CHAPTER 1

Decoding and Foreseeing

> … we looked back at our previous games and realized that our solo campaigns have never prepared anybody for an online experience at all. That never worked, right? We always sort of touted it that way — "It's going to prepare you" — but it never really did.
>
> — DUSTIN BROWDER, LEAD DESIGNER OF *STARCRAFT II* (QUOTED IN REMO 2009)

Two Conceptions of Game Time

If competitive play often leads to epic confrontations on the battlefield, sometimes the fight is quite short. During the second iteration of TeamLiquid StarLeague (TSL) in 2009–2010, the American Terran player Gregory "IdrA" Fields was matched against the American Protoss player Tyler "NonY" Wasieleski (nevake 2010a). IdrA had won the first game and started the second one by building a relatively fast second Command Center with one of its SCV—Terran's worker unit—to have an economic advantage. Each player usually sends one of their first worker unit to scout, and that is what NonY did. But this worker unit also frequently annoys their opponent by attacking their counterparts while they perform their normal tasks—a tactic called "harassing."

And thus, NonY harassed IdrA's SCV building its Command Center to try to disrupt his attention. IdrA decided to switch the SCV building the structure: the normal maneuvre would be to click on the SCV performing the task, and to click ESC, then the next SCV could take when the first left. Unfortunately, IdrA misclicked: he clicked on the Command Center rather than on the SCV before hitting ESC, and thus cancelled

the construction. He lost 100 minerals, but more importantly lost precious game time since he would need to start over the construction. To the surprise of the casters, IdrA then precipitously but irrevocably typed "gg" (for "good game"), losing the second game of the match.

While in a lot of sports, every second counts and can be the one where a goal is scored, in *StarCraft* competitive games the first seconds of the game have repercussions on every second of the rest of the game, since its game economy is based on growth. *StarCraft* has a strong positive feedback loop regarding time: the beginning of the game is crucial to determine the endgame. If the experience of video games tends to be expressed in terms of space, competitive real-time strategy games should definitely—as their name suggests—be considered first and foremost as temporal experiences. The value of the first seconds is higher than that of the last seconds of a game.

This conclusion might seem counterintuitive, since a player can have a lot of time to recover from an earlier mistake. It is also not true of every *StarCraft* experience; it is mostly true in competitive or skirmish games. It is what makes the difference between what I will call the *decoding* paradigm and the *foreseeing* paradigm.[1] The information management of *StarCraft* is wholly different when playing a solo battle or in the campaign—in the *decoding* paradigm—or when playing an online battle—in the *foreseeing* paradigm. As we will see, each of these paradigms have a history of their own.

The very definition of strategy is different depending on the paradigm considered. The decoding paradigm is when the player must detect tendencies in the actions of the artificial intelligence opponent (called "computer" in the game) to anticipate future actions. They cannot be certain of the exact series of actions that will unfold or be possible; often, one can be sure that some actions, while theoretically possible, will not take place since it is not a tendency for the AI to do so. The decoding paradigm is the player "guessing" what the "rules" of the games are.

Most contemporary multiplayer strategy games rather fall under the foreseeing paradigm. Game actions are literally "foreseeable." Every possible action in the game rules has its own prerequisites and players can easily manage to either read about them beforehand in the game manual

1. I used to call these two paradigms "decryption" and "prediction" to translate the French words "décryption" and "prévision" (see Dor 2014a). But I sense these words do not induce any clarity. I must thank Bernard Perron for suggesting the new translations and using them in his work (Perron 2018, 113).

or try them in a custom game. Just as every player knows that a poker game has four Queens, they know that training Arbiters implies the construction of a Citadel of Adun, a Templar Archives, and an Arbiter Tribunal, and thus can anticipate Arbiters if they scout those buildings. The foreseeing paradigm *starts* when the player knows the rules of the game.

Real-time strategy (RTS) games before 1998 were often games solely inscribed in the decoding paradigm, and their legacy lived on in *StarCraft*. By being an accessible and competitive multiplayer game, *StarCraft* also built on games from the foreseeing paradigm: you can foresee your opponent's actions by knowing their gaming possibilities and reading their strategy. Both these paradigms are important when considering the historical importance of *StarCraft*. The goal here is thus to see how the emergence of the foreseeing paradigm in the 1990s reaches a landmark with *StarCraft*. This chapter describes the historical background from which both paradigms of RTS gaming emerged.

The first section will describe the core rules of *StarCraft*, applying to both paradigms. The second section will describe the history of decoding, showing how Westwood Studios' games since *Dune II: The Building of a Dynasty* (Westwood Studios 1992) have influenced the RTS genre. The third section will show, on the other hand, how multiplayer games such as those from Ozark Softscape played a significant role in the history of foreseeing, until modem play became more widespread. I will argue that *StarCraft* in the history of gaming is the main point of convergence between the decoding and foreseeing paradigms.

A Classical RTS

StarCraft is a classical RTS game and a common point of comparison for other games in the genre. The player must collect two types of resources (minerals and vespene gas), create and manage buildings and units to destroy every building of their opponents. The player clicks on their units to give them specific orders (move, attack, patrol, etc.) or use their special abilities.

Each unit is a "type" following what game designers Andrew Rollings and Ernest Adams called a tile-based aesthetics (2003, 340): 12 Marines will be represented as clones, with the same figure on the interface, the same voice when responding to orders, the same morphology on the game space. This aesthetics makes the game space easier to understand cognitively and strategically: the same unit types will have the exact same properties. Players also have an attributed color so that their units and

buildings can be easily recognized. The game is set on a pre-created map that determines the topographical elements of the game: starting and resource locations, bridges, cliffs, ramps, etc. It is a quintessential example of why Pascal Garandel argues that video game space is first and foremost a means to an end, almost every element of it being geared towards play (2012, 131).

Each unit, building, technology, or upgrade costs a certain number of resources—minerals and gas—spent once. Worker units collect resources by moving back and forth from the main buildings to mineral patches and vespene geysers. By training more workers, the collection process will be faster and, eventually, the player needs to build a new main building near another resource location: an expansion.

Buildings have different functions: to collect resources, to train military units, to research upgrades and technologies, or to fight. For every faction, buildings are organized in a technology tree: creating a building will unlock other buildings, but also units, upgrades, and technologies. For instance, if a Zerg player wants to build Mutalisks, they will need a Spawning Pool, then to upgrade a Hatchery to a Lair, which will let them build a Spire needed for this flying unit (Fig. 3).

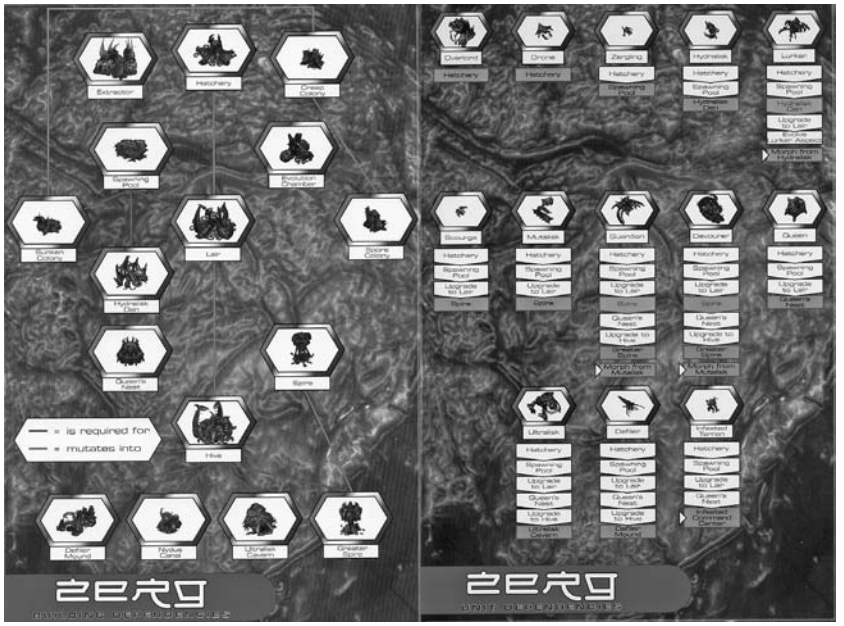

Fig. 3. Zerg building and unit dependencies, part of their "technology tree," as annexed in the game box

StarCraft is a "real-time" strategy game since players perform their actions simultaneously. The units and buildings respond to player-issued orders through mouse clicks or hotkeys; the action then takes a certain time and the unit or the building accomplishes it autonomously. The player can then manage something else during that time. They must thus use their time efficiently, trying to time each action without having to wait for a certain task to finish. Units are either ground or air units. Their weapons can be melee or ranged, and attack ground, air, or both unit types.

The actions of the opponents are hidden from each player; they must explore and periodically scout their opponents' bases to know what they are doing and where their units are. *StarCraft* uses the same kind of fog of war as popularized by *Warcraft II*. The game space is initially hidden to the player; they must scout to discover the topological elements of the map. Friendly units must be around enemy units to reveal them; otherwise, they remain hidden (in the "fog of war") (Fig. 4).

The Decoding Paradigm

The *StarCraft* campaigns are a series of levels (or "maps") to complete linearly. Each map is contextualized narratively and gives a set of predefined units, buildings, and topography. The player must fulfill specific objectives, stated in a mission briefing: in the "Norad II" scenario, they must bring a hero and two dropships at the site of a crash to rescue the survivors. In other scenarios, they must "survive 30 minutes," "protect a building," or simply "destroy all enemy buildings." Rather than having to create their buildings and units, computer players already have their own bases and will mainly react to the player's actions and execute prescripted moves. In the "Norad II" example, the Zerg computer player already has plenty of Spore and Sunken colonies protecting the site of the crash.

The campaign is a clear and strong example of the decoding paradigm: the goal is to decrypt the opponent's patterns or scripts and respond to these actions; the player "decodes" patterns in a figurative sense. The interest of decoding is to offer a challenge to a single player. They must somehow find an efficient strategy to overcome each obstacle or each map one after the other. In most maps, the equilibrium to reach is between having an efficient resource-collecting flow and defensive or versatile military units.

Fig. 4. The “fog of war” principle. Top, a hidden location. Middle, a location actively revealed by friendly units. Bottom, a previously explored location without active friendly units

In "Norad II," the player's buildings are already under attack, and they must be defended. Once the first blitz is countered with our initial military units, as Bart Farkas underlines, "there's only a short period of time for you to get your defenses back up before the next onslaught" (Farkas 1998a, 102), while "six to ten SCVs to start gathering resources" should be enough (103). Building a too large amount of worker units could lead to insufficient defense.

In the decoding paradigm, the opponent acts in a precise way "encoded" in the game. In a successful game design, the player can anticipate their opponent and respond efficiently; this optimization characterizes this paradigm. By trial and error, or by intuition or habits, the player will know what to do and not to do. But in some cases it is not predictable at all:

> Whatever you do, don't attempt to build any units near Norad II, and don't attempt to launch any attacks from this position. If you do, you'll bring the wrath of the Zerg down upon you, and the mission will be over.
>
> (Farkas 1998a, 101)

Of course, there is no narrative nor strategic reason for the Zerg to not attack the crash site immediately. When the opponent starts with four bases and an army size at maximal capacity, they would quickly win, but the game would be too hard: the computer will never do that. The rules by which the AI plays must be decrypted.

In the decoding paradigm, a player never "knows" what an opponent *can* do; they must guess what it will do based on their previous experience of the game. Using the terminology of the pragmatic philosopher Charles S. Peirce, a player will forge a habit of mind, "[t]hat which determines [them], from given premisses, to draw one inference rather than another" ([1877] 1991, 147). In strategy video games, one could talk of *strategic habits*: players forge inferences by observing their opponents acting or reacting similarly under similar circumstances.

In the decoding paradigm, these *strategic habits* are forged through experience with anterior enemy actions. For instance, players will know how to counter Hydralisks or Mutalisks, provided they have seen them before. They could also forge more general patterns of mind, observing for example that AI units tend to attack in small squads on base entrances. The player will play similarly when similar circumstances happen again: they are perceived as specific occurrences of a same phenomenon. In a case when they failed to predict the algorithm, or the opponent unleashed an attack larger or different than expected, they

can simply reload the game from a previous state and try again to refine their habits by trial and error. Strategic habits of the player (how *they* will act predictably) are forged according to strategic habits of the AI—what players observed and how they forged inferences.

The player is a privileged agent in the dynamic of each strategy: it is important that their experience is interesting and that they have a chance of winning. As such, the decoding paradigm is reflected in a specific conception of game balance. In *Fundamentals of Game Design*, Ernest Adams states that a game is balanced when it provides meaningful choices and when it places player's skills as a main criteria for success (2014, 404). In a decoding paradigm, game balance is to be evaluated in a player-versus-environment setting (PvE): the game is said to be balanced when there is a stable difficulty level and when the player's enjoyment is maximized (Adams 2014, 418). The goal is not that each player (human or computer) has an equal chance to win the game, but that the privileged agent—the human in front of their computer—has a satisfying experience, whatever that means. It is more similar to Roger Caillois' *ludus* than *agôn*: "The difference from *agôn* is that in *ludus* the tension and skill of the player are not related to any explicit feeling of emulation or rivalry: the conflict is with the obstacle, not with one or several competitors" (Caillois [1958] 2001, 29).

To borrow a concept from game theory, the decoding paradigm is a "mixed motive game."[2] The AI does not have any interest whatsoever in winning, except if it makes the game interesting for the single player. The human player can have the goal to win the game, but the computer players aim to deliver an interesting experience for the human player(s). Whether the AI plays like a human or not is only a secondary goal.

The heuristic circle of gameplay that Bernard Perron put forth (2006, 66), inspired by the work of the cognitive psychologist Ulric Neisser (1976, 112), is appropriate to describe cognitively the player's experience in decoding mode (Fig. 5). To explain the perceptual cycle, Neisser uses the concept of schema, which is internal to the perceiver and "directs movements and exploratory activities that make more information available, by which it is further modified" (1976, 54). The player perceives images and sound from the game (new state), which will help to select the correct schemata and direct sensori-motor action that will act on keyboard and mouse. For example, hearing the voice of the adjutant robot

2. Elizabeth Bruss (1977, 159) adapted this concept to illustrate the relationship between author and readers in literature; Bernard Perron (1997, 234) adapted it similarly for films.

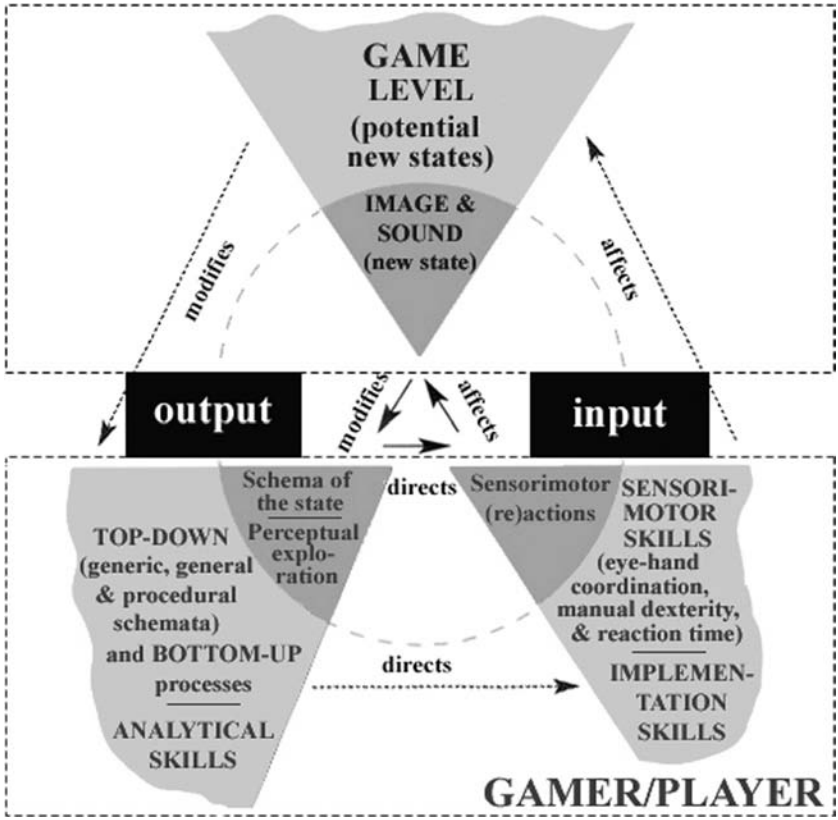

Fig. 5. The Heuristic Circle of Gameplay by Bernard Perron

stating that their base is under attack, while seeing a red ping that signals where the attack is will prompt them to click on the spacebar to quickly move the camera there. They will click on a control group to select one of their army and bring them where they are needed; the game will show the result of their actions on screen.

The outer circle works simultaneously: the potential new states in the player's mind will change how schemata are selected, and how sensori-motor actions will prioritize mechanics.[3] Their reaction to the

3. The word "mechanics" has a specific meaning in RTS play games. Liquipedia defines them like this: "Mechanics is your execution of micro and macro. Fundamentally, your mechanics, as a player, represent the degree to which you have bridged the divide between mind and game—that is, your ability, as a player, to do what you want to do" (TeamLiquid 2019a).

visual and sound cues were not innately inscribed in their mind; it is inscribed in a schema established because they *learned* that these cues mean they are attacked. If they know that their base is often attacked by small squads before they have time to build a proper army, they will have forged a schema in their mind to prioritize that choke points must be rapidly well guarded. Having executed repeatedly game mechanics such as moving armies fast will improve general sensori-motor skills and will open potential new states that takes advantage of these skills. This is not to say that variations on a strategy will not work. As one reviewer notes on *StarCraft*, the missions "are wide open to your particular style" (Coffey 1998, 168). Yet, in the decoding paradigm, some strategies will work while others will not, and the only way to confirm that a strategy works is by trial and error.

Let's analyze the scenario "Shadow Hunters," ninth episode of the original Protoss campaign, to show how Perron's heuristic circle of gameplay works. Protoss units face a Zerg army and must eliminate two Cerebrates, special buildings that are massive brains controlling the Zerg forces in the lore. As in a few previous episodes, the player begins without any building. The fact that they start with workers modifies the initial schemata: they must find a resource location to build a Nexus. They also start with a few Zealots and Dragoons—the first Protoss military units—but a new unit is introduced in this scenario: an Arbiter, a Protoss vessel that acts as a spellcaster and that "cloaks" surrounding units, rendering them invisible except if they are detected by specific units or buildings. They also have two hero units: Fenix, a Dragoon, and Zeratul, a Dark Templar that has a permanent cloak. If the player explores north, they will see Zerg creep and Sunken colonies defending the path; they will know that either east or west are better directions to go with their precious and fragile workers. Since a base takes some time to be established, the player can deduce that enemy attacks will not come in the first minutes; "setting up structures and researching upgrades" before building many units seems to be working (Kasavin 1998b).

The two resource locations are at the bottom of the map, but at the extreme opposites, and a choke point in the middle of the map redirects every ground unit movement. In fact, most strategy guides underline that ground attacks frequently come from the center: Photon Cannons (Farkas 1998a, 200) and/or Zealots (Dark Vortex 2007) should be positioned there. The map is quite large, and defending two bases could be difficult; yet, even flying units such as Mutalisks and Guardians will mostly attack at the choke point rather than at your mineral lines

(IGN-GameGuides et al. 2017), although Kasavin insists some Photon Cannons should also protect "the north edge of your base" (1998a).

The opponent uses Defilers for the first time in the campaign: it is "constantly" used (Dark Vortex 2007) and their Plague ability "bypasses Protoss shields and cuts straight to your hit points, which are irreplaceable" (Kasavin 1998c). Their Dark Swarm ability reduces to zero the attack score of ranged units within a large orange fog that has a long duration. While the static defense with Photon Cannon and Shield Batteries are assimilated as a good habit for defense, the Defiler adds a new challenge by forcing Protoss forces to move to avoid staying under the Swarm. The schema must be modified: static defenses are not enough when Defilers come into play. The introduction step by step of new abilities helps to learn the basic rules of game units that will also be at play in the foreseeing paradigm.

The IGN wiki identified that eliminating an Hatchery in the middle of the map will result in no more reconstruction of defensive buildings from the Zerg and fewer attacks on the player's bases (IGN-GameGuides et al. 2017). In the same vein, Greg Kasavin states that since most detector units are located north of the map, Zeratul can eliminate a lot of Sunken colonies, a Hatchery, Ultralisks, and the Ultralisk cavern, the latter being a way to "face far fewer Ultralisks over the course of the battle" (Kasavin 1998b). Here, only a specific decoding could tell us why an Ultralisk cavern cannot be built again when one is destroyed. Eventually, the player should have a sufficiently large force of Carriers to protect the center of the map (Dark Vortex 2007), while their main ground army supported by one or two Arbiters strikes the north bases one at a time.

Scrutating a map in such details shows us that it is built around specific challenges to confront existing strategic habits. If we were to oversimplify them, these challenges become sort of "puzzles" to overcome. In the decoding paradigm, strategic habits establish what can happen. If the campaigns are the quintessence of the decoding paradigm, it also works for custom games played against computer opponents. Decoding has been the dominant paradigm in the early history of strategy games.

Decoding the Origins of the Genre

In the 1980s, some wargames or strategy games were similar to contemporary RTS, in the sense that actions had to be conveyed quickly and without interruption. Most games set in a military context and with

an "arcade" aspect or with actions implemented under a certain stress (with or without a "pause" phase) could bear similarities with *StarCraft*. In general, the games with a strong verisimilitude with real war were called "wargames," while other more fantasy- or sci-fi-based were "strategy games" (see Dor 2019). *Eastern Front (1941)* (Chris Crawford 1981), *Stonkers* (Imagine 1983), *Combat Leader* (Strategic Simulations Inc 1983), *The Ancient Art of War* (Evryware 1984), or *Crusade in Europe* (MicroProse 1985) are common examples. These prominent cases are all single-player games: the decoding paradigm is paramount at this period.

In September 1990, *Computer Gaming World* starts its description of new strategy games presented at the *Consumer Electronic Show* like this: "Real-time strategy is becoming extremely popular" ("To 'Knight' The Knights" 1990, 76). *Star Control* (Toys for Bob 1990) is one of its example, even though today it would probably be called a turn-based strategy game with real-time combat sequences. In July 1991, Lawrence S. Lichtmann also begins a review underlining the importance of the genre, this time for *Overlord* (Probe Software 1990): "Real-time strategy games are a hot item right now" (1991, 58). These games are still quite different from *StarCraft* and its influences. Three games from the 1980s–1990s are direct influences on the decoding paradigm still at work in *StarCraft*: *Populous* (Bullfrog Productions 1989), *Dune II*, and *Warcraft*.

Populous is the prototypical "god game." Garth Fitzmorris describes it as a "real-time strategy game hit from Europe where the players fight cosmic battles from a quasidivine perspective" (1989, 40). Each level puts the player in the seat of a god managing a village and having to deal with a rival god managing their own populace. The god changes the environment to make it suitable for the villagers, which have their own autonomous agency: they build and upgrade houses, recruit new villagers, and fight mostly by themselves. Having a larger population unlocks new skills for the god to influence the world. Game reviews barely mention the multiplayer mode. *Mean Machines* state that it is "neither arcade nor true strategy" ("Populous" 1990, 52). While Rand and Glancey states that the 5,000 maps on the Master System version are very repetitive (1991, 105–6), Evan M. Brooks underlines indirectly the decoding aspect of the game when he insists that the maps are diverse, and that they "*require* a slightly different strategy" (1991, 37, emphasis mine). The goal is not to outwit an adaptative opponent, but to find the right strategy for each level.

The decoding paradigm is emphasized by the strategy guides. One of the most dreadful opponents seems to be the knight. Two guides from

GameFAQs suggest a strange way to counter the knight, aside from building a knight of your own: change the game options so that the water becomes fatal to units, and then drown the knight in the water using divine powers (Darth GGW 1996; Jabu-Jabu 2000). Jabu-Jabu asserts that this strategy is necessary to survive levels 50–72, while Darth GGW is quite realistic about the exploit that it is: "The only reason I myself don't always drown the computer's knight is because it takes the challenge away. The computer doesn't drown your knight" (Darth GGW 1996). The simple fact that a strategy works everytime to "outplay" the AI underlines how it can be decoded.

Dune II: Building of the Dynasty almost unanimously claims the title of the most influential RTS even though it was not qualified as a "real-time strategy" in 1992, nor even perceived as a ground-breaking strategy game at its release date (see Dor 2014a) (Fig. 6). It mostly corresponds to the checklist of RTS characteristics from the 1990s and onward, and clearly popularized them. Colin called it the "game that started it all," and underlined how it already has "three races, each with their own special weapons and missions" (1998) as in *StarCraft*. *Dune II* is often compared to *SimCity* (Maxis Software 1989) and *Populous*; in terms of decoding, they work similarly.

Fig. 6. *Dune II: The Building of a Dynasty*

Dune II is presented as the sequel to *Dune* (Cryo Interactive Entertainment 1992) while both games were developed simultaneously. The publisher Virgin Games had the rights for *Dune* and commissioned an adventure game from Cryo Interactive Entertainment but, losing confidence in this team, bought Westwood Associates—which will become Westwood Studios—to develop a strategy game. However, Cryo did not forfeit and decided to promote its future game on its own budget, calling the press (Ichbiah 2009, 198). The developer will later send an almost finished game to Virgin Games, who will publish both games the same year.

One of *Dune II*'s game designers, Joe Bostic, indicated in a more recent interview that they took inspiration from their earlier games, *Eye of the Beholder* (Westwood Associates 1991) among others, and from *Populous* to decide that action would take place in real time (quoted from *NowGamer* 2009). The player character is a commander from one of the three houses fighting for the control of spice on the planet Arrakis to gain the Emperor's trust. They must thus produce spice harvesters, construct buildings to manage a base, and create military units to defend them. Every house has some unique units compared to the others, which they will have to fight on the battlefield, alongside the sandworm attacks in the desert that target every faction. The expression "real-time strategy" seems to have been publicly introduced retrospectively by Westwood Studios with the release of their *Command & Conquer* series: a review of *Red Alert* notes that *Dune II* "created the real-time strategy category" (Broady 1996) while a writer from *Amazing Computing* states that it is the game that "installed the mantra 'real-time strategy' in the PC vocabulary" (Olafson 1997, 42). If the description of *Dune II* evokes the classical RTS, there is a fundamental difference: contrarily to *Populous*, it does not have a multiplayer mode.

The role of scouting in *Dune II* shows how it is inscribed in the decoding paradigm. A walkthrough suggests that the player should send their initial military units to scout in order to map the location of resources and the opponent's base location, and then to reload the game with this information in mind (DKennedy 1995). The time lost collecting the information is thus regained. The same walkthrough states that the player should not "explore too far until you have built a good defence up. The computer at the start works on a strategy of 'If you can't see him [the computer opponent], he can't see you!'." Of course, this strategy works because the computer has been programmed to respond to the player's actions. The strategy that Jeff James suggests to counter the

Death Hand is even more obviously one in a decoding paradigm. The Death Hand is a mass destruction weapon that is used by the opponent in late campaign levels, and that can pulverize more than one buildings with only one strike: James simply suggests to "save often" (1993, 112) to know where the strike will be and reload to minimize the damage. There is no way to foresee potential actions rather than having already seen them in a previous playthrough. As we have seen, *StarCraft*'s campaign works in a very similar way.

In 1991, Frank Pearce, Michael Morkhaime, and Allen Adham found Silicon & Synapse—which would quickly be renamed Blizzard Entertainment (Blevins 2001). Their first released games were ports, but they eventually developed their own original games for Nintendo consoles: *The Lost Vikings* (Silicon & Synapse 1993) and *Rock & Roll Racing* (Silicon & Synapse 1993).

The first Blizzard RTS game, *Warcraft: Orcs & Humans*, was mostly seen as a *Dune II* emule or copycat by game reviewers (Falcoz 1994, 148; *Coming Soon Magazine!* 1995; Lombardi 1995a, 228) and it was explicitly assumed (at least later) by their developers. It was acknowledged by Patrick Wyatt in a blog post that the obsession of their team for *Dune II* led to the production of *Warcraft*, especially because it was "obvious that this gaming style would be ideal as a multiplayer game" (2012a). Both games are similar up to a point that seemed very uncommon in 1994. Lombardi (1995a, 228) notes that a *Dune II* player can probably beat half of *Warcraft* without looking at the game manual, which says a lot about the role of manuals in the 1990s.[4] Its main original aspect is the multiplayer mode (Geryk 2001; Walker 2002a, 2; Fahs [2009] 2012, 1). *Warcraft* was quickly eclipsed by its sequel, *Warcraft II: Tides of Darkness*, that would be released less than a year later.

In *Warcraft*, each single player map offers a new challenge where previous strategic habits will not always work, offering new "puzzles" to be decoded. It "is a trial and error process requiring you restart a half dozen times before you figure it out" (Lombardi 1995a, 232). As with the *Populous* and *Dune II* examples, the precision of some gameplay description is unequivocally a decoding experience. Kang and Asher give tips for the fifth Orc level, stating that 12 spearmen are necessary: seven to

4. Cusick also underlines the crucial role of the game manual for *Dune II*: "You have to be prepared to spend time learning how to play games like this, although the lengthy manual is very helpful and easy to digest" (1993, 114).

defend the left bridge, five for the right bridge (Kang and Asher 1995, §5-2-5). While it is obviously arguable that this army composition is the only valid one, such a precision in stating the defensive position only makes sense if the opponent's strategy stays sufficiently stable. Their tips for the twelfth Human level were similar to the Ultralisk cavern underlined with our Protoss example earlier. Daemons attack the town periodically, but "[a]ll daemons are created by one single orc warlock, if you find him and kill him – no more ugly daemons to worry about" (Kang and Asher 1995, §5-1-12). If this behavior were not decoded, the player would suppose that their opponent would simply create a new warlock to summon daemons.

Some strategic habits seem to outlive a single level: such is the case of the "puller." The player must establish a defensive line in a choke point (ex: a bridge) and send a single unit that will drag the enemy into it. The opponent "will follow you to your 'ambush.' Works every time" (Lin 1995) (Fig. 7). Some levels will need variations of this strategy: for the tenth Orc level, Boehmer states that a conjurer will attack on the first few minutes, but once "you've setup your formation, just treat it *like any other level*, send in a puller, kill the defenders, and destroy" (Boehmer 2009,

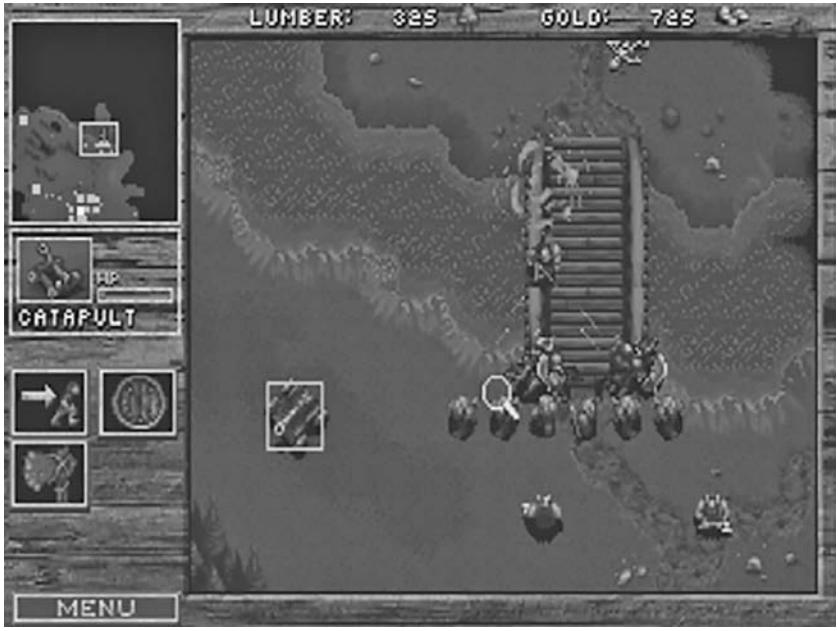

Fig. 7. *Warcraft: Orcs & Humans* and the puller strategy

§3.10, my emphasis). The simplicity of these behaviors is somehow balanced by the uneasiness of the controls and the fact that the computer evidently "cheats": for example, clerics can heal other units without any magic points restrictions.

The decoding paradigm has its legacy in other game genres. The player has to decode the pattern of boss fights in *The Legend of Zelda: A Link to the Past* (Nintendo EAD 1991) as in *Elden Ring* (FromSoftware 2022). The "die-and-retry" pattern has a lot in common with decoding, albeit maybe with a shorter cycle than in most strategy games. In role-playing games like *Octopath Traveler* (SquareEnix and Acquire 2018), one must anticipate the weaknesses of specific enemies and choose their actions wisely. In tower defense games like *Plants vs Zombies* (PopCap Games 2009), the player has to balance the growing of plants that generate energy and those attacking zombies with specific defensive devices. Decoding is a more precise way of describing one way to play strategically in games.

The decoding paradigm is strongly based on the fact that RTS games are games of "imperfect information" (Salen and Zimmerman 2004, 204). The opponent is hidden in the fog of war and knowing what it will do is one important aspect of the strategy. But, as Elliott Chin puts it in a strategy guide for *CGW*, "solo play leaves much of the richer strategy hidden" (1998, 236).

Michael Freed and his colleagues underlined how there is a wide gap between how a player plays whether their opponent is perceived as a human or as an AI:

> For instance, inhuman weaknesses in computer play encourage new players to develop tactics, prediction rules and playing styles that will be ineffective against people. Game designers often compensate for weaknesses in the computer's play by providing it with superhuman capabilities such as omniscience. However, such abilities render otherwise important tactics ineffective and thus discourage players from developing useful skills.
>
> (Freed et al. 2000, 1)

As soon as the player realizes that their opponent has omniscience and can see what their actions are, they do not develop the same skillset and game reflexes than when they play against a human opponent. To use James Paul Gee's expression (2004, 138), playing in single-player mode sets us straight on a "garden path" if our goal is multiplayer: the habits learnt for one situation are wrong habits for the other. As Dustin

Browder reminds us in the quote opening this chapter, "our solo campaigns have never prepared anybody for an online experience at all" (quoted in Remo 2009).[5] The skills needed to play in solo and in multiplayer are not the same. Consequently, creating the ruleset of *StarCraft* meant creating it for both paradigms.

The Foreseeing Paradigm

StarCraft can also be played in what the game calls "Custom Games." The player selects a map file (*.scm for *StarCraft* original, and *.scx for *Brood War*-only compatible maps) designed by Blizzard or by a member of the gaming community. While some maps are created with a precise experience in mind, most custom games retain the topographical elements of the battlefield and are meant to be experienced through the "melee" mode or its derivatives.

In this melee mode, a player starts with a single building and four workers. The goal is to destroy every building of your opponents. Maps are usually designed so that starting locations have a reasonable number of resources. They can be creative in terms of where other resources are reachable: a first expansion can be easily defendable or could need a flying transport unit to be reached. Up to eight players (including computer opponents) can fight together in a single game, and alliances with human players could theoretically be negotiated during the game. Other similar game modes ("Free for All," "Team Melee," "Top vs Bottom," "Capture the Flag") changed the general experience while maintaining the core aspect of the "melee" mode.

The "Ladder" mode is specific to multiplayer games and is the equivalent of ranked mode in other games. Players would join the Battle.net servers and play against opponents from different skill levels on maps specifically approved for this mode. Each victory would make the player

5. One reviewer of *StarCraft* though otherwise and, as Browder underlines, I remember that it was a relatively shared feeling at the time throughout the campaign: "I was left with the distinct feeling that the single player missions were simply training for people to play multi-player" (Colin 1998). Each map of the campaign would often introduce a new unit or a new mechanic to be learnt. For example, mission "Legacy of the Xel'Naga" of the *Brood War* Protoss campaign introduced the "Disruption Web" ability of the Corsairs to block attacks from ground units or buildings. Olafson echoes in some way Colin's criticism, but by underlining its virtue: "The designers let you into the game in a careful, gradual manner, in which business feels like fun and vice versa" (Olafson 2000).

earn some points, and each defeat would make them lose some, depending on the difference between their own ladder score—as an Elo rating system would do.

The foreseeing paradigm depends on custom games rather than campaigns. It is a game where players can foresee with more or less exactitude the future actions of their opponents to plan their strategy consequently. What is at stake is the anticipation of future actions by the observation of actual actions. In order to work, game rules must have some sort of predicting mechanism, for example, a technology tree. Foreseeing also relies on strategic habits. Each player, whether they are AI or human, could have been a human player. Foreseeing the actions of other players is possible because they could have been at their place.

The foreseeing paradigm can explain how gameplay is perceived in most multiplayer games. To win, they have to know what their opponents are up to and anticipate what they could do following game rules (having a Stargate means a Corsair can be in the game in 60 seconds, two Corsairs in 120 seconds, etc.) and following gaming habits—it is unusual, and therefore rarely foreseen, that a Scout will be out after a Stargate. In a classic playing-card game like the bridge, each set has the same cards (4 As, 4 Ks, 4 Qs, etc.). If I do not have a card in my hand, one of the other players must have it; anticipating your ally and your opponents' hands is crucial. Each decision would not be meaningful if it could not be foreseeable.

An RTS player can know beforehand that an attack can happen at, say, eight minutes of game time and anticipate approximately its force, since it could have been made by themselves if they were in their opponent's chair. They can, consequently, concentrate on countering this attack if it is possible. Ideally, designers of a strategy game in the foreseeing paradigm will make sure every efficient strategy can be scouted beforehand and can be countered by a strategy of some sort.

In this paradigm, game balance is not a question of game difficulty levels but of game players themselves. The goal is that every action has a way to be anticipated and can be—in some cases with high execution skills—countered. Historically, this paradigm emerges unsurprisingly with multiplayer games.[6] The arrival of modem games, where two

6. It is not to say that foreseeing is impossible with a computer opponent, but in *StarCraft* the AI was not efficiently coded to foreseeing: the opponent would be omniscient and respond to your actions whether they could normally see them or not. *StarCraft II* and other RTS games made it possible to some extent.

players can connect online through a telephonic line on their computer, can offer "authentic human competition as opposed to mere human-versus-computer action" (Brooks 1991, 37). No strategy is dominant in this ideal dynamic: the general game balance is more important than focusing only on a single player, since every actor in this dynamic can be a human player. In a two-player confrontation, for every winning player there is a losing player. Therefore, defeat is almost inevitable and core to the game.

The Heuristic Circle of Real-Time Strategy

Thinking of multiplayer strategy games as a paradigm where gameplay is "foreseeable" seems counterintuitive when it is clear that—as opposed to a computer opponent—a human player is mostly unpredictable. That is why a "foreseeing" paradigm is more suited to describe this experience than a "prediction" paradigm. In the *StarCraft* campaign, it is impossible to anticipate a future problem, except by trial and error, by learning the AI patterns and hoping they stick to them in other maps. The multiplayer experience is structured around foreseeability. Contrary to a single-player experience, reloading a game earlier is not possible; and, even if it were, it does not guarantee that an opponent will use the same strategy. Rather than being revealed by a previous experience, possible and plausible actions must be foreseen following game rules and strategic habits.

The Heuristic Circle of Real-Time Strategy will illustrate how the foreseeing paradigm works (Fig. 8).[7] Game states in the player's mind are of three kinds: (1) immediate, (2) inferred, or (3) anticipated states. These three states corresponds to: (1) the immediate seen and heard space; (2) a projection of the present unseen and unheard space; (3) a projection of the upcoming potential game states. An *immediate state* would be if a Zerg player perceives a Photon Cannon blocking a choke point in front of the Protoss expansion location. An *inferred state* would be the conclusion the Zerg player would make from what they saw, even if they did not see everything. The Photon Cannon has a prerequisite: a Forge. But they can also create an inferred state based on gaming habits: the Protoss probably has a Nexus protected by the Cannon to have an expansion operational. The player does not have to see the Nexus directly to

7. I presented this heuristic circle in previous research (in French in Dor 2010; in English in 2014b).

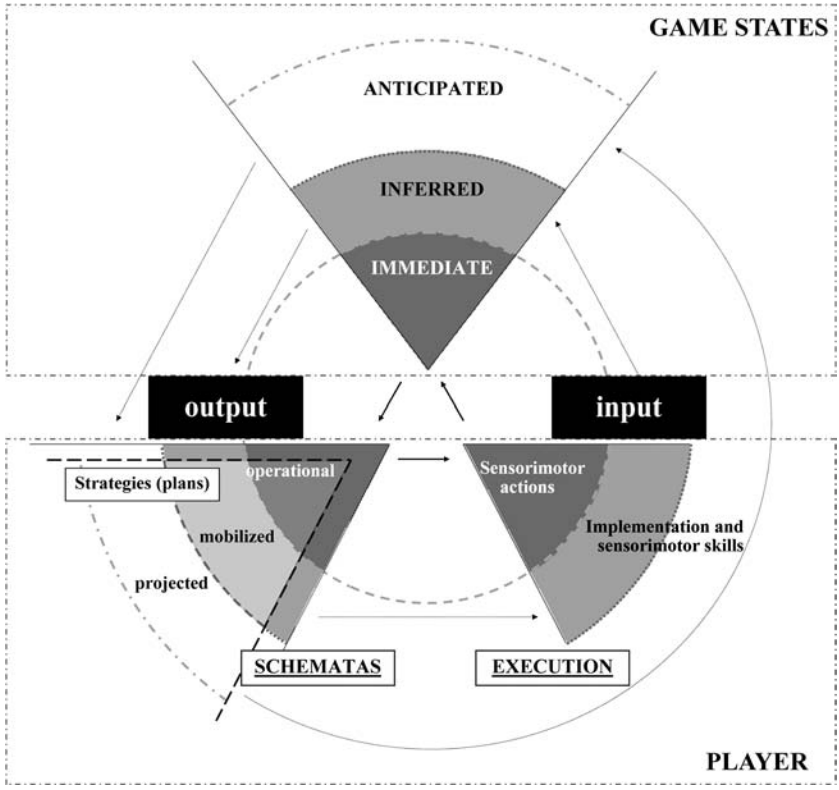

Fig. 8. The Heuristic Circle of Real-Time Strategy

know that it could be there. A third type of game state can come into play: the *anticipated state*. Following the fact that an expansion is established, the Zerg knows that the Protoss will not attack soon—for the cost of both an attack and an expansion is too high—but will have an economic advantage in a near future.

These three types of game states will help forge player's strategic plans, which I also describe in three levels: (1) operational (present and immediate); (2) mobilized (in short-term memory); and (3) projected (mostly stored in long-term memory). Following the anticipated state that an economic advantage could come for the Protoss, the Zerg could muster a *projected strategy*: to take an economic lead. This projected strategy helps to choose more narrow goals: *mobilized strategies*. I use this expression to describe smaller plans stored in long-term memory but rememorized as a task: take an expansion, protect the choke point, make a drop on the opponent's main base, etc. These mobilized strategies can be

mustered in a relatively small number at a time (in short-term memory) and can alternatively take the role of *operational strategy*. The strategy is operational when it *directly* guides the player's action. Expert players can very quickly alternate between different mobilized strategies, while novice players can have difficulties to operate more than one. Strategies and game states become so common in a player's mind that they create strategic habits to operate more quickly. Seeing that a Barracks is missing from a Terran's base automatically creates the logical inferred state: that Barracks must be hidden somewhere, and an attack is coming soon. The player's mobilized strategies will change to respond to that attack.

Let us see how they unfold dynamically. I will describe the skirmish between Korean Protoss player Doh "Best" Jae Wook and Korean Terran player Jun "Midas" Sang Wook, on the map "Grand Line SE," during the 2009–2010 Shinhan Bank Proleague (nevake 2010c). From an external observation, Midas' projected strategy is to have an economic advantage. He mobilizes at least two strategies: build a quick expansion and have a few military units. Best chooses a standard build order. His projected strategy is to be moderately aggressive at the beginning of the game. Amid his mobilized strategies, he has: (1) the harassment of Terran units with a Zealot; and (2) his standard opening. The Zealot supported by the scouting Probe managed to delay the Terran expansion by eliminating the SCV building the Command Center. Best retreats to keep his Zealot alive, but adapts his plan by sending another Zealot.

Midas then adopts an audacious projected strategy that goes against the current habits in Terran versus Protoss: to have an army composition of infantry, supported by a few Tanks, to take his opponent by surprise. To do so, Midas mobilized three plans: (1) defend his expansion by building a bunker; (2) progress in the tech tree by building a Factory and an Academy; and (3) grow his unit production by having four Barracks. Since Best knows that Midas is securing an expansion, he changes his projected plan: he will seek an economical advantage and keep map control. Best has three mobilized plans: (1) send a Dragoon and developing the Singularity Charge technology to attack the Terran bunker without a counter-attack; (2) build two quick expansions; and (3) build a Robotics Facility and an Observatory to have Observers. Since he does not know what is in the Terran's base, Best's mobilized plans are not really adapted to a quick attack. But, since Midas does not know either that Best has three bases, he attacks too late and cannot take advantage of a military superiority.

Best has enough time to have four Gateways and a Reaver before Midas' army strikes. Best takes Midas by surprise by engaging the combat: the Reaver quickly eliminates most infantry units before Dragoons join the fight. After this exchange, Protoss aims to contain the Terran player to his two bases, while Best will secure a fourth base. His economic advantage will translate into a larger army than Midas, while having the units more adapted to counter the Terran's infantry. Having possible expectations and playing with their opponent's expectations is at the core of foreseeing play.

Foreseeing the Future of the Genre

Multiplayer games are not all encompassed by the foreseeing paradigm. Don Daglow's *Utopia* (Mattel 1981) bears similarity with a lot of RTS. Two rival islands must manage their survival against natural disasters and their opponent's sabotage or naval skirmishes. Yet the game is a game of perfect information since an action cannot be hidden. Therefore, the foreseeability of an action is not necessarily an important aspect of the game. The same can be said of *Herzog Zwei* (TechnoSoft 1989), another one of the numerous games identified as the first RTS (Geryk 2001; Shaka 2001). The player's mech can morph in a plane and controls numerous bases to recruit new troops until everything is conquered (Fig. 9). Multiplayer matches are possible on a split screen; it is thus not a game of foreseeing. Strategy is seen as "limited" (Lapworth 1990), even if "everything happens in real time" (Glancey 1990, 103).

Unsurprisingly, multiplayer games were extremely rare before the advance of the Internet. There were early precursors like game designer Danielle Bunten Berry and her company, Ozark Softscape. Her vision for game design was very much focused on multiplayer experiences, whether they were offline like with *Computer Quarterback* (Bunten Berry 1981) and *M.U.L.E.* (Ozark Softscape 1983), or through modem play like with *Modem Wars* (Ozark Softscape 1988), to the point that she would be called the "Modem Master" (Emrich 1992).

Modem Wars is often identified as a predecessor for RTS games (Donovan 2010, 300), when it is not directly called an RTS (Gorenfeld 2003). *Modem Wars* was supposed to be titled "Sport of War" (Hockman 1989, 32), for the game dynamic is very similar to a sport like football. Each player starts with an army of grunts (infantry), riders (cavalry), boomers (artillery), and spies (reconnaissance units), and no new units

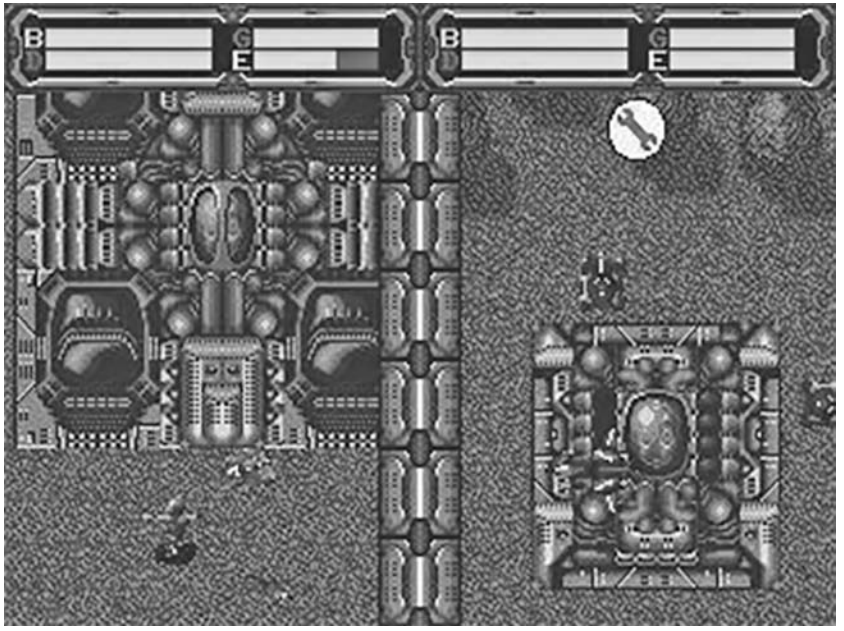

Fig. 9. *Herzog Zwei*

can join the combat. Various maps will modify terrain, starting units—often asymmetrical—and certain game rules, such as a preparation phase. Clicking on a unit and then at its destination will make it move and attack units in range automatically. Topography (hills, forest, etc.) changes line of sight and combat results. Rather than a minimap showing the whole space, the map occupies the majority of the screen while a smaller frame shows the specific space around the cursor (Fig. 10). The goal is to eliminate the opponent's command center.

In one of the only game reviews published around the release of the game, Daniel Hockman focuses on its competitive aspect, on the imperfect information of the game and the tension it creates, and on the "ability to develop strategy and tactics in real time" (1989, 32). He underlines that there are groups on the Quantum Link online service to play *Modem Wars* (1989, 32). The rarity of modems was probably the main reason why it did not reach commercial success (Gorenfeld 2003; Lowood 2008, 181; Donovan 2010, 300; Baker n.d.). Hockman even insists that the game is not worth it if it is only played against the computer (1989, 33), the solo mode being explicitly called "Practice with solo trainer." There was also

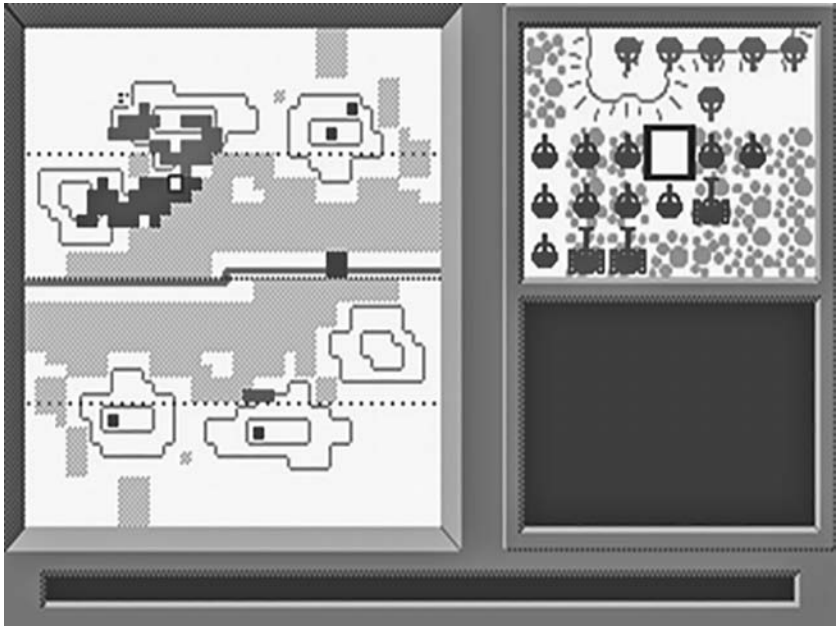

Fig. 10. *Modem Wars* (DOS version, emulated through DOSBox)

a "replay" function in the game, which let players watch their past games to improve themselves. It is also an indicator that it was designed for competitive play. Gameplay is more a question of positioning and tricking the opponent to engage in the wrong battles. Its gameplay is based on a certain foreseeing, but not necessarily as later RTS games would be since the anticipation is not based on any technological and economical progression. Bunten Berry will follow with *Command H.Q.* in 1990 and *Global Conquest* in 1992, which will reach a certain critical success but fail to be strong commercial hits.

The History of Two Paradigms

Modem Wars preconized multiplayer gaming, while the descriptions surrounding *Herzog Zwei* and *Populous* underlined solo gaming, even though all three games allowed both gaming types. The case of *Warcraft: Orcs & Humans* is more complex since both decoding and foreseeing paradigms are acknowledged in game reviews. For Thierry Falcoz, multiplayer gaming is sufficient to say that *Warcraft* is excellent (1994, 148). Travis Fahs from *IGN* describes exactly how multiplayer makes the translation from

decoding to foreseeing: "What was once a game of patterns became a battle of wits" ([2009] 2012, 1).

1995 saw the *Dune II* and *Warcraft* model being consolidated in game reviews and more clearly defined. The model of what RTS will be is slowly constructed with more precise conventions. While he acknowledges the direct influence from *Dune II*, Chris Lombardi from *Computer Gaming World* links *Warcraft* to Danielle Bunten Berry's games: "Here is where Warcraft really comes alive! Fast-paced, fun, and flexible enough to support a wide variety of tactics, Warcraft ranks up there with such classic two-player slug-fests as Command HQ and Global Conquest" (Lombardi 1995a, 232). He notes that the game needs an alternance between a "long-term planner and octopedal micro-manager with a quick but steady mouse hand" (Lombardi 1995a, 230). But it is in a preview of *Command & Conquer* that Lombardi uses the expression "real-time strategy" probably for the first time, albeit using "real-time" in quotation marks. He describes quite clearly RTS as a game genre:

> These games are very similar to your typical war and strategy game except that they don't afford the luxury of time to plot your moves. You give a command to a unit and it responds. Bang! There's no time to calculate attack factors, no counting movement points, no such thing as a well-considered stratagem. You make your decision now, or the enemy will be climbing down your throat. If you make the wrong decision, well, you quickly assess and adjust.
>
> (Lombardi 1995b, 32)

Still, it is quite difficult to know how most *Warcraft* or *Command & Conquer* plays would unfold in households. The game speed is influenced by the CPU cycles of the computer running it; there is no "canonical" speed. It is also quite complex to know how long a game would last or what were the most common strategies.[8] *Warcraft* shows how a game is more than the sum of its features: it can be a quasi-plagiarized version of *Dune II*, and yet build on the tradition of Bunten Berry's multiplayer games.

8. It is still interesting to note that a discussion that took place on a Bulletin Board System forum in November 1994—the month when the game was released—a user notes that they never have time to go beyond footmen/grunts and archers/spearmen units in the tech tree, while another one is surprised since they always have on their side catapults and spellcasters "in under an hour" (Bob Kusumoto, responding to Hulsey 1994). Considering that an hour-long RTS game is exceptionally long today, we can only imagine how long strategy games were in general in 1994.

In an interview with Jack Sorensen, president of LucasArts, in the May 1996 edition of *PC Powerplay*, the magazine notes how "there are *far too few*" real-time strategy games, with which Sorensen answers that the new ones "won't be nearly as good" as the originals like *Command & Conquer* (Sorensen 1996, 33, emphasis mine). In June 1998, two months after the release of *StarCraft*, the same magazine covered the development of *Star Wars: Force Commander* (LucasArts Entertainment Company 2000). Ironically (or not), their discourse is radically opposed: "You'd think that the real-time strategy game onslaught might have abated by now, but no—they just keep coming" (St John 1998, 16). In two years, RTS passed from an original genre which needs new iterations to an "onslaught" of new titles. In the 1998 context, *StarCraft* was clearly not an innovative step in the strategy gaming field. Yet, as we will see in the next chapter, it occupies a strange place in between innovation and conservatism.