix 1 –Summary of General Design Theory for RTS 33](#_Toc333473388)

[RTS Meta-Law 1 – The Meta-Law of Intrinsic Design Conflict 33](#_Toc333473389)

[Corollary of Meta-Law 1 33](#_Toc333473390)

[Expansion of SML 1 - Realism, Scaling and Playability 33](#_Toc333473391)

[RTS System Meta-Law 2 - The Meta-Law of Realism 34](#_Toc333473392)

[Corollary of Law 2 34](#_Toc333473393)

[SML 3 – The Meta-Law of Economy, Supply and Logistics 34](#_Toc333473394)

[Quick Explanation of Law 3 34](#_Toc333473395)

[System Meta-Law 4 –Determinants of the Strategic RTS Game Model 35](#_Toc333473396)

[High Level Design Goals 36](#_Toc333473397)

[Basic Design Rules 37](#_Toc333473398)

[A Note on Laws, Goals and Design Rules 38](#_Toc333473399)

[Appendix 2 – Further notes on Law 3 – The Law of Realism 39](#_Toc333473400)

[Newtonian Physics and the Modelling of Mass, Force, State and “Attributes” 39](#_Toc333473401)

[An “Attributes” Model 41](#_Toc333473402)

[Sundry Notes 41](#_Toc333473403)

[Appendix 4 – Discussion of Exponential and Linear Growth Models 42](#_Toc333473404)

[Appendix 5 – Economy Realism in RTS 44](#_Toc333473405)

[Appendix 6 – Military and Philosophical Quotes 45](#_Toc333473406)

[Appendix 7– An Approach to Naval and Air Power in Realistic RTS 46](#_Toc333473407)

[Appendix 8 – The Gathering Equation 47](#_Toc333473408)

# 1. Introduction

In paper 1 it was proved that RTS computer games are mathematically, geometrically and algorithmically governed systems models of the real world. As such they have invariable relationships both internally and with regard to the real world which they model. These relationships can be described by Laws. The law bound nature of RTS models was demonstrated at two levels. First it was demonstrated that some laws of accurate RTS models unavoidably replicate some empirical laws of the real world. Second, it was demonstrated that, as a formally and internally consistent mathematic / geometric / algorithmic (MGA) system, the genre of RTS models possesses a single set of internally consistent laws. These laws are akin to and indeed in many ways formally related to the axioms of Euclidean geometry and the Laws of Newtonian Physics.

Next, it was shown that each genre of MGA games – and RTS is such a genre - will possess its own internally consistent laws. From the laws of each genre we can derive the System Meta Laws of that MGA genre. In a very real sense, the System Meta Laws of a genre reveal the basic nature of that MGA genre, how it works fundamentally, how to make it do what you want it to do and how to make those choices work best. The System Meta-Laws (SMLs) derived function as firm heuristic goal-seeking guides to the design, specification, building and use of the complex mathematic-geometric-algorithmic (MGA) system in question. Finally, it was implied that most commercial RTS design has now settled for a re-hashing of tactical RTS design and baulked at the difficult obstacle of genuine strategic design.

# 3. Overview of RTS Game Design 1996 to Present

The first explosion of innovation in RTS design occurred in the decade from about 1996 to 2005. This was the era of genuine intuitive design genius in RTS concept design. This refers to overall concept design and not to technical design in detail. The basic theory of design components (e.g. path finding and tree search algorithms) had been elucidated much earlier in the fields of mathematics and game theory. The essence of RTS concept design circa 1996 - 2005 was that people began creating excellent tactical games but often lacked any over-arching formal concept design theory which would have aided in the creation of larger scale and strategic games.

Both premeditated and intuitive design decisions taken to make the games playable, commercially viable and technically feasible on computers slow by today’s standards, led very naturally to the creation of small scale tactical games, although there were some exceptions to this rule. This combination of experimentation and intuitive design genius is fruitful in the formative stages of any new art, discipline or enterprise. In fact, it is one of the prerequisites to create a new field. However, it is no longer enough when a second or mature stage is reached. At this stage, a theoretical review and formal exposition of high-level design elements is required to enable designers to synthesise new combinations and break new ground.

Since about 2005, the RTS genre has become conceptually stagnant. This is a bold claim but a supportable one. The formulaic paradigm of small maps, small unit limits, short sight-ranges and tactical fights over limited and scattered resources has rarely been improved upon except in battlefield oriented games which are not pure RTS, like the Total War series, or in some mods made by aficionados of military realism. This remains true even though the technical limits preventing the production of larger scale RTS games have been removed. While the technical limits have been overcome, the conceptual and commercial limits largely remain in place and a new theoretical leap is required to take RTS to the next level.

Much of today’s increased computing and graphics power is being used to develop graphically impressive 3D RTS games but these games remain trapped in the tactical paradigm. The extra computing power is not being used to move RTS into the strategic realm for two reasons. The first reason is the clear imperative for tried and true playability and commercial return. The second reason is the failure on the part of many commercial computer game developers to fully understand what the term “strategic” means. Without any military background or specialist military reading they are not equipped with the required concepts.

# 2. Why Develop a Strategic Game?

The RTS game market is saturated with tactical games. These rapid tactical games are the province of youthful players. An innovative strategic game which combines more realistic space and time scaling with large maps and large unit limits, but still retaining genuine playability, would be a great advance. Such a game would appeal to more mature computer game players. This is a niche market right now but will grow over time if nurtured in the correct manner. As the current cohort of RTS players in their twenties move into their thirties, and beyond ,they will very likely leave RTS gaming behind when no longer able to micro the rapid tactical games competitively against younger players.

 However, these older players, who are maturing intellectually, will appreciate the deeper challenges of a truly strategic game. Such a game will be more about positional understanding, manoeuvre, classical military theory and understanding concepts such as centres of gravity, concentration of force and indirection (feints and deception) rather than being mainly concerned with rapid mouse clicking and lightning tactical tricks. Such a game will possess a positional and strategic challenge equivalent to traditional intellectual games like Chess and Go.

## 2.1 Definitions of Strategy and Tactics

Strategy (as “strategics”) is the science or art of combining and employing all the means of war in planning and directing large scale military movements and operations. A strategy comprises an overall plan, method, or series of manoeuvres for obtaining a final goal or result. A tactic is a smaller scale or local plan of action - and its ensuing manoeuvres - designed as an expedient for gaining an intermediate objective. This intermediate objective is usually a step in furthering the overall strategic plan. This holds true even if a tactical action is an indirect or diversionary action. Good tactics are characterised by clever thought, skilful actions and adroit manoeuvring. Good strategy is characterised by profound long term thinking and planning.

## 2.2 Effects of Game Modelling on Military Strategy and Tactics

It is true that very tactical and (let it be said) excellent RTS games like Starcraft also do have overall strategies. However, the rapidity of the tactics means only the fastest players have enough spare thinking time and finger speed to run a coherent strategy in-game as well as execute the rapid squad tactics in detail. Artificial “tech trees” often ensure that strategies are primarily synthetic “tech up” strategies along differing tech lines which only secondarily and dependently manifest themselves as “military” strategies. There are also certain design factors and structural limitations in tactical RTS which mean that classical military strategy (as opposed to essentially artificial RTS expansionist strategy) is not employable. This clear difference between classical military strategy and the artificial expansionist strategy of most RTS games is caused partly by the simplifications inherent in RTS modelling but it is much exacerbated by the particular modelling decisions being taken (usually instinctively) to make the game rapid, fun and instantly appealing to younger players.

Many RTS games are won by rapid expansionist play which generates a very tactical style and many economy based wins. The economy model and the resources/map model have much to do with this. Where key resources are spread in relatively small clumps more or less all over a map, the game can favour rapid expansion and squad tactics at each resources site. When the resources are exhaustible and the resource mine points are non-upgradeable this further favours squad tactics at each resource site. The game potentially becomes a chaotic tactical squabble over scattered resources. We can see that a game design suffers from this syndrome when map design is used to counter the problem. The maps artificially set up excessive chokes, obstacles and impassable terrain sectors to generate natural zones of control for each player and reduce the chaotic tactical element.

This rapid expansion game style can lead to a dispersal of forces which is the antithesis of classical military internal lines strategy. (Internal lines and external lines are explained in the next paragraph.) In such a game model it is more difficult to play an early to mid game internal lines strategy in any effective way unless map design implements multiple chokes to assist internal lines strategy. Starcraft 1 and 2 provide examples of this multiple choke approach in virtually all official and competition maps. When map design in itself (open terrain versus chokes for example) has such a greatly determining influence on the most general of strategic possibilities (the situational or game phase use of internal lines versus external lines for example) then the game model itself can be judged to be over-influenced by specific map design choices. The game model is arguably too “map dependent” or “map-malleable”.

A re-worked game model could encourage more classical military movements whereby “internal lines” or “external lines” are each employed in their appropriate context by the advanced player, even on open, naturalistic terrain. An internal lines posture means maintaining the main army mass in a relatively compact and powerful form whilst having small detachments, scouts or skirmishers ranging out to generate a large field of view, cordon or picket line around the massed and powerful centre. Internal lines are often maintained when an army is trying to locate and come to grips with an enemy army in the field. The early warning provided by a large scouted field of view means that inferior enemy forces can be massed upon and forced into an engagement. Superior enemy forces can be avoided or lured back to an engagement at a rendezvous point with supporting friendly forces.

Internal lines are also often maintained when trying to perform the dual task of seeking the enemy in the field but not moving the main army mass too far from the defence of the main operational base or supply dumps which are the army’s supply centre of gravity. Strict internal lines may also be sounder during the unsettled process of first setting up a base when supply is either assured (by large central stores arrived at base) or resources are available close at hand. There is little to gain and much to lose at this stage of operations by ranging out blindly. Where an army is so large that it is able to maintain a corps system (each corps essentially being a small army in its own right with adequate complements of infantry, cavalry and artillery and also sufficient supply) then each corps may maintain a separate internal lines set-up and be capable of sustaining a fight against a contacted army for a day or so until reinforced by the other corps. This is true of the historical era (Napoleonic) later put under design consideration.

An external lines posture essentially means forming a strong perimeter or front line. External lines can be appropriate for a large secured base. Such a base will need a strongly secured perimeter. All assets inside the base must be guarded. However, a compact base with perimeter defence (ostensibly external lines), a good field of view and long lines of fire on all attack approaches can also be said to be in an internal lines posture. A classic case of external lines occurs when a besieging force surrounds a base, fort or fortress. The besieging army sets up an external lines cordon around the enemy base or fortress. It wants to prevent all communication, reinforcement and resupply to the fortress. Another classic case of external lines occurs when very large armies (typically numbering in the hundreds of thousands or millions) form long front lines in a massive theatre of war like WW1 or WW2. It can perhaps also occur when smaller armies confront each other across a relatively narrow front where significant terrain features preclude flanking movements.

A more classical military style RTS game can be created by ensuring key resources (like mines for key minerals) are in relatively compact groups separated by much open territory. The mines should perhaps be upgradable a number of times. Incidental resources (like timber and stone) should be more or less ubiquitous in clumps over most of the map and relatively slowly exhaustible so players can get enough in any general operational area. The economy model should be of gatherers (peasants or peons) and of linear rather than exponential growth. Reasons for linear growth were discussed in Paper 1.

A base that can be walled or trenched and emplaced frees up troops from pure defensive duties and allows a greater army to operate in the field, so these features could be incorporated in the design. All of these factors create many chances for classical internal lines play and full army clashes in the open countryside. Normally, winning a large battle in the open (if one gets to the middle game) will lead to “winning the field” and thus controlling the open territory which will be a prerequisite for establishing say a second base or perhaps destroying the enemies’ second base as the case may be. Then the controlling player can take control of more territory, build up further, switch to an external lines posture and cordon the main enemy base to invest or lay siege to it. This is all very classical military play. This is what a good game model will give us.

The key here is to mimic (if it is a full RTS) the economy / army interactions of the period chosen (if an historical era is chosen). The standard RTS model with scattered key resources on small maps does not do this. The standard RTS model vastly exaggerates, by compressions in time and space, the ability of an army to forage and grow from plunder and new resources found in a conquered countryside. Armies campaigning on the Napoleonic model may have maintained themselves to some extent by looting the countryside for provender, forage and general supplies. However major military supplies, new cannon (apart from some captured pieces) and masses of newly trained and reliable recruits cannot be raised easily from conquered lands, at least not in the short to middle term. They must come from the demographic and economic centre of gravity of the campaigning nation i.e. the established national power which gave the army the critical mass to commence the campaign.

# 3. A Statement of Design Intent

In any first attempts to utilise this new General Theory of RTS (as per Paper 1) we should limit first experimental game designs to ground forces and to an era without air power and without mechanised vehicles. The central reasons for limiting a first design attempt in this way is to limit the extent of distortions of time and space necessary to make the RTS playable in the real world time of the players. Air power, naval power and mechanised power present extra modelling problems for the purist approach which insists on full space and time modelling realism. This paper does not attempt to deal seriously with these air and naval problems.

This is not to say that further refinements of these theories could not solve these issues. However, following the “divide and conquer” principle for problem solving it is the correct method to limit initial analysis to ground forces and to an era where the fastest unit is cavalry. Many modern RTS style games do incorporate naval and air power but the extreme ranges (and speeds in the case of air power) are never handled realistically. Sometimes firepower is called in from off map to hit map coordinates in “deus ex machina” manner. Appendix 7 nevertheless makes some brief suggestions for approaching design issues re modern naval and air power.

The reasons for being cautious about the precise game model at this point will become clear as we consider the full design intent. The design intent will provide clear principles to guide the game concept design. The game that evolves will be the outcome of both practical and imaginative design decisions taken during the full design process. These decisions will be taken to translate a purist and militarily classical design intent into a playable and commercially viable game. Therefore we are not starting with any preconceived idea to make either a pure battlefield game or a standard RTS. Instead we will start the concept design phase with the intent of being guided by clearly elucidated design principles in the firm belief that this high level of rigour will lead to a genuine advance in game design rather than the production of another battlefield or RTS clone. (I use the pronoun “we” as I assume the reader has come on board imaginatively to consider this project.)

What do we mean by commencing with a purist and classical design intent? In simple terms, we propose to make a genuinely strategic game and not merely a tactical game. Further, all military actions must play out with historically accurate military realism. The first and obvious difference between a fully strategic game, which includes tactics, and a merely tactical game is in its size and scope. Strategic games have large unit counts and wide landscapes. The second clear difference is that a genuinely strategic and classical military game needs realistic relative scaling of all distances, sight ranges, weapon ranges, unit action times and traverse speeds; in short, the accurate scaling of real time and space relationships into the game. Any game directly built with such parameters without attendant design innovations will take an inordinate time to play out. As we saw in Paper 1, this introduces our first design problem. This design problem may be stated as the First System Meta-Law (SML) of RTS computer game design; the Meta-Law of Intrinsic Design Conflict.

As the stated design goal is a purist and militarily classical design, the first step (perhaps unsurprisingly) should be to create the military aspects of the game. This will involve the units, military structures and field works as well as the landscapes and maps. This first cut test design should be created as a battlefield game with an editor function to allow the placement of units, formations and structures for battlefield testing. Landscapes must be naturalistic and all fighting methods of the era must be possible. Furthermore, vegetation and landscapes must be fully deformable and tunnels and earthworks feasible. The reason for proceeding with the military design first is that no RTS-style economic build system can possibly generate and maintain a militarily classical game overall unless the military aspects in themselves are already historically accurate and classically integrated. Furthermore, the stated design goal at the outset is to scale the game accurately in all respects. To further this goal, we must now consider real scale data from the era.

# 4. Physical Scales

To scale historically accurate battles into this game we must first consider some facts. The afternoon engagement of the battle of Marengo, 14 June 1800, took place in an area which can be measured off as 10 miles by 8 miles. The battle of Borodino, 7 Sept 1812 can be represented in its essentials on a map depicting an 8 miles square. Napoleon’s Waterloo, meaning the actual battle of Waterloo on 18 June 1815, including Blucher’s arrival to support Wellington, is easily encompassed in a 10,000 yard square. These are surprisingly small distances and perhaps not impossible to depict in a game. However, just four key days of the Waterloo Campaign, 15–18 June, require a map 60 miles by 60 miles to demonstrate the main army movements.

This would indicate that the best we can hope to represent somewhat realistically and yet also stylistically and in a playable manner in a battlefield or RTS game of this sort is a piece of terrain equivalent to a relatively compact, single day battle site of the era. This we will take to be a square 10,000 metres by 10,000 metres.

## 4.1 Small Arms and Artillery

In terms of a battle around the turn of the century, circa 1801, what can happen in an area 10,000 by 10,000 metres? History shows us that two or even several armies of up to 64,000 men per army can take part. Next, we must consider musket performance and artillery ranges. The following musket performance comes from Prussian field trials in the late eighteenth century and represents average outcomes on the equivalent of a battalion frontage (probably about 500 men in 3 ranks with the 1st rank kneeling, 2nd and 3rd ranks standing, all both ranks firing simultaneously).

Small Arms Performance

Range Hits

225 metres 25%

150 metres 40%

 75 metres 60%

Note: One assumes these field trials were made against cut out targets or straw targets of some kind. There is a considerable body of research that shows many humans are not entirely willing or able to fire directly at other humans in real battle. Plunging cold steel into a person at arm’s length can be even more difficult to come at. Depending training, veterancy and so on, many real battalions may not achieve the 75 metres figures, for example, against live humans under the stress of battle and against their innate and socialised human feelings. It has been found that under the influence of these feelings a proportion of soldiers, often unconsciously, tend to shoot high or low. We might want to consider such issues for our probability model for hits.

Artillery of the Gribeauval System

(Ranges quoted are approximate, allowing for less than perfect gun performance.)

 Gun Max Range Effective range Canister Range

12 pounder 1,800 m 900 m 600 m

 8 pounder 1,500 m 800 m 500 m

 4 pounder 1,200 m 700 m 400 m

Gun Max effective Min safe effective Shell burst danger zone

 6 in. howitzer 1,200 m 600 m 40 m diameter

If we round things back up just a little we can see the maximum range of a 12 pounder gun crosses a fifth of the recommended map and the effective range crosses a tenth of the map.

# 5. Commencing the Design Process

## 5.1 Bypassing Systems Modelling in Initial Trial Cuts

Earlier analysis (in Paper 1) identified three basic systems in the Systems Model that is the game model. These systems are;

* The environment
* The economy
* The military

It was further identified that the environment is the “given” system and the economy and the military are dependent systems. The economy is dependent on the environment. Then the military is dependent on the economy. This might be thought to indicate that the resource environment is the first layer of our game system or game model and that the environment ought to be fully considered first. This turns out not to be the way to proceed.

A stated design goal is realistic military action and realistic military interactions with the environment. The question of this being an RTS or a battlefield game (or some amalgam thereof) has been left in abeyance at this point. Thus the aspects to begin with are a basic environment only and first elements of the military. A layering method of design suggests we start testing our assumptions with the simplest environment possible. Consistent with our goal of realistic scaling this should be a rendition of a flat plain scaled to represent 10,000 metres square of plain ground.

As a trial cut, this wide and perfectly flat plain of 10,000 metres square should be created by a chosen standard method. A terrain map this size will allow full armies of the period to manoeuvre and clash. This plain will actually be at the mid-point plane in a 3D space allowing for suitable depressions and elevations to be added (on a map contingent basis) at a later stage. A 1,000 metre total vertical axis should allow sufficient space. In other words, a cube is defined that is 10,000 m by 10,000 m by 1,000 m and represented in the chosen scale. The scale, appropriately chosen for clear rendition purposes may be 1:100 for example. The intention is to trial accurately scaled renditions of all landscape, unit and weapon relationships. Trials and choices of physics engines, 2D/3D renditions of sprites and player point(s) of view will also occur at this point. The landscape will be 3D. Sprites (both active like units and passive like trees) will preferably be 3D but given the unit numbers being sought (possibly up to 64,000 units per player!) they might need to be 2D superimposed on the 3D landscape. Another possibility is a combination system of 3d sprites when zoomed in and 2d animated sprites when zoomed out.

Special 3D viewing options (like zooming in, zooming out, panning in both planes and others) may not need to be part of the player view options in ultimate game design as many such options are not really viable in competitive play. For example, rotating the landscape (say from viewing a centre-situated mountain from the far north to viewing it from the far south) is both unrealistic, as a player point of view implementation, and unusable in practical play because a player will become confused about directions. However, all landscape viewing options should be available in the test engine.

# 6. Making the Game Playable in the Player’s Real Time

I do not consider that technical hardware or programming limitations will be the major obstacles to creating such a game. A landscape space of the size outlined above and unit limits of at least 16,000 per player ought to be eminently achievable on contemporary computer. The most challenging aspect of building this type of game will be making the game playable in the players’ real time. This is where some real design innovation will have to occur.

First, we ought to consider what period is a reasonable amount of players’ real time to require for a battlefield or RTS game of this style. Younger players playing tactical RTS seem to want the whole game to happen in 30 minutes or less. They want it to run at a rapid tactical pace. This rapid pace coupled with small maps and exponential growth models allows for rapid build ups and then quick game conclusions. However, we are advocating a design for mature players and aficionados of classical military strategy. In this case, we can “demand” a greater investment of player time. Serious chess or Go players are willing to devote at least one to two hours to hard fought individual games. We suggest that this time span form the basis for our planning. Not all games will run for this long but serious games between well matched expert players will often do so. We already stated that, “The afternoon engagement of the battle of Marengo, 14 June 1800, took place in an area which can be measured off as 10 miles by 8 miles.” Thus an afternoon of four to six hours (still too long) could see a decisive clash of the era played out in scaled space and real time meaning real time both in the game and in the player’s real world.

## 6.1 Essential Elements of the Time and Space Problem

An innovative solution to the time problem is possible. The first step is to consider the elements of the problem. Given the size of the recommended terrain (10,000 metres by 10,000 metres) we must consider movement speeds. A steadily marching column on good open ground will cover, we shall assume, 5,000 metres in an hour at 5 kph). Suitable allowances must be made for difficult ground on the one hand and marching at the double across good ground on the other. Horses ambling along with infantry or pulling wagons and gun carriages over good cart tracks we can also assume cover 5,000 metres in one hour. Heavy or armoured horses like Cuirassiers “charge” at a moderate trot. This is about twice the walking speed mentioned above; approximately 10 kph to 15 kph. Let us assume a nice round number of 200 metres per minute for charging at a trot. Light horse or hussars can charge at a canter working up to a moderate gallop. Let us assume double to treble the speed of the trot-“charge” and nominate a rate of 400 metres to 600 metres in a minute over average ground. All these cavalry horses, heavy and light, are likely to be quite winded after say a four hour walking-march, more hours standing in the field and then say a one to three minute charge with relatively heavy military burdens.

Let us assume armies are moving from the side edges (not corners) and drawing up as rapidly as possible for a large battle in the centre. This battle will commence at a nearly effective artillery range of 1,000 metres. (It is easier to test our first assumptions in simple round numbers.) Assume simple plains terrain. As the map is 10,000 metres across, each army needs to move about 4,500 metres. We will assume that each army can do this in one hour but bottlenecks and problems of drawing up in battle array account for another hour. This is probably an optimistically short time but let it stand.

In the context of a computer game, two hours to merely march and line up the major armies in the centre is laughably long. The issue becomes even worse when we consider that at this point this is just a battlefield game model with no RTS component. It seems clear that we must compress time and distance or distort relative time spans and distances in some manner to make the game playable. These combined physical scale and time scale problems stem from the fundamental conflict embodied in SML 1; the conflict between the desirability of a realistic rendition of time and space and the requirement for “playability”. How can we deal with this scaling issue? There is the old way of compressing distance which is the wrong way for military realism purposes.

It is clear that compressing distances is wrong in principle for a game seeking to generate military realism. Correct physical scales are necessary to render battles accurately. If we make maps small then we get effects like artillery being able to shoot right almost across a map. If we scale down artillery ranges to prevent this then we get effects like musket ranges being too great relative to artillery ranges. If maps are too small then formations cannot deploy properly with realistic and historically accurate physical formation sizes and troop spacing. These are some of the problems that Cossacks, good game as it was, failed to solve. We are forced by all these issues to decide which principles we will or will not compromise in our quest for a military purists’ style rendition of the era. It is best to set up perfectionist rules and then seek innovative ways to obey the rules. I would suggest a key strategic RTS design rule as follows.

* **DESIGN RULE 1 – There will be no compromise on accurate physical scaling and physical effects.**

This means all distance scaling, ballistics and basic Newtonian physics must be accurately modelled. The unmitigated effect of Rule 1 would be that the game would take too long and be unplayable unless it was a pure battlefield game with the armies commencing in battle array and the battle being decided in about one long afternoon of play. The question now is how do we mitigate or modify the effect of rule 1 on playability without compromising the rule itself?

## 6.2 The Two-Speed Real Time Engine

A possible solution is as follows. It involves a two-speed real time engine both being real-times running in two different timescales. There will be a strategic timescale and a tactical timescale. Each time scale will have its own appropriate display. The strategic timescale has a strategic map and the tactical timescale has a tactical map. These displays must be combined and switched to show in foreground or background as required and in a manner that is as seamless as possible to the player. The best way to describe this process and how this whole aspect of the game would work is to describe the user interface or player display as we may term it. This approach grows out of existing approaches to RTS games and will feel like a natural evolution to keen RTS players.

Something only remotely like this has been tried before in Medieval Total War but with jarring (non-seamless) switching between a turn based strategic phase and a real time tactical phase. I am not proposing that. What I am proposing are real time strategic phases interspersed with real time tactical phases all with seamless switching directly controlled by the game engine but flowing from player actions. In strategic time, game time will run faster, probably 60 times faster, so that a minute of player time sees an hour pass in the game world. In strategic time, formation movement orders are given, long movements occur and construction of a military nature (like palisades or entrenchments) is ordered and undertaken. Formation movements (of friendly forces) are shown by the movements of formation symbols. Formation movements of opposing forces are shown by symbols but only when sighted by friendly forces.

This model is most assuredly NOT like the Supreme Commander model where the player can scroll in and out at will from a “strategic view” to a “tactical view” and back again. The Supreme Commander model makes no changes to time. Time is always tactical time in Supreme Commander and the game still makes many of the standard tactical RTS design compressions and distortions of speeds, sight ranges and weapon ranges. Supreme Commander is innovative in this area of game design but the zoom, though it is a great graphic feature which enhances game appearance and user control, does not alter the treatment of time as tactical time.

In the proposed design, players will not be able to initiate or conclude a tactical time phase other than by engaging or disengaging a militarily significant force with or from the enemy. Engaging and disengaging is this context will mean approaching or leaving a defined potential engagement range called an “action range”. Single scouts could never initiate a tactical phase by contact with the enemy. The orders of single scouts would in any case be to return with information of enemy movements and not to engage in any suicidal actions. The life and death of single scouts in the fog of war (until they return to HQ) will be decided in black box fashion by the game engine based on what they encounter.

To the above end, scouting ordered in the strategic phase (or scouting ordered in the tactical phase for that matter) works like this. Single scouts or small scouting groups move out into the fog of war. The player loses sight of them and they provide no sight of the map. They also provide no further information until they return. When they return, to HQ or to a main formation with a dispatch rider which must then relay the message to HQ or the General’s current location, the terrain the scout(s) sighted is then revealed but in a misty grey not in the clear light of direct vision. Fixed features like roads, woods and villages are revealed in this manner.

The display in a tactical time phase will look like and offer control much like a standard RTS screen display, albeit in a realistic scale. That is to say the tactical map will then occupy most of the screen and the strategic display will occupy the lower right corner or lower left corner of the screen at the player’s discretion. The strategic display will always carry strategic information, including formation symbols, even in the tactical time phases. This may make the strategic symbols too small to see therefore the strategic display can be enlarged at will by the player but never to more than ¾ of full screen in a tactical phase.

The game engine’s automatic implementation of the full screen strategic display (after a tactical phase) will signal a shift back to the strategic time scale. This will happen and persist whenever no significant opposing formations are in action range of each other. Action range will be accounted to be 500 metres when no artillery is present locally on the map. Action range will be accounted to be 1,500 metres when artillery is present locally. A formation which can be hit (e.g. by cannon) but which cannot immediately hit back is still accounted to be in action range.

The overall purpose of the strategic time scale phase is to allow the time to assemble armies, traverse them and draw them up in battle array all in a length of players’ time that is relatively quick. Nevertheless, strategic phases will require high levels of player skill and knowledge in their own right. A commander will have to know how to assemble, manoeuvre successfully and correctly dispose his army in battle array. It will be rather chess-like and yet still require excellent scouting and considerable rapidity of thought in the 60x speed time scale. Then engagements will be resolved in tactical time and the tactical display will give the player all the standard enjoyment of tactical RTS engagements. Yet, at all times, the game will operate on a strategic scale with a full range of strategic and tactical challenges.

# 7. The Trial Approach

Consistent with the trial approach, I would advocate the creation of an initial experimental engine at low cost. As already stated, the engine would require a 10,000 m square landscape with no or few terrain features, an editor capacity to place several musket formations for two nations or factions and suitable basic mechanics for simple musket formation engagements. In addition, the experimental engine would contain the strategic and tactical components and interfaces of the time management engine. The idea would be that developers and testers could trial traversing and manoeuvring musket formations (shown as symbols) in the strategic time display (at the 60x time speed) until engagement range was achieved. Then they could fight tactical engagements, continue engagements and/or seek to disengage (separately and jointly) and see how the trial engine handled it.

Just as importantly, they would be seeing how they handled it. That is they would be assessing the playability and subjective feel of such a game design. Such trials could be made both with and without a realistic fog of war. A realistic (as opposed to stylistic) implementation of the fog of war will need to be developed.

## 7.1 Whole of Game Balancing

We need to further note here that what is envisaged for this game are a suite of innovative design approaches right across all aspects of the game. This will particularly involve “game model balancing” which is about balancing the environment model, the resource/economy model and the military model in relation to each other (holistically) to achieve the stated design goals. In this wide purview, game balancing goes well beyond mere weapons balancing and faction or nation balancing. It is about balancing all the interacting systems models to get the desired overall game balance and game design. We can identify other components which are both overarching ones (the game physics engine) and sub-components (morale models) which all also must be integrated and balanced holistically and accurately to get the required game design.

There are a great many innovative ideas which can and should be trialled and incorporated in some form. Naturalistic terrain for example could perhaps be obtained from Google Earth (on licence as required) or some similar source. Historical battle sites (Austerlitz or Wagram for example) could be taken from chosen source sites and then re-mapped and re-rendered using historical research to match the precise terrain features, village sites and so on as they existed at the appropriate period in history. I am recommending a game design so accurate, so comprehensive and so user friendly that historical researchers, military buffs and even war colleges could want to buy it for research and teaching purposes related to the study of historical engagements of the era. I see no reason why it cannot be both a simulator of this high level of sophistication and a superlative battlefield or RTS game in its variants as selected by user menu choices.

# 8. Towards Fully Integrated Design – User Control

Battlefield / RTS design will produce much more interesting and playable games for the mature user if things like scaling, physics implementation, good modelling precepts and the question of catering to the full range of possible players receive comprehensive treatment. There is absolutely no excuse for negative factors like poor AI, poor path finding, poor formation handling, failure to cater for tactical rushers and strategic players in the one game model and permitting stupidity in autonomous unit actions.

Let us take one compounded example as an illustration. When poor formation options, poor formation handling and stupid unit decisions (when they act autonomously as singles or in formations) are inflicted on the player by the game design despite the player’s best efforts then this might merely indicate poor algorithms. However, it probably goes deeper and indicates a lack of thought by the designers about a key concept. This key concept we might term the “User Control Philosophy”. Before we design a user interface and user command functions we must develop a very comprehensive User Control Philosophy and then build the model to implement it.

Where is the player or user going to be placed on the spectrum that slides from total manual tactical command of every tiny action to the issuing of the broadest strategic commands only? Will the player be placed statically or allowed to move up and down this spectrum? Will the player’s movement up and down this spectrum be player managed, game engine managed or both (in some form)? Will all of this be managed “in-game” or will we give the player some ability to further tailor control settings to his preferred playing style in a game utility before and within live games? Such a utility could enable the setting of preferred defaults, preferred target orders, preferred formations and so on for different units; a kind of standing orders system in fact. A player could even be permitted to design his own formation types.

A proper approach to integrated design would take cognizance of the fact that integrated design is only as good as its weakest link. Let us look at a very appropriate example and relate it back to the User Control Philosophy. AI is Artificial Intelligence (so-called) which in this context means how the computer plays the game against a human opponent. However, in an integrated design the AI does much more than just manage how the computer plays against a human opponent. It will also manage all automatic actions within the Player Control Module according to the parameters laid out in the User Control Philosophy. That is to say when the combination of engine determined settings and user determined settings taken together indicate that a player tactical action is to be engine controlled then it will be engine controlled by the AI to the appropriate level of “intelligence”. Designed properly, this will do away with autonomously or semi-autonomously acting units belonging to the human player doing excessively stupid things that the human would never do if he/she was watching and managing every unit all the time. This stupidity moreover is something which the computer AI often does not inflict upon itself or which the computer AI instantly recovers from on its own behalf because it can “micro” so well.

These issues affect both player versus player and player versus AI games and become more and more important as unit numbers climb. As unit numbers rise, the game becomes less tactical and more strategic. As unit numbers rise, the volume of tactical actions we can reasonably expect the human player to ‘‘micro” (as a percentage of total human player “faction” actions) must necessarily decline. For all these reasons, the quality of the AI module and the concept design of the User Control Philosophy are interwoven. Similar arguments can be employed to show that path finding, flocking, swarming, formation movement algorithms and so on must all be of the highest order and all integrated into the AI module. From thence they are available to the computer for computer play and partially available to the human (mediated through engine control and user control decisions) to handle that level of ‘microing’ which the human never handles or handles progressively less of as unit levels climb.

# 9. How Humans Play RTS – Lessons for AI design

## 9.1 Compact Bases and Nodal Spread

We must consider the range of capabilities of the expert human RTS player and use this information to assist us in AI design. Thus the lessons of praxis, that is to say the lessons of how humans play RTS must be learnt, taken back and built into our high level concept design as this will enhance the whole design. Of course, this process is not a direct literal transcription of method in the case of AI. How humans play RTS teaches us many fundamental things about RTS but we must then use these insights creatively. It is certainly not just a matter of trying to get the AI to play exactly like a human. Such techniques do not translate in simple one to one correspondences.

The method for developing most successfully in the majority of RTS games can be explained by using the concepts of “compact base” and “nodal spread”. Cossacks, for example, rewards compactness but it is a special form of compactness which still achieves the necessary expansion to achieve a widening area of control. This process we can term “step-wise nodal spread”. Each resource-base node must remain compact and defendable. Each new node is expanded to or “jumped to” at the right time. Expanding or spreading is mainly about economic growth in RTS games though it is also about general territorial control. You spread out to exploit far flung resource sites in order to grow your nation or faction. We will bear in mind throughout this discussion that our design will seek to avoid the excessively expansionist and thus non-classical strategies of the tactical RTS style but expansion will still play a role.

“Step-wise nodal spread” has classical military application in terms of the theory of taking, holding and subduing an extensive territory. Open ground where one’s forces are vulnerable and which one must traverse whilst achieving no immediate tactical or strategic advantage must be crossed quickly. The rapid traverse might be undertaken to seize and hold an eminently defendable strategic bridge for example. The bridge is a key “node” which is “stepped to” quickly. Of course, rapid movement entails risks as well as potential rewards so before moving all factors must be considered including proper scouting.

There is a dynamic contradiction inherent in the “step-wise nodal spread” process for growth in RTS. The economy and infrastructure require a spread or dispersal to achieve growth but the military requires a concentration to achieve supremacy. (These general observations are true of the real world too although strategically speaking it can take years of conventional warfare to break down the production and manpower capacity of an extensive and powerful nation state.) The synthesis of these contradictions in practical game play strategy is often contained in exploiting the possibilities of step-wise nodal spread whilst maintaining some form of internal lines until one reaches a decisive engagement stage or a final siege stage. Spread confers other advantages like sight of the map and also perhaps some defence in depth. However, speaking militarily, excessive spread and thin external lines suffer from all the usual strategic problems related to the dispersal and dividing of forces. Sight of the map and defence in depth can be achieved in other ways.

Cossacks (like Age of Empires and Starcraft) has a peasant or peon collection style of economy. When a game needs peons as Cossacks does, the limiting factor in resource collection is how many workers you have and how fast you can grow the worker population. Spreading out very early is not viable as at that point in the game you do not have enough workers to properly exploit your immediate starting environment let alone exploit resources at further points on the map. This means that early on the Cossacks player can develop a compact and secure home base with a concentrated defence rather than seek to spread too early to other points where spare mine sites may be located. For its first defences, this early compact base can rely on various versions of internal or external lines. There are several ways to attempt such defences and these have been discussed elsewhere. (They are discussed in my manual on how to play Cossacks OC Mod.)

Games like Cossacks, AOE and Starcraft are very much about following this step-wise nodal spread plan (to achieve economic growth with military security) rather than following a continuous amorphous spreading plan. In Starcraft, the game theorists talk about “jumping to your natural”. This means expanding at the right time with a quick jump to your next natural resource point. On most maps, Starcraft has a close natural expansion point for new resources which is often called a “natural” for short. You have to grow your economy and military to a certain point to properly exploit and defend your first resource node at the home base before you “expand” or “jump” to a new resource node, attack the enemy base or prevent him from jumping to a new resource node.

Total Annihilation and Supreme Commander are a bit different in that workers (construction units) build structures but don’t gather resources except in a secondary way. The main resource generation is done by fixed structures like mass extractors on mass deposits. Mass deposits are often sprinkled widely across the map rather than being concentrated in clusters. This map design difference alters the emphases on various aspects of the “spread for resources / concentrate for military power” equation. It makes very early expansion both more necessary and yet more difficult to protect. However, the basic principles of good economic and military development, “spread for res and tight for fight”, and the fundamental dynamics of relating the two dialectically in winning play still remain the same.

# 10. A Proposed Advance in RTS AI

A way into understanding the theory of fully integrated design is to consider AI as a case in point. Now, I am going to be a purist once again and set a very high standard which is still a realistic and logically attainable standard in 2012 and beyond. If an RTS game developer builds any sort of cheat into the AI at any level then the project has already failed. Let me repeat that in bold type. **If an RTS game developer builds any sort of cheat into the AI at any level then the entire project has already failed.** In fact, the good developer should need to build in, as options, iteratively weaker versions of the AI to allow learners to middle level players to achieve some wins and some feeling of progress. Only the highest calibre players with a full tactical and strategic repertoire should be able to beat the best non-cheating AIs and maybe not even then in some cases.

Clearly an alpha-beta algorithm tree-search approach to RTS AI as a whole does not bear thinking about. I am sure nobody has been foolish enough to try. How big is the array of a decent 2-D RTS map let alone a 3-D map? How many moves occur when one considers that each frame is a move or a “ply” in digital move tree-search terms? Does this consideration mean that we have to immediately scurry to a Pandora’s Box of “cheats” to get RTS AI to have even half a chance? If you know RTS as a player or a programmer then you know the common drill. AOE2, on the higher difficulty settings, gave the computer AI instant resource bonuses after progression to each new era. Many other RTS AIs also use resource cheats. Many RTS AIs also use sight cheats. You are in the fog of war but they are not. They can “see” the whole map all the time so sneak bases and tactical surprises hatched by you are out of the question. Neither can you “wrong suit” the cheating AI as it knows what you are building and is already building the perfect counters in any scissor, paper, stone oriented game.

The following proposition will be obvious to any game programmer. A computer program can “not know” in one context what it “knows” in another context simply by using or not using certain data according to the routine. Thus a computer program can “know” the whole map to conduct overall unit position management but “not know” the whole map when conducting its own AI calculations. Thus there is obviously no necessity for the AI to cheat and see the whole map.

Current RTS AI, whilst often uncorking the most acute forms of early tactical viciousness (usually after a “pre-cooked” opening build-and-rush sequence), from then on often can’t plan or play strategically for nuts. Therefore we tend to see either undisciplined streaming or swarming relying on a better economy (usually better because of the resource and sight cheats) or some kind of scripted modular approach in the mature game stage which produces waves of “more of the same” coming at us like rows of spaceships in the old Space Invader game.

Why should serious RTS developers loathe and abhor AI cheats? One, they are unnecessary. Two, they distort the game. Three, they become predictable. Four, they make playing against the AI so different in character from playing human opponents that the former is useless as practice for the latter except for the rank beginner. Five, a cheating AI is much less useable for the purposes of integration with a User Control module based on a comprehensive User Control Philosophy.

## 10.1 Notes toward a Strategic AI (Resourcing)

Clearly the AI has some advantages over the human player. It can micro like crazy! (A proportion of its microing capabilities can and should be made available to the human player according to the tenets of the User Control Philosophy.) Of course, all the microing power in the world is useless if it is not intelligently directed. In the first instance, the AI should be a very good resource and economy player without cheating so the AI should suffer no disadvantage in this regard. Some RTS game AIs already show good resource/economy management seemingly without cheating and this is as it should be.

Resource or economy management (considered in isolation) is two things in RTS. Firstly it is accounting. The AI ought to be able to be a good accountant and thus be able to reckon up its income and expenditure allocations relative to its goals and requirements. This is vague at this point but will become clearer when we talk about goal-setting and requirement estimation. Secondly, resource or economy management in RTS is about “most efficient path analysis” or the “shortest solution”. This applies equally to early game build paths and late game build paths. It applies equally to physical paths (peasant paths, worker paths or unit paths for example), resource collection paths , economic growth paths, tech-tree paths and so on. In general, lower time costs in physical build and collection paths will equate to higher resource income growth in a given time span.

A game like Supreme Commander already explicitly shows ETAs (estimated times of arrival) on sequentially set (shift-clicked) build paths. This indicates such an engine already has available to it the necessary path data to start doing most efficient path analysis or shortest solution analysis. Supreme Commander also implements an “all-resource-sites-are-known” model at the start of the game despite the fog of war. If you have seen the “day-glo” display of resource sites in the fog of war in Supreme Commander you will know what this means. This is not a game model one would necessarily advocate but it assists our argument to consider it here.

This characteristic means that a Sup Comm-like engine could for example calculate the shortest path needed for the Commander to walk over the whole map capping the mass points with mass extractors. It could commence this process by trialling the expedient of walking each time to the next nearest uncapped mass point and capping it. This would be a reasonable heuristic for capping all mass points in the shortest possible time or in some time reasonably close to the shortest possible time. However, it is not an algorithm for capping all mass points in the shortest possible time. An algorithm for finding the shortest path in this type of problem can be found in the literature. No doubt it will involve a tree search, optimised by certain heuristics and iterative refinements of weightings.

However, in Supreme Commander or any RTS game, the problem (even the resource problem) is not so simple. In Sup Comm there is the issue of energy as well as mass. One begins by expending these resources in discrete output bursts of one, the other or both in order to build structures to gain a subsequent income flow of both. It is very disadvantageous to “stall” i.e. to get stuck taking a long time to build a mass extractor or an energy maker because of a critical bottleneck or lack of mass income and/or energy income. Thus amongst the numerous possible paths of building mass extractors (at set points) and energy makers (at discretionary points) there is a theoretical best path for any map which maximises the growth in both resource income rates.

Another characteristic of the Supreme Commander resources model is that is disadvantageous to “max out” as well as to stall. The mass store and energy store both have a finite upper limit (which grows with progress in the game) so income must be spent before maxing out or income is wasted like extra water over the spillway of an already full dam. The requirements to find efficient build paths and to avoid stalling and maxing out certainly complicate matters. To make matters even more difficult, factory building is possible and lesser builders can come out of factories and go off on their own build paths. However, all these factors quite obviously branch into a classic move tree amenable to a tree search algorithm supplemented by optimisation heuristics.

What is needed in this case is a way to do a tree search of limited depth to find the most efficient resource build path for the next x seconds of the game. This is much akin to doing a tree search in chess to a limited ply depth. The tree search must cycle iteratively through all economic builders and ask essentially “what can this build next?” It must test all the possibilities within a “gridded” or “tiled” time-space matrix (which will create an array of manageable proportions) and it must evaluate the results at the end of the tests down all the branches in the possibility tree and choose the highest scoring result. All this resource tree-search calculation is done without reference to enemy actions. It is a form of open ended goal-seeking where the best path to proximal goals (short-term, close at hand gains) if accurately calculated is usually a pretty good path (though not necessarily the absolute best path) to good long term growth.

# 10.2 Notes Toward a Strategic AI – Military Strategy

Since all of the above is done without reference to enemy actions and since it is all essentially defenceless resource production, you are entitled to ask “what about enemy action?” The very broad answer to this question in the first place is to reconsider the following fact about all RTS game strategy. **Resource production (i.e. economic power) requires a spread whereas military power requires a concentration.** We can leverage this insight with the application of military internal lines theory for strategic defence and “efficient force/reconnaissance in force” theory for raiding. In addition, we will seek to have the AI play for long term strategic military wins when early economy raiding and economy damage do not suffice to generate the win.

The AI will play for long term strategic conventional force wins by building an economy that is “good enough” and playing internal lines theory to maintain strategically accurate defence, winning battles in the field or “at the gates” of its own base or the enemy base by better concentration of force and thus bleeding (attriting) its opponent’s forces at a higher rate than its own forces. At times, the AI will retreat when outgunned (trading space for time), stage fighting retreats using delaying positions (hills and cover) and then re-massing forces closer to its own base (shorter logistical and reinforcement lines) to achieve greater concentration of force and make the opponent suffer from a long logistical tail and strategic attenuation of forces. All of this force behaviour should be achieved as much as possible by the weighting of factors in the strategic master algorithm rather than by additional heuristics.

This internal lines theory is a method of fighting which is empirically and scientifically based. It derives from the theories of Carl von Clausewitz. It is a method which eminently lends itself to quantification and thus to computer AI implementation. It is a method that in terms of both fluid swarming and precise formation play (as appropriate to the tactical situation) when managed by computer number crunching power will be simply devastating and demoralising for the human “victim” of the AI unless the player is a real expert. RTS computer AI does not need in–built cheats. It is just conceptual and design laziness to continue to adhere to such a view.

So-called “scissor, paper, stone” weapon relationship considerations will add great complications. This must be admitted. In addition, if it is a game which provides serious strategic weapons, the AI will have to “match the ante” on strategic weapons. This last might not be easy to build into an AI and will necessarily be game specific. However, the pressure the AI will be able to exert with classical war theory as outlined above, will reduce the AI’s vulnerability to strategic weapon surprise by reducing the time, resources and conventional forces buffer available to the opponent which would otherwise allow the opponent to generate strategic weapon surprise or strategic weapon over-trumping. We must note that “strategic” has two meanings in the context of this discussion. It can mean strategic weapons like nukes. It can also mean (more importantly for our purposes here) the strategic use of large force masses of the conventional type.

# Still Not Convinced?

If some are still not convinced that such an innovation in RTS AI is possible without any cheats then a little more concept detail is required. All RTS games have a pure economic opening phase. The duration of this opening phase time window is dependent on a number of variables but it can usually be estimated reasonably well. The RTS AI can essentially determine (mathematically of course), that it has x seconds of safety during which it can devote its entire efforts and resources to building an initial resource production base. After this x seconds of safety, the base becomes theoretically vulnerable to early (cheese) attack and thus needs some initial defence.

How is the duration of this initial safety window to be determined without recourse to AI cheats? Point one is that the AI is entitled to use all data that is potentially available to the well-versed and well-trained human player. A human player will use such knowledge with a “feel” for timing. The AI will do its mathematical calculations. Let us assume that the AI is playing a human on a well-known, predesigned 2 person map i.e. a map that comes “out of the box” with the game. The game might be something like Starcraft or Supreme Commander. A characteristic of pre-designed 2 person maps in such games is that the opponents start at predictable, diametrically opposite positions.

The AI can thus infer the following whilst not cheating by looking into the fog of war. Let’s assume the computer can talk to itself as it calculates. “OK, I am in the NE corner of a standard design 2 person map. Therefore the opponent is in the SW corner. I am inset 10 units of distance from the NE corner. Therefore I can assume the opponent is inset about 10 units of distance from the SW corner. This means the diagonal straight line distance between us is 100 distance units on the map diagonal – 20 distance units for insets = 80 distance units. I know from my opening knowledge of resource build paths (as I have an opening book dataset like a chess computer) that the earliest my opponent can get a basic offensive unit of type y out (because he is faction z) from a typical starting build path is 120 seconds. So I have to prepare for the possibility of the first raiding unit of type y arriving in 120 seconds plus traverse time. I can calculate the traverse time as I know the intervening distance, the speed of unit type y and I will modify this calculation by a traverse difficulty factor for the estimated difficulty of ground and estimated indirectness of the best possible traverse path.”

Let us assume that the AI calculates that the safe resource build time is 220 seconds plus or minus 20 seconds for estimation errors and unknowns. The estimation for error allowance may well have to be arrived at heuristically as would be the traverse difficulty factor. The AI determines it has 200 safe seconds before a military unit is needed. But the switchover lead time from pure resource building to military building/preparation and getting the first military unit made is (let us assume for argument) 20 seconds which the AI knows from its opening book. Thus the purely safe resource build start sequence ends at 180 seconds. This build start sequence may be an opening book sequence on a known map. Having an opening book is not cheating. Human players (if they are any good) learn and practice opening sequences for popular maps. During late phase game design, an opening book research utility with the AI already built can crunch away on predesigned maps working out the “opening book theory” for the maps. On unknown or randomly generated maps, the AI will use tree-search logic and heuristics to determine the best resource opening sequence given what it can “see” in terms of early available resource gathering sites. In any case, opening book knowledge will give it the first two or three builds automatically even on these maps and during these builds it has time to scout and calculate its next moves in the resource build tree.

Now comes the interesting part and it does have to be understood that this section of the paper is very much in the vein of notes toward a full theory. The RTS AI must seek to spread economically in a broadly logical, heuristically guided, tree-searching and goal-seeking manner. It must use some “pre-cooked” data in the form of opening book data and also tree searches to attain proximal resource goals most efficiently. This will be as part of an open ended goal-seeking strategy where the best path to the proximal goals (short-term, close at hand gains) is a good path to growing a long-term economy that is “good enough” to win with.

The difficulty of defending very early resource spread infrastructure and builders/workers varies according to game design and map design. It ranges from acute in Cossacks with full capture options on, to moderate in Starcraft to not all that difficult in Supreme Commander due to the great relative power of the Commander at the start. However, as soon as the base gets a little bigger and resource generation sites get more far-flung then defending the periphery of the resource spread can become much more problematic.

The key in the long run in AI terms is to use military internal lines theory as well as a few other stratagems. This will involve maximum use of the AI’s calculation ability and microing ability to;

1. Rapidly gain Intel on distant events (like sneak resource spread) by rapid active scouting;
2. Raid under-protected enemy spread (when detected) with sufficient force but no more;
3. Dispel the fog of war around the home base and well beyond it by evenly spread passive or patrolling sight-range scouts (assuming it is a large unit limit game in the many thousands);
4. Keep the main army mass or masses near the home base and covering it;
5. Be ready to dispatch the main army or sufficient detachments to meet incoming challenges;
6. Calculate the quantity and quality of incoming attacks or raids and thus despatch a superior force to overwhelm them quickly;
7. Calculate and update an estimated interception point and send the superior force to the desired interception point;
8. Always cover the main base by keeping the main army close to the main base and between the main base and any threatening main army;
9. Calculate terrain advantages and give weightings to defending on advantageous terrain.

The above are just some of the main points. What we will be making use of is the AI’s unparalleled ability to counter threats if it has adequate resources converted into an adequate army and also adequate Intel i.e. sight of the map. For the human player it will be a bit like playing computer ping pong against a game that is programmed to never miss the ball. Because it makes all the calculations perfectly accurately and on time in ping pong, it can never miss the ball. Thus an incoming detachment or army on attack (think ground armies for the moment) is in essence an incoming ping pong ball. The AI as defender must detach a sufficient force or “bat” to the calculated interception point to smash it. If the AI is built well enough it will be able to do this all day against an unimaginative opponent, batting away every attempt to deal it a body blow.

This is not a cheating AI in the traditional sense of cheating AIs. This AI will still be beatable in many ways. The chances of using indirection (e.g. feints) against it should be quite good if the human player is skilful and imaginative enough. We can see that this AI’s predilection to send a superior concentration of force to an interception point (say double the attacker’s force) could leave it open to falling for a feint and then failing to stop the main drive elsewhere. The AI would have to be refined further to detect quickly and respond to game junctures where there is just such a switching of the main incoming threat.

What this AI will do well is spread relentlessly, scout and raid unremittingly, defend its main resource bases doggedly and keep its main army numbers more nearly intact while almost surreptitiously bleeding the human enemy’s main army numbers faster. It will achieve this last outcome by winning encounter after encounter with superior concentration of force at the battle sites. Replacement rates in standard RTS designs cannot replace the attrition of a series of major clashes quickly enough. The AI’s more efficient use of the military will also stand it in very good stead on exhaustible-resource-model maps. The AI’s raiding forces will always be of a tactical raiding size not of a strategic size so that if trapped and lost they do not compromise main army numbers. It will be almost as if this AI’s main goal is to not lose rather than actually seek a win.

One of the key issues in programming such an AI will be to detect when and in what way to go for the win when the enemy has suffered sufficient attrition. One can also imagine how sapping it will be for any other than the sharpest and most imaginative of human players to play against this kind of relentless, tireless and inhumanly patient AI. One way for this style of AI to go for the win would be to have it fundamentally follow an inexorable boa-constrictor style to the last. That is, it would always be trying to incrementally increase its sight of the map, increase its safely held territory, keep its main army at the centre of its sight range “circle” (which would tend to see its main army gravitate to the centre of the whole map and radiate power from there) and take opportunities of “pouncing” on inadequate enemy forces and resource structures in the open whenever such opportunities present themselves. Furthermore, just as a chess program’s position evaluation function accounts every enemy asset as a negative in the sum score that it is programmed to maximise, so would this style of AI account every enemy structure as a negative which it would try to eliminate provided the value of units lost doing the job was less than the value of the enemy asset.

# A Simplified “Clausewitzian” AI - Proof of Concept Example

This section assumes a simplified proof-of-concept model with only two types of units, peasants and fusiliers (soldiers with muskets). Structures should give no sight range into the fog of war other than of themselves as it were. Only the peasants and the fusiliers would give sight range. The AI will seek first to place one fusilier on a point adjacent to each friendly structure to protect it from capture. It will always seek to do this as a minimum defensive posture. Whilst no enemy units are in sight, the AI’s military algorithm will then seek to maximise total sight range (dispel and minimise the fog of war) by marching all additional fusiliers as they are “built” (marching each of them the minimum distance required each time) one by one with each going to a point required to give a circular sight range that just overlaps by one pixel or “tile” the sight range of other similarly placed fusiliers. We can imagine the AI would eventually cover the whole map with units with slightly overlapping sight circles. However, it will desist doing this locally when it spots enemy fusiliers. That is to say, it will keep trying to expand its sight range elsewhere but will not march single fusiliers into engagement range with spotted enemy fusiliers.

If the map is very large the AI will start creating a centre of gravity army before covering the whole map with evenly spread scouts. There will be a rolling mathematical-algorithmic solution to this particular problem. The solution will be related to the distances and geometry of the situation, the traverse time that enemy formations must undergo and the actual scouting sight range required to adequately dispel the fog of war in the general region of the home base. In addition there will a mathematical calculation of what force size the enemy could dispatch as formations and when they would arrive even assuming best travel path and that the enemy is committing all available forces to the attack (and thus leaving his own base unguarded). Strategic attenuation, regarded here as the time it takes these forces to arrive, works in favour of the defender as he will have a larger force assembled locally (assuming equal growth rates) by the time the enemy arrives. This gives the defender the luxury of spreading some units as solo scouts whilst assembling the rest as a concentrated centre-of-gravity army. The geometry of the defence situation refers essentially to the position of the home base (supply centre of gravity) and the positioning of friendly defending formations in positions of mutual support. The essentials of the geometry will be to intercept the enemy formations before they reach their objectives by interposing the nearest friendly formations and then to delay the enemy with these holding forces while supporting forces come from each flank (usually) to win by concentration of force and cross-fire effects.

Once the algorithm has got close to maximum possible sight coverage of the map it will concentrate all further fusiliers in one compact main army of mutually supporting formations. Appropriate algorithms would be developed, refined and weighted by repeated testing in the development engine. This main army mass will be placed in the centre of the total sight coverage “circle”. This mass will move about to stay in the sight coverage centre as this centre may drift. The AI algorithm will do this subject to some more rules nominated below. If an outlying scout fusilier is threatened by a single approaching fusilier it will hold its ground. It has about a 50% chance of winning according to the game rules (some randomness in individual hits at range). If it wins, the military algorithm holds it there. If it loses, the military algorithm seeks to “spread” a new scout or maybe two to that vicinity. Always, the algorithm seeks to have as much sight coverage as possible without pushing forward (initially) to challenge spotted enemies. This approach is modified later in the struggle.

If outlying single AI scouts are challenged by a superior enemy force of 2 or more fusiliers they retreat towards the centre of sight coverage (where the main mass of fusiliers will also be accumulating once total fusilier numbers are sufficient). The retreat will always be managed by the AI such that maximum sight of “what’s coming” is maintained whilst staying out of engagement range. If the AI routine is written correctly it will also make other and new scouts adjust to cause its total “sight coverage” to flow around and “envelop” an incoming enemy fusilier group whilst keeping these scouts out of engagement range. Thus the AI will be able to “see” the incoming group, determine its strength and further to “see” if it is supported or not by reinforcements coming behind it. We can perceive here that the AI will envelop the incoming enemy detachment into its sight range a bit like an amoeba enveloping a prey particle.

This sight range envelopment in fact is preparatory to a military engagement or a full military envelopment. The AI will need to determine the objective of this incoming detachment i.e. it will determine where the incoming path appears to be heading and will determine an appropriate intercept point. The intercept point will be before any structure objective or key salient can be threatened or captured. The AI will send a force (say double or treble the size of the enemy force if possible) to the intercept point covering any structures which need defence. With enhanced routines, the AI would also be able to send a trapping or holding force behind the incoming force on its natural line of retreat. The incoming detachment will be annihilated and the AI will achieve a very favourable kill ratio due to the concentration of superior local force (at least 2 to 1). It will have higher local numbers due to the long reinforcement lines of the attacker. The AI will then begin to calculate how much it estimates itself to be ahead of the opponent in unit numbers given its better kill ratio and the estimated relative attrition-replacement rates for itself and the opponent. Based on this estimate it will become more aggressive but more of that later.

Will the average human player fall for this? Assuredly, the answer is yes. Average human players tend to “economy raid” in RTS as they have few military concepts. Their raids are often comprised of force levels that one would characterise as “reconnaissance in force”. They will be sending in a detachment to try and destroy a structure or kill workers. They typically do this without scouting properly as only the best players scout effectively. Thus average players will be decimated by this AI approach. Good players who do scout properly and only engage with superior force will find themselves continually countered and thwarted by this AI and thus forced into retreats and further manoeuvres. Eventually even good players will probably succumb to fatigue or impatience (or boredom as this is a simplistic proof of concept model) during all this manoeuvring.

The AI will need to have within its algorithm or algorithm set a method for calculating its resource centre of gravity and its military centre of gravity. The resource centre of gravity will be arrived at by an equation that is essentially a position evaluation function that assigns value weightings to resource structures depending on their productiveness. Based on these weightings, it will calculate a geographic resource centre of gravity. Initially, the military centre of gravity would tend to aggregate under this model (as outlined so far) at the centre of sight coverage. It may prove necessary to develop an algorithmic or partial algorithmic and heuristic method to modify placement of the military centre of gravity relative to the resource centre of gravity and relative to the “threat-direction” possibilities i.e. the potential avenues of incoming threats. (Obviously threats can’t come out of the map edges but only across open terrain.) This would have to be done in such a way as to ensure all incoming threat formations could be intercepted in time. To use another analogy, the main mass will act like a cover defender in rugby league or rugby union always tackling the enemy formation before it reaches its goal. Essentially, the designer could experiment with different weightings and models to find the optimum “dynamic geometry” of this system.

We still need to consider how such an AI could go on the offensive. Once it had calculated/estimated its numeric superiority was sufficient for a win it could seek to push forward. Exactly what is a sufficient numeric superiority for the AI model to commence pushing forwards would be arrived at in practice by continually testing and re-weighting the model. Single scouts pushed forward could seek sight of vulnerable groups. Sufficient superior force could then be detached from the main force or the main force itself pushed forward (but always in a covering manner) to again win battles a lower loss rate by the use of superior local force concentrations. In a very simple model like this it would be the speed and accuracy with which the computer AI could execute this process of endlessly probing, advancing, melting away in the face of superior force and advancing again at another potential weak spot that would wear out and defeat a human opponent.

In this simplified proof of concept model, what we are essentially demonstrating is a kind of “intelligent swarm” behaviour for the AI’s fusiliers. This intelligent swarm behaviour incorporates fusiliers who are designated as “scouts” and fusiliers who are designated as “C.O.G. soldiers” i.e. “centre of gravity soldiers”. More complex games will have several specialist scout types and several specialist C.O.G. soldier types plus formation options, cavalry and artillery. Naturally, a more complex game design with more unit types will complicate the AI issue and help the more adept, imaginative and theoretically sound human player to match the AI. The simple proof of concept model above should demonstrate convincingly that all AI problems are solvable in principle by algorithmic methods. In practice, limits on calculation time in complicated games will cause the need for some introduction of heuristic short-cuts. However, there is no reason why even games with complex unit sets and unit relationships cannot be given extremely powerful AIs under this general model.

Author: Rowan A. Pryor

Brisbane, Australia. August 2012.

Copyright – The Author asserts sole rights to this work.

# Appendix 1 –Summary of General Design Theory for Strategic RTS

## RTS Meta-Law 1 – The Meta-Law of Intrinsic Design Conflict

* **The requirements for modelling verisimilitude on the one hand and practical playability on the other are in direct opposition to each other.**

### Corollary of Meta-Law 1

* **How we creatively resolve this intrinsic design conflict will determine the overall shape and success our game project from high-level concept design right through to finished product playability.**

### Expansion of SML 1 - Realism, Scaling and Playability

* Reality is too big and too complicated to reproduce without scaling and simplification.
* The standard RTS modelling response has been to rescale, simplify and then make further relative distortions of space and time.
* Re-scaling space does not in and of itself change the time scale. This is a **key point**.
* Time is compressed deliberately in standard RTS designs.
* “Civil time” or “build time” is further compressed relative to “military action time”. In the real world it takes much longer to build a building than for a squad level fire-fight to occur. In the RTS game world it can take about the same time for these events to occur.
* Sight ranges and weapon ranges are often shortened and changed relative to each other.
* A purist or realist modelling solution would demand the removal of all these space and time distortions.
* Accurate space scaling and real time spans are needed for realism, especially for military realism.
* Greater realism of course equals greater game engine complication.
* This complication is no longer a computational or graphical problem due to advances in computing power, graphics and algorithmic solutions.
* Accurate space scaling, real time scales and the avoidance of space and time distortions when combined with large maps and large unit limits for a strategic game will introduce major playability problems. To put it simply, the game will take too long to play.
* Conclusion: A major design innovation is needed to implement a military purists’ solution to scale and time problems without introducing insurmountable playability problems.

# RTS System Meta-Law 2 - The Meta-Law of Realism

* **If you model with realistic accuracy then your model will behave in a realistic manner.**

### Corollary of Law 2

* If you model elements with unrealistic and arbitrary routines then unforeseen anomalies in the behaviour of the model may not always have a clear solution.

## SML 3 – The Meta-Law of Economy, Supply and Logistics

* **The System Meta-Law (2) of Realism requires that we accurately model the essential aspects of economy, supply and logistics.**

If we expand the need for supply, introduce some secondary and tertiary logistics and re-model the RTS economy a more realistically overall then we will move the emphasis of the game model from an artificial expansionist RTS strategy to a more classical military strategy model.

### Quick Explanation of Law 3

1. The environment (terrain, resources and weather) is the “given” or “independent” system.
2. The economy is dependent on the environment.
3. The real world military is dependent on the economy but in a special way.
4. Although in the real world the broader economy supports the creation of a military and sustains the military in the long term, it (the economy as a whole) does not directly mediate in the interaction between the military, the opposition and the environment in immediate military operations. There are two exceptions to this rule. The built environment as “terrain” is one exception and key infrastructure which is providing immediate supply and logistical support is the other. Apart from these exceptions, in immediate military operations the environment (in particular two key aspects of the environment namely terrain and weather) and the opposing military forces have a direct interaction quite apart from the economy. Immediate military operations are essentially dependent on the force elements present (friendly and opposed), supply logistics and the environment (terrain and weather). The above facts must be born in mind when trying to model RTS realistically on the real world.

# System Meta-Law 4 –Determinants of the Strategic RTS Game Model

(This is a 10 part Meta-Law which completes the Meta-Law exposition for RTS Game Systems.)

**Characteristics needed for a complete strategic RTS game are;**

1. **Large unit population limits.**
2. **Large maps. (Naturalistic terrain rather than artificial chokes.)**
3. **Linear growth style RTS economy and appropriate resource or supply points.**
4. **Naturalistic unit relationships only. (Definitely no magic units.)**
5. **Excellent path finding routines.**
6. **Excellent formation options, proper morale system and a standing orders system.**
7. **Naturalistic damage. Units usually die or are rendered ineffective upon taking a direct hit.**
8. **Structures and economy are relatively difficult to damage.**
9. **Military and economic units are suitably "intelligent".**
10. **Low overall requirement for rapid "micro mouse-clicking" in the game.**

The opposites of the above points define the more tactical RTS game.

The rest of this page is intentionally blank. See over page.

# High Level Design Goals

Expanding on the key design goals from the body of the paper;

1. The design will represent a paradigmatic advance in Battlefield / RTS.
2. The game will be truly strategic in style, scope and praxis (real play).
3. Battles will play out with complete historical military realism.
4. All phenomena of a physical and directly military nature will meet a “strong test” of optimised computational-realism. (See Appendix 2 for an explanation of this term.)
5. Economic modelling will meet a “weak test” of optimised computational-realism as the economic-civil modelling will be distorted as required to generate a strategic military game.
6. Whether supply and logistics fall into point 4 or point 5 (i.e. meeting a strong test or a weak test) will be decided during the design process.
7. All algorithmic solutions to typical problems of RTS programming namely path-finding, formation creation and movement, flocking behaviour and so on will be of the highest order and will thus meet the “strong test” of optimised computational-realism. (This point is implicit in point 4 in any case.)
8. AI (how the program plays the game) will be of the highest order and will use NO cheats of any kind. Instead, it will use a combination algorithmic-heuristic method based on the author’s “compact base - nodal spread” RTS theory combined with a Clausewitzian employment of classical military internal lines and external lines as appropriate to each game phase. The AI will also use its microing ability to scout widely and effectively, to maintain wide sight in the fog of war and to raid with sufficient force but no more.
9. The game program AI will also use no resource cheats. Instead it will use a combination algorithmic-heuristic method based on shortest path analysis and tree search algorithms always seeking best proximal (near) resource goals as a “good enough” path and very probably a “near to optimal” path to long term resource goals.
10. In the AI, economic-supply infrastructure will be fully defended by the pure military algorithmic-heuristic routines supplemented by further algorithmic-heuristic routines which calculate both military and economic-supply centres of gravity and thus maintain optimised defence posture geometry designed to defend the economic-supply centres of gravity with the main ground armies. This will entail the use of concentration of force, favourable terrain and the holding of key points, coupled with calculation of intercept points to enable superior friendly forces to combine and block mobile enemy formations from reaching their probable objectives and to destroy them with superior local force.
11. The AI and the User Control Philosophy (see body of essay) will be fully and properly integrated. Fine detail tactical control will be managed by the game engine and not by the player. This is appropriate for a strategic game. The game will be about overall strategic ability and tactical skills in using formations but not about fine detail tactical micro skills with individual units or small squads (i.e. not a rapid mouse clicking game).
12. A paradigmatic design breakthrough will be needed to smoothly combine real time strategic phases with real time tactical phases at a time ratio of perhaps 60:1 with seamless switching governed by the game engine according to a definition of significant opposing force elements (formations) coming into or leaving tactical range. Continued over...

# Basic Design Rules

The basic design rules include both absolute rules and rules granting some design process discretion.

1. There will be absolutely no compromise on accurate physical scaling.
2. Tactical time will be 1:1 (1 second in the real world equals 1 second in the game world).
3. Strategic time will be 1:60 (1 second in the real world equals 60 seconds in the game world).
4. An innovative “time engine” will manage time switches and overlay appropriate displays.
5. In tactical time, point of view of terrain and armies will be a standard ¾ eagle eye view.
6. In strategic time, point of view will be a standard ¾ eagle eye view with formation symbols.
7. Develop appropriate rendition scales for tactical and strategic displays in the design process.
8. Depiction style may be orthographic or isometric depending on design decisions.
9. Graphics in style and quality will approximate photorealism.
10. Do not use unrealistic modelling with pseudo-random functions to obtain outcomes.
11. Employ accurate physics, geometric modelling and probability tables to obtain all outcomes.
12. All algorithmic solutions must be state of the art.
13. The AI must play with no cheats.
14. The AI must use algorithmic-heuristic routines only. The AI must not use “blind scripts”.\*
15. A comprehensive “User Control Philosophy” must be developed.
16. The AI and User Control Philosophy must be fully integrated.

\*Note: There is a great difference between “blind scripts” and algorithmic routines optimised with heuristics. A “blind script” is non-responsive to the opponent’s actions and innovations. Opening book sequences, whether in RTS or chess, are blind scripts. These blind scripts are acceptable and even useful if the opening theory is well established and sound. If an innovation occurs in opening theory then both book theory and scripted moves in a program would have to be updated by hard coding i.e. by printing a new book or coding a new opening sequence into the program and releasing a patch.

Continued over...

# A Note on Laws, Goals and Design Rules

* Laws describe the invariable relationships in a system. Hence Laws are unchallengeable. Computer games are mathematically and algorithmically governed systems models of the real world and as such they have invariable relationships both internally and with regard to the real world which they model. These relationships can be described by Laws.
* Goals are objectives set by us. The Design Goals should be ambitious if we desire to make a paradigmatic advance and transform the field of RTS.
* Design Rules are made by us. Design Rules must set high standards in conformity with the Laws of RTS in order to reach the Design Goals. If the Design Rules are met then the Design Goals very likely will be met. Once the Design Rules are set they should remain set. Design imagination and design rigour are needed to keep the Design Rules inviolable whilst attaining the Design Goals.

End of Appendix 1.

# Appendix 2 – Further notes on Law 3 – The Law of Realism

## Newtonian Physics and the Modelling of Mass, Force, State and “Attributes”

Modern games physics engines do not always appear to implement full Newtonian physics. The clearest examples are discrete physical bodies not fully occupying the space appropriate to them. By “bodies” I mean not just human bodies but all discrete physical bodies or objects both animate and inanimate excepting contiguous landscape. There is often a deliberate design disconnect between the graphic of the object and the space occupied by the object mathematically. In terms of the shape and space occupied by the object polygon, it is often simpler and smaller than the graphical depiction implies. Either this is the case or object polygons of different objects are permitted to impinge and overlap by the game engine. This is done to lessen computational loads, to ease path-finding problems and to reduce “logjams” of crowded and crushed units.

It must be admitted that here (as in all other aspects of design) the demand for full realism is in conflict with “computability” and “playability”.

 **The touchstone for realist design must be that the disparity between graphical depiction and polygon calculation (in the form of allowance for polygon-occupied space overlaps) is just below the threshold of human perception. This we must regard as the computational-realism optimum.**

That is to say, the player should never perceive that two graphics or sprites have impinged on each other in any manner such that a part of each appears to occupy the same space at the same time. Graphics or sprites will always then appear to behave in a fully realistic manner and the general physical and graphical behaviour of the game model will be an excellent approximation of the real world.

The physics model also needs a model of states and attributes. A logical category structure needs to be deduced, developed and adhered to for the entire project. The Newtonian physics model will deal with mass and force. Mass (material) then must be categorised as solid, liquid or gaseous (state). Appropriate physics models for each state of matter must be applied, for example, to allow all solids to be fully deformable under force. Solid masses must have further attribute categories. For example, solid masses may be “very hard contiguous” (rock) “hard contiguous” (wood), “glutinous contiguous” (wet clay, flesh), “friable contiguous” (earth), block form (large block masonry), large aggregate (broken rock), small aggregate (gravel), small-grained (sand, gunpowder).

With appropriate physics and appropriate computational shortcuts and approximations, a deformable landscape design can thus be achieved. For example, a theoretical 1 metre cube of dry sand sitting on the ground as a perfect cube the instant before it slumps does not have to be treated as “n” grains where “n” is a very large number. It is simply treated as a solid volume of attribute “small-grained” with a certain slumping coefficient. With the appropriate slumping coefficient applied we can calculate the shape of the cone it will slump into. The actual slumping process may appear to be tricky to run but again imaginative design can assist us. To get the perfect dry sand cube (initially in stasis as an idealised example) to slump progressively into a cone we might do the following.

Step 1, calculate the cone that will result from the slump. Step 2, arbitrarily and temporarily treat the cube of sand as a liquid with a viscosity appropriate to its slumping coefficient and slumping friction. Step 3, compute its flow as a liquid according to our physics rules for liquids, the viscosity assumption and the force of gravity. (Move a set array of surface and embedded polygon points in the liquid body according to the relevant laws.) Step 4, we “flow” the cube into the cone by making a one-way permeability assumption about the cone boundary. Namely, we allow the “sand-liquid” to flow into the cone down through the cone boundary but not to flow back out of the cone around or above the base. Therefore we allow the “sand-liquid” above the cone’s sloping surface to drop straight down under the influence of modelled gravity and only to be arrested once inside (either by the cone boundary or the “floor” of the cone or by other material already in the cone).

In this model we might or might not implement accurate gravitational acceleration and we might or might not implement some calculations of momentum effects. Some simplifications might be needed as momentum effects, for example, would change the final outcome of the cone boundaries. These simplifications might well be acceptable as we could still get a realistic (and one must say) fairly rapid slumping effect. Again, the touchstone for realist design would be that the disparity between model behaviour and real world physical behaviour is below the threshold of player perception; in this case a player who is a layperson when it comes to physics. This again we can regard as the computational-realism optimum. Such slumping effects and partial cone effects can then be used to generate part cone slopes of rubble accumulating below bombarded breach points in masonry walls. Rubble (depending on average unit size) will have a different slumping coefficient to sand.

The “holy grail” in this area of design would be to create a fully integrated logical model to allow deformations of all materials to be modelled according to the laws of physics but to be so modelled elegantly and economically with relation to computational requirements and to meet the test of all inconsistencies being below the general threshold of ordinary player perception. Military physics phenomena (such as the effectiveness of enfilading fire or the ballistic trajectory of cannon balls) would have to meet a higher test of all the inconsistencies being below the general threshold of a military professional’s perception.

## An “Attributes” Model

Determining an “Attributes” model for all material would not be difficult. Amusing as it might seem, solid masses could conveniently and usefully be given the attributes of animal, vegetable or mineral. The object’s behaviour both under physical forces and in terms of autonomous or non-autonomous behaviour could be derived from this and other attributes. Other attributes would exist. For example, a further important attribute would be a flammability index. This could be a graduated scale running from 0 for water or “saturated with water” to say 32 which would equate to the explosive chemical reaction of gunpowder. Then, the spread of fire or explosion in any part of the terrain model or built environment model can be modelled via a contagion model using flammability indexes, adjacent flame or heat factors, cinders, wind direction effects and so on.

A full attributes model would help determine all behaviours and interactions of matter. Again the touchstone rule would be to get materials to behave such that the disparity between behaviours depiction in the model and behaviours in the real world is below the threshold of player perception. This should be possible for all materials and autonomous entities except probably individual modelled humans in the game. Even the humans’ en masse behaviour in formations in combat should meet this test.

## Sundry Notes

* Control systems must be appropriate to what is being controlled.
* In some current game engines (games on the market and legacy games) units sometimes oscillate looking for a path through obstacles and thus they get caught in an endless loop. This occurs when the path finding algorithm keeps “changing its mind” about the best path. To prevent such cases it should not be difficult to design an adjunct procedure to the algorithm which will “short circuit” the oscillation and make the unit “decide”. This adjunct procedure could be invoked after three “changes of mind” which are taken as the “flag” that the unit is in a path-finding oscillation loop. What constitutes three “changes of mind” can be precisely defined as three repetitions along a path which process does not bring the unit any closer to its final waypoint. I may be behind the times here. This issue may well be resolved in modern path finding. My crucial point is that path finding, unit “intelligence” and so on must all be of the highest order in such an ambitious game design.

# Appendix 4 – Discussion of Exponential and Linear Growth Models

In terms of unchecked growth, the natural model from the real world is that of exponential growth. Despite this, I recommend a linear growth model to achieve a strategic RTS game model. The detailed practical reasons are given in the body of this paper. The proof from first principles is as follows. In the long term, population and economic growth of human societies has empirically proven to be (in the main) exponential. This will remain true until the limits of growth are reached in a world of finite size and limited resources. As an aside, I shall note without further comment that we are getting close to these global limits now. However, in the time scales modelled in RTS (hours and days or maybe a few months in a short campaign model) any real economic growth rate would be approximately linear. That is to say, a very short piece of a long growth curve taken and magnified will appear almost straight. Also, the raising and deployment of a large army may mount at an exponential rate in its early phases but once supply, transport and infrastructure capacity constraints (bottlenecks) are reached the growth in the deployed army is likely to be roughly linear from that point and will finally plateau.

An RTS game model if modelled with fixed costs for productive infrastructures and an unlimited resources will naturally generate exponential growth. If we want linear growth, we must build some growth restraint or capacity restraint into the model. One way to do this is do what the Cossacks model has done. Cossacks has implemented an “inflation” model. Structures like Town Centres and Barracks (which directly produce peasants and soldiers) have their costs inflated for each subsequent building. This means early buildings are cheap (so the game starts up quickly) but later buildings become more and more expensive. As the building costs inflate exponentially, this pushes the growth model back to the linear form.

This “Cossacks” method as implemented in a further modified form in the “OC Mod” has some odd effects. One odd effect is that when buildings are lost then rebuilding becomes cheaper again. This favours the player who is on the back foot and suffering damage. By making his buildings cheaper again, the model allows him to make up his structure and production deficits somewhat more easily. The effect is that a player with partly damaged infrastructure can prove sometimes to be inordinately difficult to put out of commission or stamp out. When this factor is combined with the classical advantages of defence and a self-contained base with no supply problems then the defence is arguably too heavily favoured and protracted games can result. A way around this could be to never allow building costs for an individual player to deflate once they have inflated. This would strongly punish infrastructure loss and aid the attacker.

Much thought needs to go into the precise method which will be used to force a linear growth model. The chosen method needs to be integrated with all other design elements to ensure a properly paced game. The game must be balanced for attack and defence and must be terminable ultimately by the use of traditionally successful siege methods. The precise model chosen will affect the overall dynamic balance and feel of the game.

Part of the dynamic balance in any game can be called a “hill balance” or a “valley balance” (or perhaps sometimes a “plains balance”). What I mean by this can be illustrated as follows. A “hill balance” game would be like two men fighting at the top of a hill to roll a large stone down the other man’s side. As soon as they get off the top of the hill the game favours the attacker or one in the ascendancy. A “valley balance” game would be two men fighting at the bottom of a valley to roll a large stone up the other guy’s side. As soon as they get off the floor of the valley the game favours the defender. In example one, the game inherently wants to roll away from parity. In example two, the game wants to roll back to parity. It is probably best to avoid the excesses of both models. Traditional exponential growth RTS games usually exhibit too much a “hill balance” character.

# Appendix 5 – Economy Realism in RTS

It is not possible to realistically model a real nation economy in RTS. Economies use a vast range of resources and this long list must be simplified to a handful of key resources in RTS. Typically two to six resources are used in most RTS models. More fundamentally, economic time is much more sedate than military action time. In a nutshell, we can characterise this by saying that what takes months or years to build can be blown up in days, hours or even seconds. This means that if the RTS economy incorporates civil building then an extreme compression of civil build time versus military action time is forced on the designer. Even a two-scale time engine at the recommended 60:1 will not really solve this problem.

I would like to see RTS design seek a creative way around this conundrum. It ought to be possible to make the “economy element” of RTS feel like RTS economy play to the seasoned player whilst actually designing it as a deployment, supply, and military construction phase. Instead of building a civilian township as in AOE2 or Cossacks or a colony as in Starcraft, the player could actually be building a military encampment and defences in the field. Instead of economic growth, the player could be raising and deploying a dispersed army from town billets and moving it onto active duty in bivouac. The scenario could be of a section of frontier or border where events escalate rapidly from a sleepy but false peace to open hostilities.

The growth in forces generated in “standard” RTS could be mimicked by the growth in active forces as combat ready elements, general elements and reserve elements are progressively raised from a large town or city and put into the field. The proper deployment of HQ units, military provosts to raise the billeted soldiers, engineering elements for bivouac and military works, quartermasters, field kitchens and supply trains would occupy the player in the early stages. These tasks would be made no more onerous but not much less onerous than construction and expansion tasks in standard RTS. We would keep in mind the idea that the player can be required to do micro tasks early in the process but that he should be progressively relieved of micro detailing as the mobilisation proceeds. An example would be that initial orders are to an inner corps of HQ officers and adjutants. As the mobilisation of a full HQ occurs then automated (AI) quartermasters begin organising the supply trains. However, orders must still come from the top about where to deploy to, what roads are to be militarily secured for supply trains and so on.

# Appendix 6 – Military and Philosophical Quotes

1. “Fighting with a large army under your command is nowise different from fighting with a small one: it is merely a question of instituting signs and signals.” – Sun Tzu
2. “There is no higher and simpler law of strategy than that of keeping one's forces concentrated." – Carl Von Clausewitz
3. “To command nature you must obey nature.” – Francis Bacon

Quote 1 was chosen with the “User Control Philosophy” in mind. That is to say, playing a computer game with large armies should be no more difficult (in a manual control sense) than playing a computer game with small armies or even squads. This will be true if the user interface controls (the “signs and signals”) and formation options are properly designed and integrated with all other game systems. Of course, playing a large army game will be more difficult in terms of formulating and carrying through an overall strategy.

Quote 2 was chosen for its obvious applicability to an essay about strategic game design. The lessons of playing truly strategic RTS in practice have certainly convinced this author of the profound truth of Clausewitz’s saying. Time and again one can get too wrapped up in lesser tactics and in pursuing lesser goals (i.e. getting lost in the detail). Finally, almost every defeat can be traced back to forgetting this fundamental law in one way or another. This law is also applicable to project development. Minds must remain concentrated on the overall strategic goals of the project whilst simultaneously executing design detail.

Quote 3 was included because of its applicability to systems modelling design. We can understand Bacon’s insight with a physical example. Imagine that engineers want to divert a river while they build a dam. Diverting a river is “commanding nature”. One way to do this is to excavate a diversion channel. However, the course of the channel must obey the laws of nature, specifically the laws of gravity and hydrodynamics. The channel must have a downhill slope. Alternatively, large pumps may be used if there is no other engineering solution. The river might be in a deep gorge. In this case it requires energy to drive the pumps and lift the water. Nature must be obeyed, energy as exergy (energy for useful work) must be supplied. Similarly, one must obey the fundamental laws of RTS to design a good RTS game.

# Appendix 7– An Approach to Naval and Air Power in Realistic RTS

A fruitful way to commence dealing with naval and air power issues in realistically scaled RTS would be as follows. Tackle the problem first by taking a small, local theatre approach. The World War 2 battle for Milne Bay in Papua New Guinea would provide a good model. The action takes place in a coastal strip on a map measuring 20 kilometres by 10 kilometres. In this area the Australian forces and facilities comprise several brigades and battalions, a HQ and Brigade HQ, 3 airstrips, a small group of Kittyhawk aircraft and various trucks and machinery. Japanese forces land by barge (disembarked from a naval group) and comprise Infantry, stores and light tanks. One Japanese ship may provide close coastal support for some landings.

This would make an ideal scenario for realistic action of the battlefield genre rather than of the classical RTS genre. The speeds, distances and time-spans involved could be handled by my recommended 60:1 strategic-tactical time engine. Construction, repair, resupply and formation movements happen in strategic time illustrated on the strategic display with formation symbols. The engine switches to the tactical display and 1:1 tactical time whenever significant force elements engage.

Handling the Australian side and the Japanese side would take somewhat different skills. One can see that the force asymmetry lends great interest to the scenario. The Australians have first occupation of the ground, some entrenched positions and air power from the local airstrips. The Japanese have a larger and more powerful infantry force if fully landed and the advantage of light tanks. The Australians for example become very vulnerable if they lose the airstrips or the heights overlooking those airstrips. The Japanese are very vulnerable to air power during the process of landing men, equipment, stores and tanks and before they get under jungle cover.

Just for the historical record, the Australians won the Battle for Milne Bay demonstrating great valour and tenacity in the process. The Japanese were formidable and seasoned jungle warfare exponents but suffered in particular from being caught out at least once by the Kittyhawks during landing operations. The entire Papua New Guinea campaign from Kokoda to Milne Bay to Gona would provide sufficient scenarios for such a WW2 battlefield game. A good reference is the book entitled “A Bastard of a Place” by Peter Brune.

# Appendix 8 – The Gathering Equation

A peon (resource gatherer) usually travels between 2 points. Point A is the Storehouse and point B is the resource node. Typically, peons in RTS take a finite time to travel A to B and B to A. The peon also takes a finite time to “harvest” the resource and gather a full load to carry. At the other end (the storehouse) the peon typically offloads the resource instantly. Don’t ask why this is so. It is simply the standard RTS convention and it does not actually matter in terms of developing a simple gathering equation. If we wish to, we can simply regard the gathering time as including (mathematically speaking) the offloading time as well. Where there are peons supernumerary to an optimal gathering density, then some peons typically have to queue before gathering. If offloading times were explicitly part of the model, then offloading queues might also exist in some circumstances. Then a whole set of issues and circumstances would determine whether gathering or offloading was the limiting bottleneck.

However, we are going to assume;

* a finite time to travel A to B (*a*)
* no queue time to gather
* a finite time to gather x resources (*b*)
* an equal finite time to travel B to A as to travel A to B (a)
* no time to offload (instant offload)

Therefore time taken to gather *x* resources by a peon in one cycle is

T = 2*a* + *b*

Note: This appendix is incomplete at this stage.

END OF DOCUMENT